DISSERTATION

Emulation and Simulation of Microelectronic Fabrication Processes

ausgeführt zum Zwecke der Erlangung des akademischen Grades
eines Doktors der technischen Wissenschaften

unter der Betreuung von

Asst.Prof. Privatdoz. Dr.techn. Lado Filipovic, MASc.
O.Univ.Prof. Dipl.-Ing. Dr.techn. Dr.h.c. Siegfried Selberherr

eingereicht an der Technischen Universität Wien
Fakultät für Elektrotechnik und Informationstechnik
von

Xaver Klemenschits, MSci.
Matrikelnummer: 11708463

Wien, im Februar 2022

Abstract

The fabrication of increasingly powerful microelectronic processors, enabled by transistor scaling, has been a main driver of technological progress in most fields since the 1950 s. Until recently, this scaling of electronic components was mainly achieved through improved control of process conditions, rather than innovations in fabrication techniques. However, with the introduction of new materials and three-dimensional transistor structures, processing techniques have become highly complex and expensive. Therefore, process technology computer aided design (TCAD) has become indispensable for manufacturers in order to evaluate possible future technologies. The simulation of manufacturing processes has already become essential in modern design technology co-optimisation (DTCO) cycles which are used to produce the next generation of smaller, more efficient and more performant semiconductor circuits.

The capabilities of a process simulator are often restricted by the numerical methods underlying its operation. Most simulators employ the level set method for the description of evolving material interfaces during manufacture. However, sharp edges, which might occur during epitaxial crystal growth, cannot be handled appropriately due to fundamental limitations of this method. In order to solve this problem, a novel numerical scheme for the exact description of crystal facet evolution was developed within this work, which finally allows epitaxial processes to be described accurately within a level set description.

Recently, the emulation of process steps, which aims to reproduce the geometric outcome of processes rather than simulating the underlying physics, has become increasingly important for advanced DTCO cycles due to its high computational efficiency. Using emulation, complex transistor structures can be generated within seconds, allowing for the fast evaluation of new transistor geometries using combined device and circuit simulations. Within the scope of this work, for the first time, the emulation of fabrication processes in the level set has been made possible through the design and implementation of a geometric advection algorithm. This algorithm allows for large changes in material interfaces to be modelled in a single step.

The development of these fundamental techniques and their implementation in a single process modelling framework, ViennaPS, combines highly physical process simulation capabilities with computationally efficient process emulation and thus allows for a full description of modern and possible future manufacturing techniques. Therefore, this work provides the missing link between process emulation and simulation, finally enabling the unrestricted combination of both methods to accelerate DTCO cycles and thus the discovery of novel technologies for semiconductor fabrication.

Kurzfassung

Die Herstellung von immer leistungsstärkeren Mikroprozessoren durch die Verkleinerung von Transistoren ist bereits seit den 1950 er Jahren ein Haupttreiber für technologischen Fortschritt in fast allen Bereichen. Bis vor Kurzem konnte diese Verkleinerung hauptsächlich durch bessere Kontrolle von Prozessbedingungen und ohne drastische Änderungen der Herstellungstechnik erzielt werden. Durch die Einführung von neuen Materialien und dreidimensionalen Transistorgeometrien wurden diese Prozesse jedoch komplex und teuer. Daher wurde Technology Computer Aided Design (TCAD) für Hersteller unverzichtbar, um mögliche Technologien der Zukunft zu evaluieren. Die Simulation von Herstellungsprozessen ist daher essenziell für moderne Design Technology Co-Optimisation (DTCO) Zyklen, mit denen die nächste Generation von kleineren, effizienteren und leistungsstärkeren Halbleiterschaltkreisen produziert wird.

Die Fähigkeiten eines Prozesssimulators sind häufig durch die verwendeten numerischen Methoden eingeschränkt. Die meisten Simulatoren basieren auf der Level Set Methode um bewegte Materialoberflächen während der Herstellung zu beschreiben. Jedoch können scharfe Kanten, welche in Epitaxieverfahren entstehen können, durch fundamentale Limitierungen dieser Methode, nicht angemessen beschrieben werden. Um dieses Problem zu lösen, wurde in dieser Arbeit ein neuartiges Advektionsschema für die genaue Beschreibung von Kristallflächen entwickelt, welches es möglich macht, Epitaxieverfahren in einem Level Set zu beschreiben.

Die Emulation von Herstellungsschritten, welche statt der grundlegenden Physik eines Prozesses nur das geometrische Ergebnis beschreibt, hat durch seine hohe Effizienz immer mehr an Bedeutung für DTCO Zyklen gewonnen. Mit Emulationen können komplexe Transistorstrukturen innerhalb von Sekunden generiert werden und neue Transistorgeometrien schnell, mithilfe von Bauteil- und Schaltungssimulationen, evaluiert werden. In dieser Arbeit wurde, zum ersten Mal, die Emulation von Herstellungsprozessen direkt in einem Level Set durch die Entwicklung eines geometrischen Advektionsalgorithmus ermöglicht. Dieser Algorithmus erlaubt es, große Veränderungen von Materialoberflächen in einem einzige Schritt zu modellieren.

Die Entwicklung fundamentaler Methoden und deren Implementierung in einem Prozesssimulator, ViennaPS, kombiniert physikalische Prozesssimulation mit effizienter Prozessemulation und erlaubt so eine ganzheitliche Beschreibung von modernen und potentiell zukünftigen Herstellungsverfahren. Daher konnte mit dieser Arbeit das fehlende Glied zwischen Simulation und Emulation geschaffen werden, welches das uneingeschränkte Zusammenwirken beider Methoden ermöglicht, um DTCO Zyklen und die Entdeckung neuartiger Halbleiterfertigungsmethoden zu beschleunigen.

Acknowledgement

Firstly, I want to thank Prof. Siegfried Selberherr for providing a working environment which harbours critical thought, lively discussions, and not only scientific but also personal progress.

Furthermore, I want to thank Dr. Lado Filipovic for being open to new ideas, for his help in formulating raw ideas into scientifically sound theories, and for making the pursuance of these ideas possible with supreme academic, organisational, and personal support. Without his supervision, this work would not have been possible.

Additionally, my gratitude goes to Paul Manstetten for lively, and sometimes almost heated, technical discussions which I believe lead to great professional improvement in my daily work, and to the creation of several lectures of the highest quality.

I do want to thank the entire staff of the Institute for Microelectronics for their support over the years, and especially Alex Toifl for his perseverance in perfecting even the basic foundation of our field which often sparked my own ambition, and Markus Kampl for his unshakeable ethics in developing reusable and open software which I have always found inspiring.

Finally, I want to thank my father who earned his doctoral degree at TU Wien almost 30 years ago and often provided a helpful outsider's perspective on the scientific problems encountered in this work.

Contents

List of Figures

List of Tables

List of Algorithms

List of Abbreviations

1D	one-dimensional
2D	two-dimensional
3D	three-dimensional
ALD	atomic layer deposition
ARDE	aspect ratio dependent etching
BEOL	back-end of line
BJT	bipolar junction transistor
CAD	computer-aided design
CF	fluorocarbon
CFL	Courant-Friedrichs-Lewy
CH_2F_2	difluoromethane
CMP	chemical mechanical planarisation
CORE	clear-oxidise-remove-etch
CPU	central processing unit
CS	cell set
CVD	chemical vapour deposition
DC	direct current
DEM	deposit-etch-multiple-times
DREAM	deposit-remove-etch-ash-multiple-times
DREM	deposit-remove-etch-multiple-times
DRIE	deep reactive ion etching
DTCO	design technology co-optimisation
EUV	extreme ultra-violet
FEOL	front-end of line
FET	field-effect transistor
FIB	focused ion beam
GAA	gate all-around
GPU	graphics processing unit
HBr	hydrogen bromide
HCl	hydrogen chloride
HfO_2	hafnium dioxide
HKMG	high-k metal gate
HRLE	hierarchical run-length encoded
IC	integrated circuit

ID	identifier
ILD	interlayer dielectric
LLF	Local Lax-Friedrichs
LLLF	Local Local Lax-Friedrichs
LS	level set
MBE	molecular-beam epitaxy
MC	Monte Carlo
MEMS	microelectromechanical systems
MOSFET	metal-oxide-semiconductor field-effect transistor
PDE	partial differential equation
poly-Si	polycrystalline silicon
QP	quadruple patterning
RF	radio frequency
RIE	reactive ion etching
RMG	replacement metal gate
S/D	source and drain
SADP	self-aligned double patterning
SDF	signed distance function
SEG	selective epitaxial growth
$\mathbf{SF_6}$	sulphur hexafluoride
$\mathbf{Si_2H_2Cl_2}$	dichlorosilane
$\mathbf{SiF_4}$	silicon tetrafluoride
SiN	silicon nitride
$\mathbf{SiO_2}$	silicon dioxide
SLLF	Stencil Local Lax-Friedrichs
SRAM	static random access memory
STI	shallow trench isolation
TaN	tantalum nitride
TCAD	technology computer aided design
TEOS	tetraethyl orthosilicate
TiN	titanium nitride

Chapter 1

Introduction

Semiconductor devices are omnipresent in today's everyday life. Phones have become high performance computers, there are microchips in almost all objects of daily use ranging from cars to bicycles, from washing machines to fridges, and having microelectronic sensors and life preserving devices implanted in humans cannot be considered unusual anymore. This abundance of semiconductor technology in our lives has been made possible by the decades-long, continuous miniaturisation of electrical circuits in accordance with Moore's Law [1]. The invention of the bipolar junction transistor (BJT) in 1948 [2] and the monolithic silicon integrated circuit (IC) in 1959 [3] paved the way for decades of ever smaller electronic devices shaping modern society. Also in 1959, the metal-oxide-semiconductor field-effect transistor (MOSFET) was invented and led to the transistor becoming the most frequently fabricated object in human history [4]. For decades, MOSFET design did not change drastically [5], while its size could be decreased due to improved fabrication techniques, such as more focused lithography processes thanks to better optics [6]. Lower power consumption and faster switching times, and thereby improved performance, were achieved through smaller dimensions and thinner insulating layers, meaning that smaller concentrations of electrical charges were required for switching. However, starting in the early 2000 s, new fabrication techniques, such as strain engineering [7], new materials [8] and three-dimensional (3D) geometries [9] became essential to keep improving upon the previous scaling processes. Nowadays, processors made of billions of single transistors contain numerous different materials in complex 3D arrangements, fabricated in meticulously calibrated steps performed at highly stable temperatures, pressures and gas concentrations for well-defined time periods.

The required reliability and reproducibility of the complex fabricated structures made it necessary to develop physical models of devices and their operation from the beginning of transistor fabrication [10]. Starting in the 1960 s, the simulation of complex models on computers was becoming increasingly important in order to improve the understanding of the underlying physical processes at play and to increase the quality of the fabricated devices [11]. As the fabrication techniques and the produced devices became more complex, so did their modelling and simulation approaches. By the 1970 s, modelling the fabrication and operation of semiconductor devices, referred to as technology computer aided design (TCAD), had become an essential and widely used tool for manufacturers. This trend was only accelerated in later years and proper modelling is now crucial in all areas of microelectronic device development, including fabrication, circuit design, as well as operation and long-time reliability. The inclusion of technological effects in the manufacture has become especially important, leading to several advances in design technology co-optimisation (DTCO), which relies heavily on TCAD.

In order to better separate the wide range of modelling approaches, the simulation of fabrication steps such as etching, deposition, ion implantation and oxidation is referred to as process TCAD, while the simulation of the electrical characteristics of the fabricated devices is referred to as device TCAD [12, 13]. In a modern DTCO cycle, the results of process TCAD are used in device TCAD to generate accurate descriptions of single devices, which are then used in circuit TCAD to simulate their interaction in large electronic circuits. In this way, entire processors can be simulated to ensure that their desired functionality and reliability are achieved [14].

As process TCAD is the first step in modern development cycles, it is crucial that process-induced effects are well understood in order to ensure the reliable manufacture of modern devices and circuits. Due to the ever smaller sizes, even small process deviations may lead to the catastrophic failure of components. Therefore, meaningful descriptions of each process step are critical for the correct prediction of the resulting fabricated structure.

Additionally, the fast development cycle of modern processors requires that large circuits and the potential interference of devices spaced closely on a microchip can be described in short time frames. Hence, the computational efficiency of fundamental simulation methods, as well as process models, is key for modern microelectronic device development.

Therefore, the main focus of this work is centred on process TCAD. Specifically, the physical description of several different deposition and etching processes is considered, sometimes referred to as topography simulation. Fundamental simulation concepts for the description of these process steps, as well as intricate physical models are developed and applied to advanced device structures. In order to meet the requirements of time-limited DTCO development cycles, efficient approximations of these models are developed to allow for the fast evaluation of new circuit designs.

In this chapter, a general process flow for semiconductor fabrication is presented, highlighting the main steps this work will focus on, followed by the motivation and goals of this research, and an outline of the thesis.

1.1 Semiconductor Fabrication Process Flow

The fabrication of semiconductor devices consists of many steps which must produce well-defined structures with little deviations from one device to the next. Previous and subsequent fabrication steps must be taken into account at every step to ensure the required high fabrication quality is achieved. Most processes are carried out in vacuum chambers by introducing different gaseous chemicals, which modify the surface by reacting with the substrate. The formation of the smallest features of an

IC is referred to as front-end of line (FEOL) [15], which most of this work will focus on. The most important process steps during FEOL fabrication are explained in the following, using the process flow of the 22 nm technology node for the fabrication of the FinFET shown in Fig. 1.1.

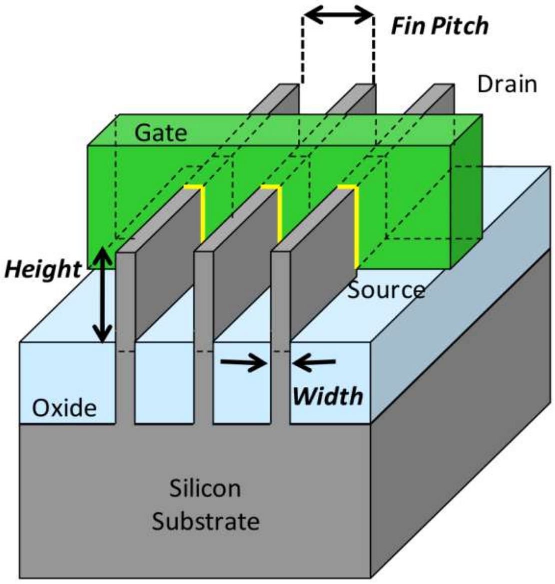


Figure 1.1: FinFET of the 22 nm technology node with a single gate and three source and drain contacts, respectively. ©2012 IEEE [16]

Illustrations of the most important fabrication steps are shown in Fig. 1.2. In-depth descriptions of how each of these steps can be modelled will be presented in Chapter 5. The FEOL steps of the fabrication of a FinFET usually are similar to the following:

1. The process initiates with a cleaned and polished crystalline silicon wafer.

2. **Photolithography**: In order to form the future conducting channel, i.e. the fin, a mask is created using photolithography, whereby only certain regions of the waver are covered by the mask material. Lithographic processes have a maximum resolution which depends on the wavelength of light used. The smallest possible dimension which can be achieved with the commonly used ultra-violet light of 193 nm wavelength is around 30 nm [17]. In order to achieve smaller features, self-aligned double patterning (SADP) is used, which generates smaller masks by depositing a thin layer isotropically using chemical vapour deposition (CVD) [18]. This thin layer is then etched directionally using reactive ion etching (RIE), leaving behind thin pillars of deposited material which are then used as the mask, as shown in Fig. 1.2a. Repeating this process a second time using the resulting pillars as the initial mask is referred to as quadruple patterning (QP). In order to avoid these complicated multiple patterning techniques, some manufacturers have already implemented extreme ultra-violet (EUV) photolithography [19].

3. **Fin Patterning**: At this stage, the crystalline silicon substrate is etched, leaving fins below the mask, which will later form the conductive channel of the MOSFET. As indicated in Fig. 1.2b, the etch process is tuned so the fins are positively tapered and thus increase in width towards the bottom.

4. **Shallow trench isolation (STI)**: Since each fin may be part of a different transistor, they must be electrically insulated from each other. This is achieved by depositing a dielectric material everywhere. The wafer is then polished using chemical mechanical planarisation (CMP) to create the flat top surface shown in Fig. 1.2c.

5. **Fin Release**: Once the surface has been polished, the dielectric can be etched selectively, leaving the crystalline silicon intact. The fins are therefore released again and reach out of the dielectric as shown in Fig. 1.2d, while being isolated from each other to a sufficient depth.

6. **Metal Gate Deposition**: A high-k metal gate usually consists of a stack of three different materials: A thin layer of a material directly around the fin with a high dielectric constant, often denoted κ, hence the name high-k; Another slightly thicker layer of a metal, referred to as gate metal; And finally a much thicker layer of a gate contact material, usually polycrystalline silicon (poly-Si). These materials are deposited using either CVD or atomic layer deposition (ALD).

7. **Gate Patterning**: On top of the gate contact material, another mask is created using photolithography. The gate materials are then etched, one after the other, leaving the gate only covering the centre part of the silicon fins, as shown in Fig. 1.2e.

8. **Gate Spacer**: In order to electrically isolate the gate from the source and drain (S/D) regions, a spacer dielectric is deposited isotropically using CVD. Similar to SADP, it is then removed directionally by RIE, leaving the polymer only on the side of the gate materials. As can be seen in Fig. 1.2f, RIE is not selective and the silicon fins are also eroded during this fabrication step.

9. **Fin Recess**: The silicon fins are then etched in order to clean them in preparation for the subsequent process, leading to an under-etch of the fins under the dielectric spacer, as shown in Fig. 1.2g.

10. **Source and Drain Epitaxy**: Now that the surface of the fins has been cleaned, crystalline S/D contacts can be created using epitaxial growth. The rate of this growth varies strongly with the crystal direction resulting in the characteristic diamond shapes observed in Fig. 1.2h.

11. **Interlayer Dielectric**: Another layer of dielectric material is deposited to isolate the S/D contacts before the wafer is polished using CMP, resulting in the final MOSFET structure shown in Fig. 1.2i, concluding the FEOL fabrication.

After the FEOL steps are completed, all transistors are connected to form electric circuits during metallisation, which is part of the back-end of line (BEOL). This is performed in several layers, each layer increasing in size, ultimately providing metal contacts large enough to connect the IC to peripheral components. A single wafer may contain hundreds of chips which are manufactured simultaneously. Since all transistors and interconnects are now formed, the wafer can be cut into single chips. Each chip is then packaged with connectors suitable for the specific application.

1.2 Motivation and Research Goals

All the discussed processing steps must be configured to tie perfectly into the entire process flow in order to yield robust device performances. Since carrying out physical fabrication is very expensive, especially at advanced nodes, the simulation of these process steps can help to understand common problems encountered in the fabrication and thus can strongly decrease the cost of optimising manufacture and designing new devices. It can also assist in tuning the fabrication settings and providing an insight into fabrication-induced variability and reliability.

This work builds on simulation efforts conducted previously at the Institute for Microelectronics, TU Wien, based mainly on the topography simulator ViennaTS [20], which employs the level set (LS) method to describe material interfaces. However, due to design restrictions, this simulator does not allow for the description of volume properties of materials, which is crucial for the modelling of certain fabrication processes. Furthermore, the LS method does not allow for the efficient emulation of fabrication processes. This means, that a process must be modelled in time and material interfaces moved discretely. However, for certain processes it is much more efficient to simply describe their geometric effect on a structure, which is referred to as emulation [21]. Additionally, due to the software design of ViennaTS, it cannot be combined with other simulators straight-forwardly to create a full DTCO toolchain.

Therefore, the goal of this work is to create a broadly applicable high performance simulation framework for the modelling of semiconductor manufacturing processes. This includes fast structure generation using emulation methods, as well as highly sophisticated physical simulations. This framework will allow for the fast creation of masks and initial geometries for microelectronic fabrication processes. The development of new fundamental methods are required to perform process emulation directly on a level set, which is one of the main goals of this work.

Additionally, a modelling framework for highly physical process descriptions, including transport, surface and volume reactions, as well as their effect on material interfaces will be implemented. This requires the application of stochastic methods, such as ray tracing, to model the transport of chemical reactants above the wafer. This allows for the straight-forward development of physical models with great flexibility regarding the computational methods required for the description of physical phenomena. This framework is then used to develop and apply sophisticated physical models, as well as efficient emulation models, describing the etching and deposition of materials during the manufacture of semiconductor processors.

1.3 Outline of the Thesis

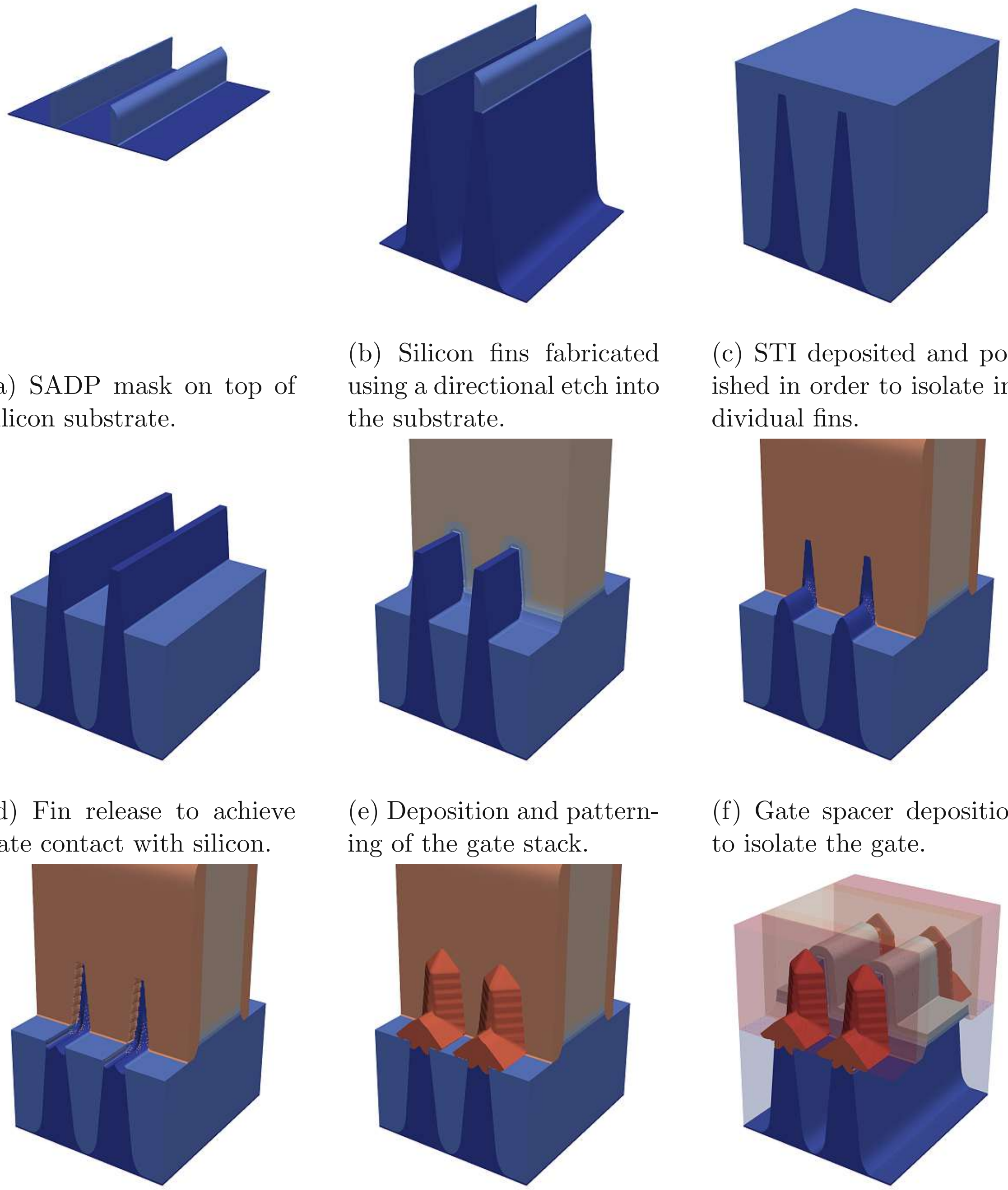
The fundamental mathematical concepts and numerical methods required for the modelling of semiconductor fabrication simulations are presented in Chapter 2. These include numerical material and interface representations, as well as the description of material evolution in time. Common modelling approaches and their mathematical foundation are presented, highlighting their relative benefits and limitations. A comprehensive review of the LS method and different variants thereof is presented, as well as commonly applied discretisation schemes for space and time.

The fundamental concepts for modelling particle transport in chemical reactors are introduced in Chapter 3, focussing on the specific requirements for semiconductor manufacturing and the related processing equipment used in this field.

In Chapter 4 the implementation of the numerical concepts introduced in Chapter 2 and Chapter 3 within the software toolchain developed in the course of this work is presented in detail. The applied algorithms and implementation details thereof are provided, highlighting potential pitfalls and limitations of the underlying methods when applied to the simulation of semiconductor fabrication processes. Results generated by individual components of the software toolchain are presented to guide the reader visually and show the capabilities of the presented simulation framework.

Chapter 5 includes a collection of complete process models, discussing their physical and chemical foundation and input parameters. Starting with models for CVD, selective epitaxial growth (SEG) and several wet and plasma etching processes, more complex fabrication steps, such as the Bosch process, are presented. Each model is discussed in detail, including physical mechanisms dominating the process, as well as the required input parameters and results for typical structures. Furthermore, the fundamental difference between process emulation and process simulation models is presented using the resulting geometries of the respective process models. A number of process flows for entire devices and circuits are presented thereafter. These include the fabrication of a replacement metal gate (RMG) FinFET at the 22 nm technology node, as it is currently produced in many industrial applications, and a stacked nanowire transistor which is a commonly proposed solution for transistor scaling beyond the 5 nm technology node.

Finally, Chapter 6 provides a summary of the findings of this work, concluding with an outlook for future directions of research.



(a) SADP mask on top of silicon substrate.

(b) Silicon fins fabricated using a directional etch into the substrate.

(c) STI deposited and polished in order to isolate individual fins.

(d) Fin release to achieve gate contact with silicon.

(e) Deposition and patterning of the gate stack.

(f) Gate spacer deposition to isolate the gate.

(g) Fin recess.

(h) S/D epitaxy.

(i) Final FinFET structure.

Figure 1.2: FEOL process steps for the fabrication of a 22 nm FinFET.

Chapter 2

Numerical Modelling

In this chapter, the computational methods required for the description of evolving material surfaces and interfaces are discussed.

First, the length scales involved in semiconductor manufacturing and the modelling approaches required and applied at these scales are discussed. Then, fundamental modelling approaches are reviewed and relevant assumptions about the modelled material surfaces and their interfaces are presented. This includes a description of continuum material descriptions and a motivation for their use in this work.

Different material representations for the continuum regime are introduced next, including a thorough review of the level set method which is applied for the modelling of material interfaces in this work. Different volume-based material representations are then presented, including a discussion of cell-based meshes in combination with the level set method.

Finally, different numerical approaches to the modelling of material evolution in time are presented. These include approaches developed within this work to overcome fundamental limitations of previous methods. The modelling of physical processes and their application to material evolution is ultimately provided.

2.1 Modelling of Reactor Regions

Modern silicon wafers are circular with a diameter of 300 mm or $3 \cdot 10^{-1}$ m [22], while the smallest features generated in the manufacture are on the order of nanometres or 10^{-9} m [23]. Therefore, a full description of an entire wafer and all of its features would span more than 8 orders of magnitude, too much to be represented in a single simulation, as shown in Fig. 2.1. However, the simulation of a process can be divided into different size scales and performed separately, using the result of one simulation as an input for the next one. The largest part to be simulated is the space in which the wafer is being modified, the reactor. Therefore, the first step is to carry out a reactor-scale simulation to determine the properties of the gas-phase around the wafer, which includes the chemical composition as well as the electromagnetic properties which influence the fabrication processes.

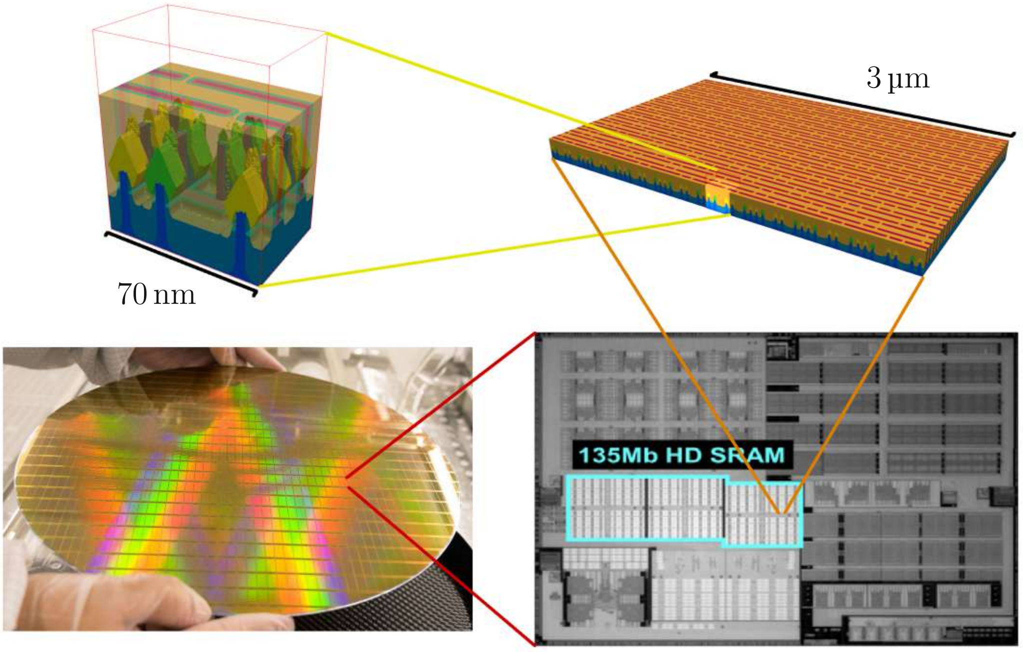


Figure 2.1: Length scales of semiconductor manufacturing. In the top left, a single SRAM cell of the 5 nm node with a total size of tens of nanometres is shown. It is part of the larger array of SRAM cells with a size of several micrometres shown in the top right. A large number of these arrays and additional control circuits form the SRAM region of the processor shown in the bottom right ©2021 IEEE [24]. One edge of the entire SRAM region is not larger than a few millimetres. The final chip is not larger than one centimetre, a small part of the wafer shown in the bottom left which is 300 millimetres in diameter ©2020 IEEE [25].

2.1.1 Reactor Scale Modelling

Every fabrication step of the semiconductor manufacturing process takes place inside a closed reaction chamber, such as a vacuum chamber or a plasma reactor. Gas inlets are used to control the flow of possibly several different reactive gases which ultimately control the composition of the atmosphere inside the reaction chamber. Using data from experiments [26] or simulations [27, 28], models for the distribution of different atoms or molecules in the gas phase above the wafer surface can be generated. Thus, engineers can optimise the reactor such that the molar concentration, as well as the angular and energetic distributions of gases impinging on the wafer surface are as uniform as possible across the wafer. This type of simulation does not model detailed features on the wafer surface, as they are too small to play a role in the description of the atmosphere above the wafer. However, the distributions generated by reactor scale models can be used as an input for subsequent feature scale simulations, which are then used to investigate how the specific properties of the reactor, such as gas inlet geometry and gas flow rates influence the feature scale topography and thus the devices fabricated on the wafer surface.

2.1.2 Feature Scale Modelling

The feature scale represents the scale at which electronic devices are built on the wafer. It is therefore closely tied to the size of the structure being manufactured and can range from a few nanometres for a single transistor [29] to several hundred micrometres for large microelectromechanical systems (MEMS) devices [30]. The different properties of the gas phase above the wafer, as discussed in Section 2.1.1, are used to model how different atoms or molecules impinge on the surface and subsequently react with it. Many different types of radicals can be involved in a single process, resulting in complex behaviour depending on the specific properties of the impinging species and the composition, as well as geometry, of the substrate. In plasma processes, energetic ions frequently hit the surface in a specific direction, heavily influencing chemical reactions and thus the effect of a process on the wafer. In order to model the impinging of atoms and molecules and the subsequent surface reactions properly, the materials in the feature scale must be represented robustly. Several different numerical methods for the representation of material interfaces will be discussed in the next section.

2.2 Numerical Material Representations

The choice of numerical methods used to represent the materials in a simulation has great implications on the modelling capabilities of a simulator. The material representation must be chosen depending on the requirements of the models to be carried out. Therefore, in order to choose the correct representation for a specific task, it is important to develop a deep understanding of the required capabilities, as well as the properties of each material representation.

As discussed in Section 1.2, the goal of this work is to allow for the description of entire electronic circuits in a computationally efficient manner applicable for design technology co-optimisation (DTCO). Therefore, fast simulation times and the ability to represent large structures for the developed simulator is critical. In the following, different modelling approaches and their relative advantages and disadvantages are presented.

2.2.1 Atomistic Modelling

The most physically rigorous way to represent a material is to consider the structure of the atoms and molecules forming the substance. Computationally, atomistic models can be simulated using molecular dynamics [31] or Monte Carlo (MC) methods [32]. Using these methods, each fabrication process is simulated considering every single atom or molecule impinging on the atoms of the substrate and modelling each chemical reaction and forces between these atoms. Even surface roughness, which describes the properties of a surface at the smallest scale, can thus be modelled. Therefore, this approach results in the most rigorous physical description possible, but as every single atom has to be considered, it is computationally very costly and it is unfeasible to model large structures, even when large-scale computational resources are available. Hence, the number of atoms which can be modelled using these approaches is limited [33], so these methods cannot be employed to simulate the manufacture of entire microelectronic devices. They are rather applied to understand specific regions of a device, such as a certain material - gas molecule interactions on the interface [34]. These models have also been used to extract certain process parameters which are then applied in a kinetic Monte Carlo or continuum model, which work at larger time scales [35].

2.2.2 Continuum Approach

Due to the computational limitations of atomistic modelling, a continuum approach is commonly employed to model semiconductor manufacturing processes. In this approach, the surface of a material is considered continuous with no abrupt changes or steps [36], as would be the case when considering individual atoms. Therefore, a modelled material is considered a single solid body in the region $\mathcal{M} \in \mathbb{R}^D$, where D is the number of spatial dimensions of the simulation domain. The interface of the material is thus described by the bounding surface $\mathcal{S}$ of $\mathcal{M}$. Any change to the interface of a material is modelled by moving the surface $\mathcal{S}$ and thus changing $\mathcal{M}$.

During semiconductor manufacturing, several materials are combined in a single structure, where each material has specific properties which are commonly assumed to be constant in the continuum model. Therefore, each material $\mathcal{M}_i$ describes a region of space with a specific set of physical properties. The interface between two separate materials is described by the two bounding surfaces, so the physical properties change abruptly across the material interface. Hence, smooth transitions in material composition cannot be modelled straight-forwardly using this approach. However, if there is no long range smooth transition of physical properties and the modelled structure sizes are greater than the lattice constant of the modelled materials, it is sufficient to model the bounding surfaces of all materials to achieve an appropriate description of all materials. Additionally, the continuum approach allows for the resolution of the modelled interfaces to be adjusted to the requirements of a specific simulation, compared to the fixed resolution of the atomistic approaches given by the size of atoms.

Due to the improved computational efficiency and greater flexibility of the continuum approach, it is the principal method employed in commercial and academic process TCAD tools [37] and was chosen for the modelling carried out in this work. Several numerical material representations using this approach are discussed in the following sections.

2.3 Continuum Material Representations

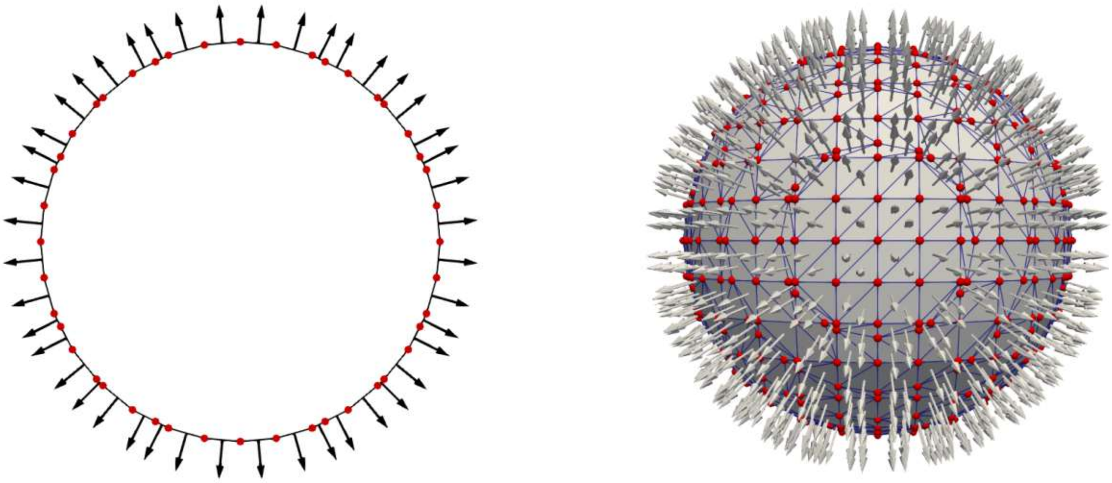
Using the continuum approach, the material $\mathcal{M}$ can simply be described by its bounding surface $\mathcal{S}$, as discussed in Section 2.2.2. If all physical properties of a material are homogeneous, describing the material interfaces is sufficient to capture all the relevant information without having to include a description of the entire volume.

2.3.1 Explicit Surfaces

The most straight-forward means to describe a surface is by defining points in space which are found on this surface and connecting them with surface elements. Such representations are referred to as explicit surfaces, because the location of the surface is given explicitly by the points, or nodes, on the surface. This approach is also referred to as segment-based, as the surface is segmented by the elements connecting the nodes [38]. In two-dimensional (2D) simulations, the surface elements are usually straight lines forming a closed loop surrounding $\mathcal{M}$. The surface normal of each surface element (i.e., straight line in 2D) is usually chosen to point toward the outside of the material, as shown in Fig. 2.2a. By convention, the normal points to the left side when going from the first node of a line segment $\vec{x}_0$ to the second $\vec{x}_1$, so the normal vector is given by:

$$\vec{n} = \begin{pmatrix} (\vec{x}_0 - \vec{x}_1) \cdot \hat{e}_0 \\ (\vec{x}_1 - \vec{x}_0) \cdot \hat{e}_1 \end{pmatrix} \quad , \tag{2.1}$$

where $\hat{e}_i$ is a unit vector in the i$^{\text{th}}$ Cartesian direction. Using this method, the order of the nodes of the line segmentation describes which side of the surfaces $\mathcal{S}$ corresponds to the inside of the material $\mathcal{M}$.



(a) Line segmentation of a circle in 2D.

(b) Triangulation of a sphere in 3D.

Figure 2.2: Explicit surface describing a circle in two and a sphere in three dimensions with nodes in red and surface normals indicated by arrows.

In three-dimensional (3D), the simplest surface elements to describe $\mathcal{S}$ are triangles which must form a closed volume describing $\mathcal{M}$, as shown in Fig. 2.2. As in the 2D description, the surface normals point outwards from $\mathcal{M}$ to indicate which side of the surface is inside of the material. The surface normal $\vec{n}$ is generated by the order of the nodes $\vec{x}_0, \vec{x}_1, \vec{x}_2$ forming the triangle:

$$\vec{n} = (\vec{x}_1 - \vec{x}_0) \times (\vec{x}_2 - \vec{x}_0) \quad . \tag{2.2}$$

Since the normal vectors of all elements are known, a point $\vec{x}$ is inside of $\mathcal{M}$ if it is inside of every surface element, thus requiring to check all segments to test whether $\vec{x}$ is inside the material.

Explicit representations are used in many applications, including graphics rendering [39] or computer-aided design (CAD) [40]. Several desirable properties make it a good choice for these applications, such as no limitation on resolution as the position of the surface is given by the location of the surface nodes, which can be placed anywhere in space. Another desirable property is the minimal memory requirement, as surface elements scale with the represented surface area and no extra memory is needed. Additionally, visualisation is straight-forward as all coordinates of the surface elements are known. If a process is dependent on the volume of a material, such as oxidation which is limited by the oxidisable substrate available inside the surface, the volume can be calculated straight-forwardly [41]. Therefore, explicit surfaces are ideal for representing complex shapes at varying resolutions and for extracting geometric properties, such as surface area or volume.

In addition, multiple materials can be represented straight-forwardly, by simply storing several different closed surfaces with a material identifier for each triangle, specifying which material the triangle describes.

2.3.2 Implicit Surfaces using the Level Set Method

Implicit surface representations do not store the location of the surface using nodes defining the exact coordinates, but rather another property is stored, which can be used to find the explicit location of the surface. These values are given by the implicit function $\phi(\vec{x})$, which returns a value for any point in space $\vec{x}$. In this work the convention is used that the exact location of the surface is given by the set of $\vec{x}$ for which $\phi(\vec{x}) = 0$. Hence, the location of the surface is given implicitly through the function $\phi(\vec{x})$.

A common choice for $\phi(\vec{x})$ is the signed distance function (SDF) $s(\vec{x})$, which returns the shortest distance to the surface $\mathcal{S}$, defined by surface points $\vec{x}_s$, from a given point $\vec{x}$:

$$s(\vec{x}) = \min ||\vec{x}_s - \vec{x}||_2 \quad , \tag{2.3}$$

where $||\cdot||_2$ denotes the Euclidean norm or ℓ_2 norm. The SDF is not solved for a variable, but used to find the set of all points $\vec{x}_c$, which let the SDF go to a specific scalar value c:

$$\{\vec{x}_c\} = \{\vec{x} | s(\vec{x}) = c\} \tag{2.4}$$

This set of points is an isocontour of the SDF, i.e. a contour of equal height, which is why this set of level points is referred to as the level set (LS) of s, $L_c(s)$ [42], giving this method its name. In order to represent a surface, c is usually chosen to be zero out of convenience [43, 44]. This convention is convenient because the zero LS $L_0(s)$ describes the exact location of the explicit surface $\mathcal{S}$:

$$\mathcal{S} = L_0(s) = \{\vec{x}|s(\vec{x}) = 0\} \tag{2.5}$$

Hence, the surface is given by all points in space $\vec{x}$ for which the SDF $s(\vec{x})$ is zero. Essentially, all points with zero distance to $\mathcal{S}$ must be on the surface itself, as expected. All points which are not part of $\mathcal{S}$ store the shortest distance to the surface with their sign denoting whether they are inside or outside of $\mathcal{M}$. In this work, the convention that points inside of $\mathcal{M}$ are negative is followed. Hence, if $\phi(\vec{x}) \leq 0$, the point $\vec{x}$ must be inside the material or on its surface and outside otherwise. This makes it very simple to check whether a point is part of $\mathcal{M}$ without any further considerations, which is not the case when using explicit surfaces.

Since it would be complex to represent all possible surfaces analytically, $\phi(\vec{x})$ is defined on a rectilinear grid with grid spacing Δg, storing its value at every grid point $\vec{g}$. For numerical stability and simplicity, $\phi(\vec{x})$ is then normalised to the grid spacing, such that the distance between two grid points is unity. In order to initialise $\phi(\vec{g})$ from an explicit surface, a signed distance transform is used to construct the SDF at all grid points from the points $\vec{x}_s$ of the explicit surface $\mathcal{S}$ bounding the volume $\mathcal{M}$:

$$\phi(\vec{x}, \mathcal{M}) = \frac{1}{\Delta g} \begin{cases} -s(\vec{g}), & \text{for } \vec{g} \in \mathcal{M} \\ 0, & \text{for } \vec{x} \in \mathcal{S} \\ +s(\vec{g}), & \text{for } \vec{g} \notin \mathcal{M} \end{cases} \tag{2.6}$$

For simplicity, the dependency of $\phi(\vec{x})$ on $\mathcal{M}$ is not written explicitly every time, but was added here for clarity. Thus, after initialisation, all grid points store the normalised signed distance to the original surface, which means that the LS values $\phi(\vec{x})$ are not equivalent to the SDF defined in Eq. (2.3) anymore due to the additional scaling. However, this scaling does not affect the signed distance property of $\phi(x)$ and can thus be applied without further consideration.

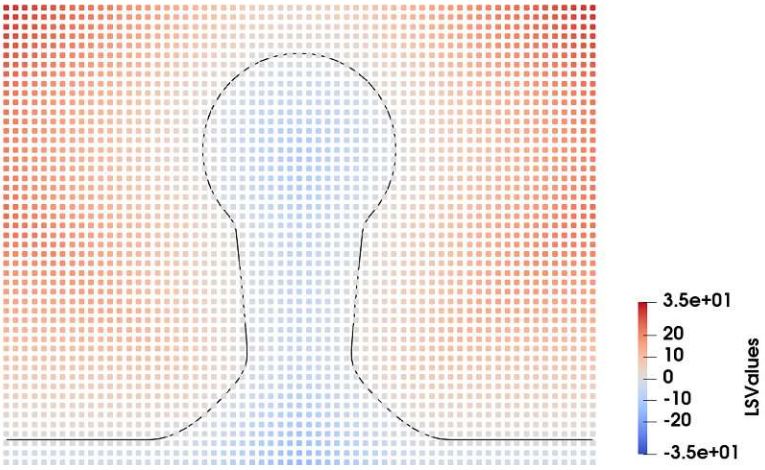


Figure 2.3: Level set method on a full grid, storing $\phi(\vec{x})$ for all grid points. The black line represents the location of the explicit surface.

2.3.2.1 The Narrow Band Method

In the LS method discussed above, all grid points in the simulation domain hold the signed distance to the surface $\mathcal{S}$. Therefore, the number of grid points used to store the location of $\mathcal{S}$ scale with volume as opposed to surface area, which would be the ideal case, achieved only using explicit surface representations [45]. In the narrow band method, the number of required grid points $\vec{g}$ is reduced, by considering that only grid points close to the surface contribute significantly to the surface description [46].

The contribution of a grid point to the surface description depends on its distance to the surface and thus its LS value $\phi(\vec{g})$. Therefore, it is useful to classify points into layers $\mathcal{L}_i$, based on their LS value:

$$\mathcal{L}_i = \begin{cases} \{\vec{x} \mid \quad i - \frac{1}{2} \leq \phi(\vec{x}) < i + \frac{1}{2} \quad \}, & \text{for } i < 0 \\ \{\vec{x} \mid \quad -\frac{1}{2} \leq \phi(\vec{x}) \leq \frac{1}{2} \quad \}, & \text{for } i = 0 \\ \{\vec{x} \mid \quad i - \frac{1}{2} < \phi(\vec{x}) \leq i + \frac{1}{2} \quad \}, & \text{for } i > 0 \end{cases} \tag{2.7}$$

Layer $\mathcal{L}_i$ then contains all points in space within one grid spacing around the set L_i.

In the LS method, all layers $\mathcal{L}_i$ for $i \in \mathbb{Z}$ are included in the set of grid points, meaning there is no maximum value for $\phi(\vec{g})$. In the narrow band method, the set of grid points is substantially smaller by only considering the layers $\mathcal{L}_i$ for $-\frac{k}{2} < i < \frac{k}{2}$ [47], where k is the narrow band width. Therefore, only grid points inside the narrow band, the region with thickness $k\Delta g$ centred around the zero LS L_0, are stored. These grid points are referred to as defined points and are shown in Fig. 2.4. The narrow band width is chosen depending on the needs of the specific application, where lower values result in better memory efficiency, but may also result in loss of accuracy depending on the application. There is no universally ideal narrow band width. Compared to the LS method, memory and computational requirements are thus reduced from $O(N^D)$ to $O(N^{D-1}k)$, where N is the number of grid points in each direction and D is the number of dimensions.

2.3.2.2 The Sparse Field Method

When the narrow band width k, introduced in Section 2.3.2.1, is reduced to 1, only the layer $\mathcal{L}_0$ is stored. The grid points within this layer are called active points $\vec{a}$, analogously to the defined points in the narrow band method, and satisfy:

$$|\phi(\vec{a})| \leq \frac{1}{2} \quad , \tag{2.8}$$

which is the same as Eq. (2.7) with $i = 0$. The value of $\frac{1}{2}$ means that every active grid point has exactly two neighbouring active grid points, as shown in Fig. 2.5. Since there cannot be fewer active grid points without creating a hole in the surface, the resulting LS is the smallest set of points being able to describe a surface robustly. This set of grid points is called sparse field [48] and leads to the optimal $O(N^{D-1})$ memory scaling behaviour, which is identical to that achieved by explicit methods.

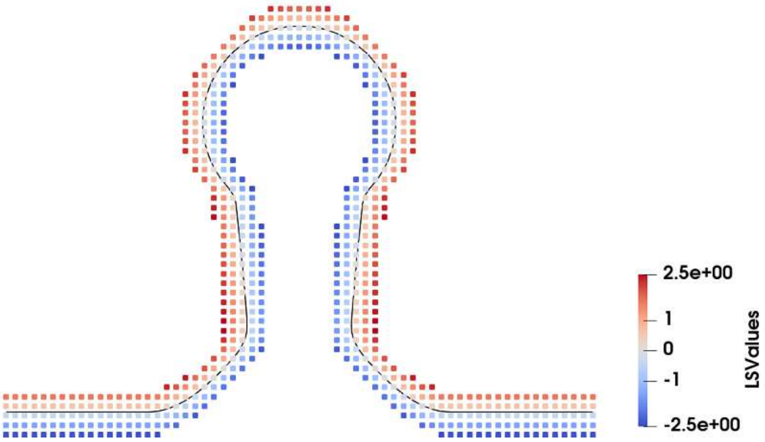


Figure 2.4: An explicit surface (black) described implicitly using the narrow band method, storing only points for which $|\phi(\vec{x})| \leq \frac{k}{2}$. Here, the LS for the commonly used value of $k = 5$ is shown.

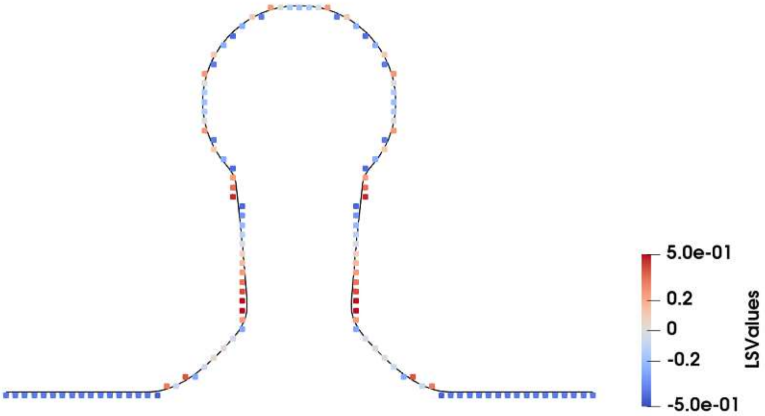


Figure 2.5: Using the sparse field method, the explicit surface (black) is only represented by the grid points in the layer $\mathcal{L}_0$, resulting in the smallest set of points possible to describe the surface.

However, this sparse set of points requires additional attention in order to form a valid and efficient surface description. By definition, the surface is described by L_0 which must be located between two grid points of opposite sign. As $\phi(\vec{x})$ is normalised to the grid spacing Δx, only one of the two oppositely signed points will be active, unless for the rare case of both being exactly 0.5. However, it is sometimes necessary to generate the neighbour points of active points, for example to compute geometric properties of the surface, as will be discussed in Section 2.3.2.3. Using Fig. 2.6a as a visual guide, the set of active points (red) is used as a starting point to generate the set of nearest neighbour points (blue). As every point should store the distance to different parts of the surface, calculating the required values is not straight-forward.

This problem is expressed in the Eikonal equation

$$|\nabla\phi(\vec{x})|F(\vec{x}) = 0 \quad , \tag{2.9}$$

which must be solved for every nearest neighbour grid point. The act of calculating neighbouring $\phi(\vec{x})$ values is usually referred to as re-distancing, reinitialisation, or normalisation when it is applied after moving full grid or narrow band level sets [49]. Currently available numerical methods to obtain the solution to this problem include the fast marching method [44, 50] and variations thereof [51, 52, 53], the fast sweeping method [54, 55, 56] and the fast iterative method [57]. Although some of these methods have optimal scaling properties of $O(N)$ for N neighbours to calculate, they may depend on certain assumptions about $\phi(\vec{x})$ and all of them still require considerable computational effort as the differential equation needs to be solved at every point.

The problem of generating neighbour points can be simplified greatly by choosing a different norm for the SDF describing $\phi(\vec{x})$ [58]. By changing the norm in Eq. (2.3), the SDF

$$m(\vec{x}) = \min ||\vec{x}_s - \vec{x}||_1 \tag{2.10}$$

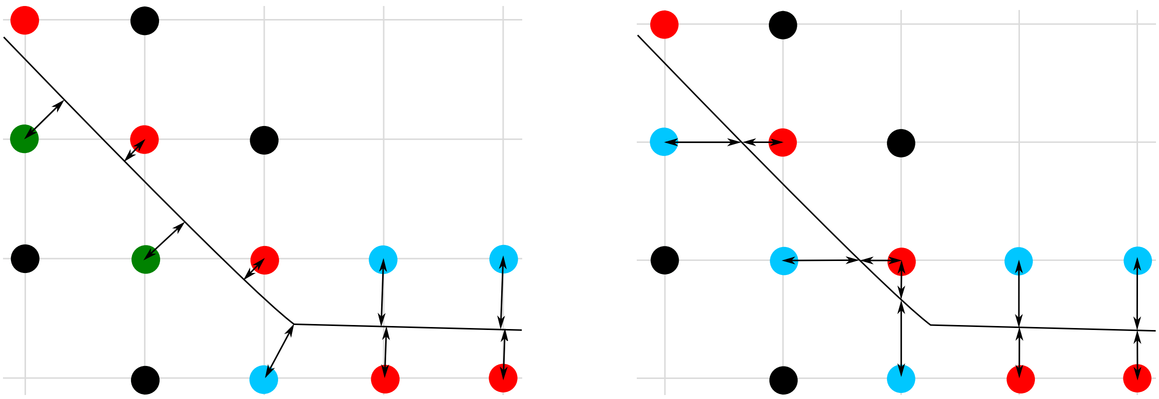
is obtained, where $||\cdot||_1$ represents the Manhattan or ℓ_1 norm. The numerical LS function is then defined as in Eq. (2.6) with $m(\vec{g})$ instead of $s(\vec{g})$. Every grid point now stores the shortest grid line distance to the interface and $\phi(\vec{g})$ can be generated by simply checking where $\mathcal{S}$ intersects the grid lines meeting at $\vec{g}$. As shown in Fig. 2.6, this norm also leads to a more sparse set of grid points, as the green points in Fig. 2.6a are part of $\mathcal{L}_0$ using the Euclidean norm, but have higher LS values using the Manhattan norm. Furthermore, neighbour point calculation is straight forward, as a neighbour point can be generated by simply adding unity to an active point. If a neighbouring point has more than one active point, the smallest value plus unity is simply taken as the final LS value, reducing the computational effort to a maximum of $2 \cdot D$ comparisons and 1 addition, a significant reduction in effort when compared to solving a differential equation several times for every neighbour using the ℓ_2 norm.

2.3.2.3 Geometric Properties of Implicit Surfaces

During simulations, geometric properties of the surface, such as surface normals or curvature are often required [59]. Hence, it is vital to obtain these properties efficiently. When applying the LS method, finding the surface normals and curvature is straight-forward by considering that the explicit surface is the isocontour of the LS at the value 0. Since $\phi(\vec{x})$ increases monotonically with the distance from the explicit surface, the gradient of $\phi(\vec{x})$ gives the normal direction from the surface. Thus, all geometric properties of the stored surface can be found by calculating derivatives of the LS.

Normal Vectors

One of the most frequently required geometric properties is the normal vector on the surface. Considering that $\phi(\vec{x})$ is a SDF with the location of the surface being



(a) Distances using the Euclidean norm.

(b) Distances using the Manhattan norm.

Figure 2.6: Different types of normalisation of the implicit function $\phi(\vec{x})$. The distance to the surface (black) a) using the Euclidean or ℓ_2 norm and b) using the Manhattan or ℓ_1 norm. Red points indicate active grid points, blue points indicate values smaller than 1. Green points indicate active grid points which are not necessary for the description of the surface, leading to an inefficient set of grid points.

defined as the zero LS L_0, the surface is simply the isocontour at $\phi(\vec{x}) = 0$. Since the gradient of a function is always perpendicular to the isocontours of that function, the unit normal vector of $\phi(\vec{x})$ at the point $\vec{x}$ is found using

$$\hat{n}(\vec{x}) = \frac{\nabla\phi(\vec{x})}{||\nabla\phi(\vec{x})||_2} \quad . \tag{2.11}$$

If $\vec{x}$ is a point on $\mathcal{S}$, Eq. (2.11) gives the surface normal at that point.

As the level set is stored numerically on the grid, the gradient is usually computed using finite differences. We therefore define the forward D_i^+ and backward D_i^- differences as

$$D_i^{\pm}(\phi(\vec{g})) = \pm\frac{\phi(\vec{g} \pm \Delta g\hat{e}_i) - \phi(\vec{x})}{\Delta g} \quad , \tag{2.12}$$

where $\hat{e}_i$ is the unit vector in direction i. The numerical derivative, i.e. the gradient $\nabla\phi(\vec{x})$, on the grid can be approximated by taking the central difference, the average of the forward and backward differences, at the grid point $\vec{g}$:

$$D_i(\phi(\vec{g})) = \frac{D_i^+(\phi(\vec{g})) + D_i^-(\phi(\vec{g}))}{2} = \frac{\phi(\vec{g} + \Delta g\hat{e}_i) - \phi(\vec{g} - \Delta g\hat{e}_i)}{2\Delta g} \quad . \tag{2.13}$$

Using the definition in Eq. (2.11), the ith component of the normal vector at the grid point $\vec{g}$ can thus be approximated using central differences:

$$n_i(\vec{g}) = \frac{1}{||\nabla\phi(\vec{g})||_2}\frac{\partial\phi(x_i)}{\partial x_i} \approx \frac{D_i(\phi(\vec{g}))}{\sqrt{\sum_{j=1}^{D} D_j(\phi(\vec{g}))^2}} \quad . \tag{2.14}$$

The normal vector at the grid point $\vec{g}$ is a good approximation of the normal at a surface point $\vec{x}_{cp}$ close to $\vec{g}$, shown as the green point in Fig. 2.7. Therefore, in order to obtain the surface normals, it is usually sufficient to generate them for every grid point in the layer $\mathcal{L}_0$, as shown in Fig. 2.8.

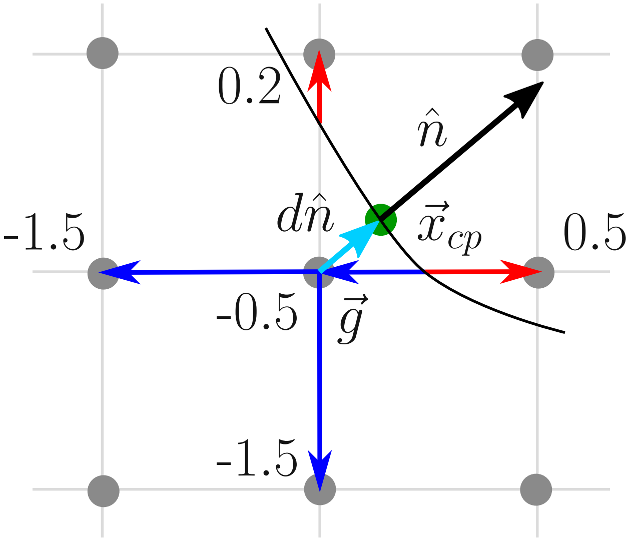


Figure 2.7: Normal vector calculation at the grid point $\vec{g}$ in the LS method using finite differences to approximate the normal at the closest surface point $\vec{x}_{cp}$ by shifting the grid point a distance d in the normal direction. This distance is calculated directly from the level set value of $\vec{g}$ using Eq. (2.15). Blue and red arrows indicate which distances were used to generate the LS values at the neighbouring grid points of $\vec{g}$.

Closest Point Approximation

If a point on the explicit surface is needed, the closest surface point $\vec{x}_{cp}$ to a grid point $\vec{g}$ can be generated directly from the normal and the level set value at $\vec{g}$ [60]. The point $\vec{x}_{cp}$ can be approximated straight-forwardly by shifting the grid point a distance d along the normal direction $\hat{n}(\vec{g})$, as indicated by the light blue arrow in Fig. 2.7. Due to the signed distance property of the LS, d is calculated by simply normalising $\phi(\vec{g})$ by its gradient:

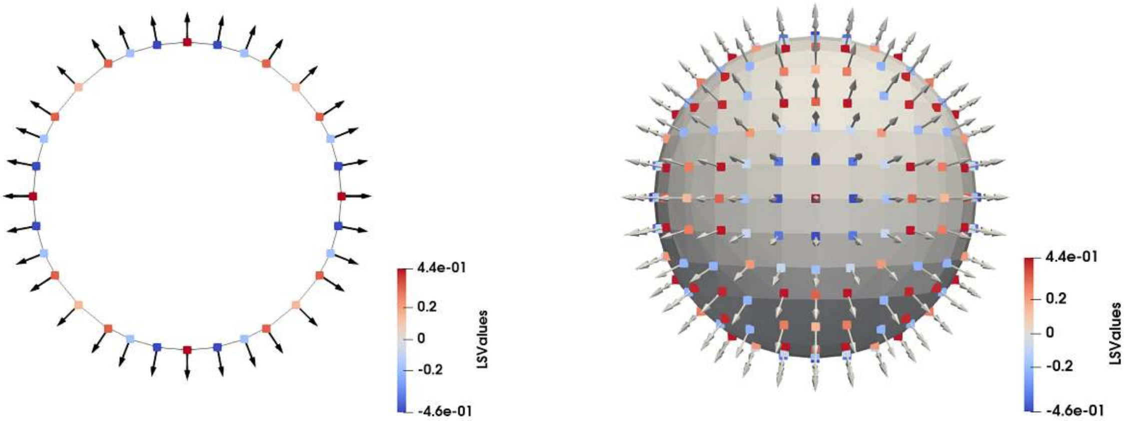
$$\vec{x}_{cp}(\vec{g}) = \vec{g} - d\hat{n} \approx \vec{g} - \frac{\phi(\vec{g})}{||\nabla\phi(\vec{g})||}\hat{n} \tag{2.15}$$

where $||\cdot||$ is the norm with which the implicit function was defined. For the SDF defined in Eq. (2.3) this is the ℓ_2 norm. If another norm was used, such as the ℓ_1 norm for the sparse field LS defined in Eq. (2.10), this norm must be used here to obtain the correct distance to the surface. By repeating this procedure for all grid points within the $\mathcal{L}_0$ layer, a set of normal vectors spaced roughly by Δg is generated, which is shown for 2D and 3D in Fig. 2.8 analogously to the explicit surfaces in Fig. 2.2.

Curvature

The curvature of a surface can be used for the detection of features, irregularities or for the smoothing of a surface as will be shown in Section 4.2.5.3. The mean curvature is given by the second derivative of the implicit function and is expressed as

$$\kappa(\vec{x}) = \nabla \cdot \frac{\nabla\phi(\vec{x})}{||\nabla\phi(\vec{x})||_2} = \nabla\vec{n}(\vec{x}) \quad . \tag{2.16}$$



(a) Shifted active grid points of a circle in 2D.

(b) Shifted active grid points of a sphere in 3D.

Figure 2.8: The closest surface points $\vec{x}_{cp}$ generated from all grid points in the $\mathcal{L}_0$ layer shown in the colour of the LS value of the original grid point. The normal vector approximated for each closest surface point is shown as an arrow starting at $\vec{x}_{cp}$.

Therefore, the simplest approximation to the mean curvature is the central difference of normals around the current grid point:

$$\kappa(\vec{g}) \approx \sum_{i}^{D} D_i(\vec{n}(\vec{g})) = \sum_{i}^{D} \frac{n_i(\vec{x} + \Delta g \hat{e}_i) - n_i(\vec{x} - \Delta g \hat{e})}{2\Delta g} \tag{2.17}$$

Although the above approach gives reasonable results and can be implemented straightforwardly, a more accurate and robust approximation to the mean curvature in 2D and 3D is given by [44]

$$\kappa(\vec{g}) \approx \frac{\sum_{i \neq j} \left(D_i^2 D_{jj} - D_i D_j D_{ij}\right)}{\left(\sum_{i=1}^{D} D_i^2\right)^{\frac{3}{2}}} \quad , \tag{2.18}$$

where the dependence of the derivatives on $\phi(\vec{g})$ is not written explicitly and the second order derivatives are given by

$$D_{ij} = D_i(D_j(\phi(\vec{g}))) \quad . \tag{2.19}$$

In 3D, the mean curvature is the average of the two principal curvatures. Their product, the Gaussian curvature, can also be calculated directly in the level set [61]:

$$\kappa_G \approx \frac{\sum_{i \neq j \neq k} \left[D_i^2 \left(D_{jj} D_{kk} - D_{jk}\right) + 2 D_i D_j (D_{ik} D_{kj} - D_{ij} D_{kk})\right]}{\left(\sum_{i=1}^{3} D_i^2\right)^2} \tag{2.20}$$

2.3.2.4 Boolean Operations

Boolean operations are frequently used to combine two geometric objects in order to generate a new one. In process technology computer aided design (TCAD) this is useful for generating initial geometries or to add masks on top of materials. Boolean operations are defined using the subspaces or materials $\mathcal{M}_A$ and $\mathcal{M}_B$ which are combined to result in $\mathcal{M}_C$. There are three basic Boolean operations which can be used to construct all possible combinations of materials. Using inside or outside of the material as a binary state, these are equivalent to common logic operations:

$$\text{Union}: \quad \mathcal{M}_C = \mathcal{M}_A \cup \mathcal{M}_B \qquad \text{OR}: \quad \mathcal{M}_C = \mathcal{M}_A \vee \mathcal{M}_B \tag{2.21}$$

$$\text{Instersection}: \quad \mathcal{M}_C = \mathcal{M}_A \cap \mathcal{M}_B \qquad \text{AND}: \quad \mathcal{M}_C = \mathcal{M}_A \wedge \mathcal{M}_B \tag{2.22}$$

$$\text{Complement}: \quad \mathcal{M}_C = \mathbb{R}^D \setminus \mathcal{M}_A \qquad \text{NOT}: \quad \mathcal{M}_C = \neg\mathcal{M}_A \tag{2.23}$$

Using surface representations, Boolean operations are used to combine two surfaces and compute the resulting interface. In order to combine explicit surfaces in such a way, the elements making up the surface must be cut and recomputed, requiring expensive intersection and reconstruction algorithms [62]. However, when representing these surfaces as level sets, Boolean operations become simple algebraic expressions which can be can be solved efficiently [63, 64]. In certain applications it was shown to be more efficient to convert to an implicit representation, conduct Boolean operations and then convert back to explicit surfaces instead of performing Boolean operations directly on the explicit surface[65, 66]. The operations on the level sets $\phi(\vec{x})_A$, $\phi(\vec{x})_B$ and $\phi(\vec{x})_C$ corresponding to the Boolean operations in Equations (2.21) to (2.23), respectively, are:

$$\text{Union}: \qquad \phi_C(\vec{x}) = \min(\phi_A(\vec{x}), \phi(_B\vec{x})) \tag{2.24}$$

$$\text{Instersection}: \qquad \phi_C(\vec{x}) = \max(\phi_A(\vec{x}), \phi_B(\vec{x})) \tag{2.25}$$

$$\text{Complement}: \qquad \phi_C(\vec{x}) = -\phi_B(\vec{x}) \tag{2.26}$$

The exact operations to carry out on the level set depends on the convention of the sign denoting inside or outside. Here, the convention of negative values being inside the material are used. Using these expressions, the resulting value $\phi_C(\vec{x})$ can be computed by simply considering the level set value at the same location $\vec{x}$ in the level sets $\phi_A(\vec{x})$ and $\phi_B(\vec{x})$, as shown in Fig. 2.9. Therefore, a single pass over the grid is sufficient to compute the resulting level set surface, meaning this operation is highly efficient. The efficiency of this approach is especially clear for the complement, where the level set values only have to be inverted.

2.3.2.5 Multiple Materials

The simulation of semiconductor fabrication processes requires different types of materials to be represented accurately. The process flow in Section 1.1 requires more than 10 different materials to be represented inside one simulation domain. As discussed in Section 2.3.1, several different materials can be stored straight-forwardly using explicit meshes. However, using the LS method, only one single surface can

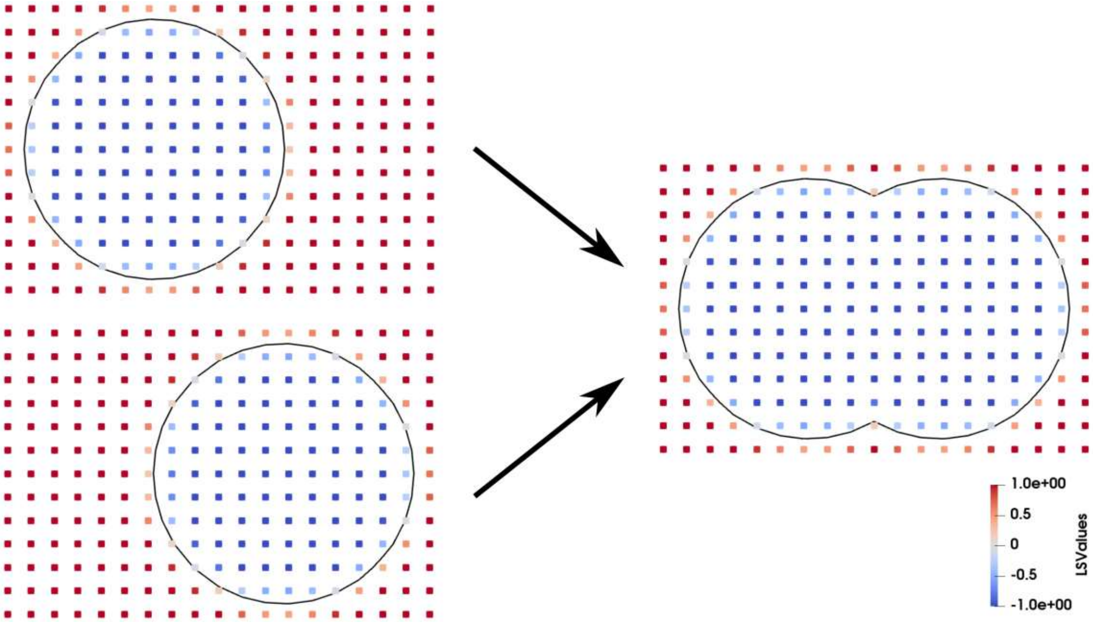
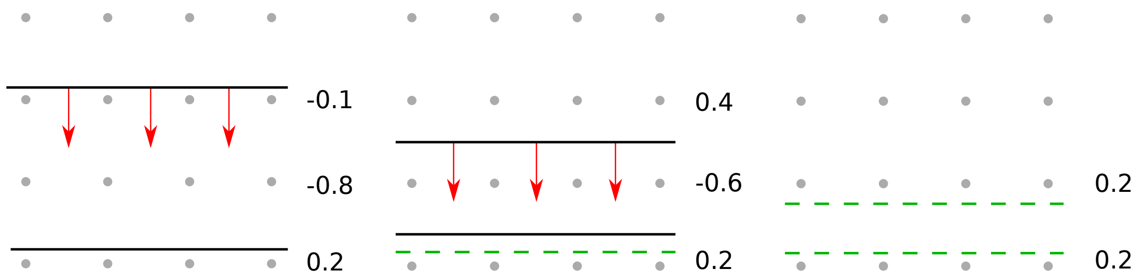


Figure 2.9: Boolean operation on the level set to generate the union of two surfaces represented implicitly. The new surface is created by taking the lowest LS value at every grid point. This automatically results in a valid LS describing the union of both surfaces.

be represented, as the LS only stores the signed distance to the boundary. Hence it is necessary to store one level set for each material. Storing the level set values on the same grid thus results in a sufficient description of multiple materials within a single simulation domain. However, when thin layers must be represented, the limited resolution of the LS grid leads to numerical difficulties, as shown in Fig. 2.10.

Materials spanning at least two grid points can be represented accurately. However, if there is only one grid point describing both surfaces, the level set values lose their normalisation and at least one of the two boundaries cannot be represented accurately anymore. Fig. 2.10b highlights this symmetric shrinking of the explicit surface if the material is too thin to be represented robustly in a LS. Since the exact location of the surface depends on two neighbouring LS values with opposite sign and the centre grid point can only store one value, both explicit surfaces must be generated using this value, leading to the observed discrepancy. This effect is even stronger if there is no grid point inside the surface since there are no oppositely signed neighbour grid points anymore which could be used to reconstruct the explicit surface. Thus, a thin material layer would disappear in the LS representation, although it may still be almost a grid spacing wide. For physical process simulations, this can lead to large problems, as several fabrication steps, such as plasma etching [67], depend on very thin passivation layers protecting the surface. Furthermore, the deposition of few-atom thick layers [68] is an important fabrication step for various modern devices and therefore must be represented appropriately.



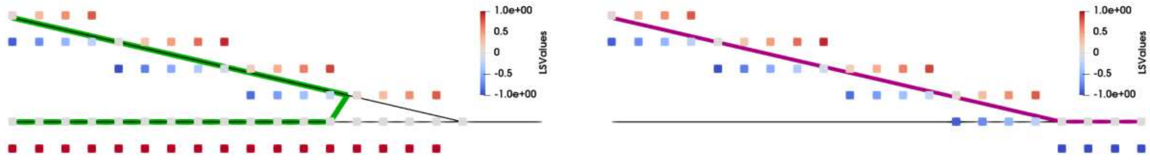
(a) Initial layer, only two grid spacings wide, whose top surface is moved downwards (e.g., etching).

(b) As the layer is thinned to only one grid spacing, the level set values are not normalised anymore, resulting in symmetric shrinking.

(c) Once the last row of grid points is outside of the layer, it ceases to exist, although it should still be almost one grid spacing wide.

Figure 2.10: Different position of a plane surface represented in a level set. The difference to the next position is indicated by red arrows. Grey points indicate the grid, black lines the surface and dashed green lines the correct position of the surface. The level set values of each row are shown to its right.

In order to solve this problem, it helps to consider that such thin layers are only formed on thicker substrates and usually do not occur as thin pillars or spikes on their own, but rather as envelopes covering an underlying substrate. Thereby, the bottom surface does not need to be represented at all since the combined knowledge of the top surface and the surface where another material starts are sufficient to describe a thin layer. This essentially describes this material wrapping or enveloping the material below it. The level set which represents the thin layer then describes not only the thin layer material, but also the material on top of which it was grown. The difference is shown in Fig. 2.11 for a thin passivation layer on the surface of a substrate, as commonly encountered in plasma etching processes.



(a) Thin material without layer wrapping.

(b) Layer wrapping used to accurately represent a thin material.

Figure 2.11: a) Thin layers (green) of a material cannot be represented accurately by the level set method. b) If the thin layer is defined on top of another material, as is the case in semiconductor fabrication processes, layer wrapping can be employed to achieve an accurate surface representation (purple) matching the explicit surfaces. The explicit material interfaces are indicated by thin black lines.

The so-called layer wrapping approach thus leads to the expected results and can be used to appropriately describe thin layers. In order to achieve a robust material representation, including thin layers, each LS $\phi_i(\vec{x})$ must be defined to encompass the union $\mathcal{M}_i$ of all underlying materials $\mathcal{M}_j$:

$$\phi_i(\vec{x}) = \phi_i(\vec{x}, \mathcal{M}_i) \quad \mathcal{M}_i = \bigcup_{j=1}^{i} \mathcal{M}_j \tag{2.27}$$

Unwanted effects, such as spaces between materials, symmetric shrinking and disappearance of thin layers are thereby solved robustly, as shown in Fig. 2.12

Great care must be taken when choosing the ordering of materials in order to decide which LS should include and wrap which other surfaces. The most natural and straight-forward ordering is to number the materials by the time of creation during the simulation. The initial substrate is then labelled as material 1, the next one as material 2, and so on. Since a new material can only grow on top of already existing materials and the new layer will wrap those, unwanted side effects such as spaces between materials will be avoided using this strategy.

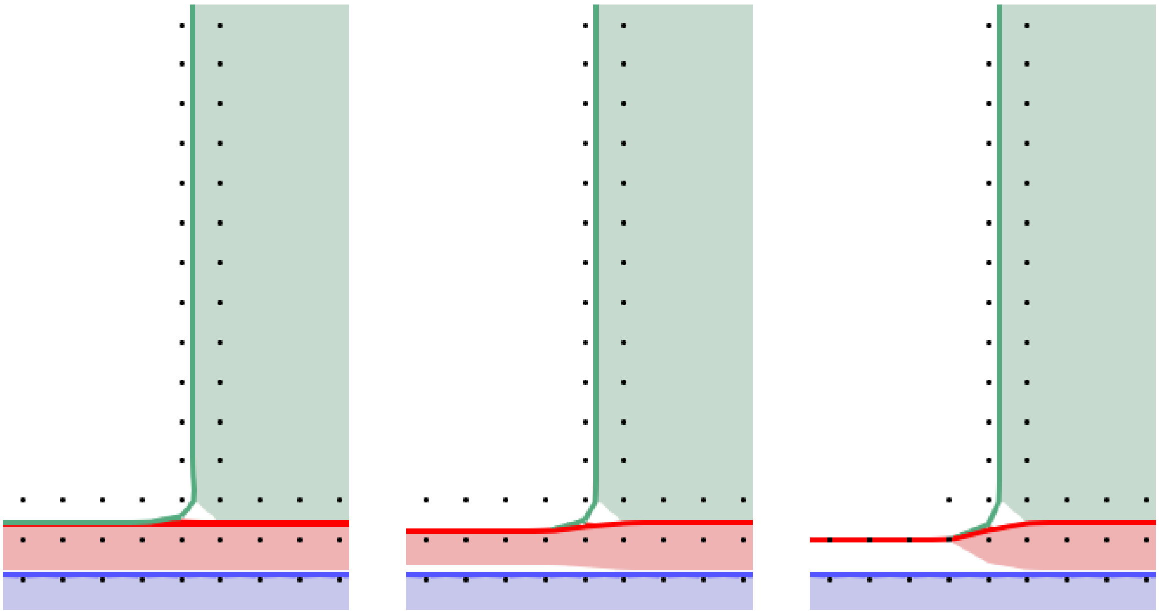
However, if several materials are introduced at the same time, or complex shapes are simulated with intertwined materials, special considerations may be necessary. Nonetheless, the layer ordering described above leads to robust and predictable behaviour and has been found to be the most straight-forward and efficient means to include thin material films with sub-grid accuracy.

2.3.3 Volume Representations

Modelling only the surface of a material is sufficient when the physical properties are homogeneous throughout the entire material, as discussed in Section 2.3. However, microelectronic devices are very much dependent on certain properties varying throughout a single material, such as doping concentration [69] or strain [70]. In this case, the entire volume must be represented numerically.

2.3.3.1 Tetrahedral Meshes

Volumes can be represented numerically using explicit representations, such as a tetrahedral mesh [71]. A volume mesh can also be made up of many other geometric shapes such as cubes or hexahedrons [72]. However, tetrahedrons are usually used as a they can be built from a triangle by simply adding a single node, leading to minimal memory consumption. Such explicit volume meshes are used extensively in the fields of visualisation [73], mechanical engineering [74] and in microelectronic device simulations [75] mentioned in Chapter 1. Much like explicit surface representations, this approach offers optimum memory efficiency and simple generation of geometric properties. However, certain applications require specific conditions to be met regarding the exact structure of the mesh, such as fulfilling the Delaunay criterion [76]. Furthermore, all parts of the volume must be stored explicitly, so large areas with the same physical properties require more memory than necessary. Additionally, since all tetrahedrons are independent, neighbour information has to be stored separately, leading to additional overhead [77].



(a) Initial geometry already showing voids if layers are not wrapped.

(b) The thin layer shrinks symmetrically when etched from the top creating a void above the substrate.

(c) Thin layer disappearing completely, leading to a jump of one grid spacing in the process.

Figure 2.12: Level set layers representing different materials during an etch process of a thin layer (red) above a substrate (blue) both protected by a mask (green). The filled areas show the material if no layer wrapping is applied and the coloured lines show the corresponding LS when employing the layer wrapping approach. Voids in the corners of geometries (a), symmetric shrinking (b) and loss of sub grid materials (c) are all handled appropriately using this approach.

2.3.3.2 Cell-Based Meshes

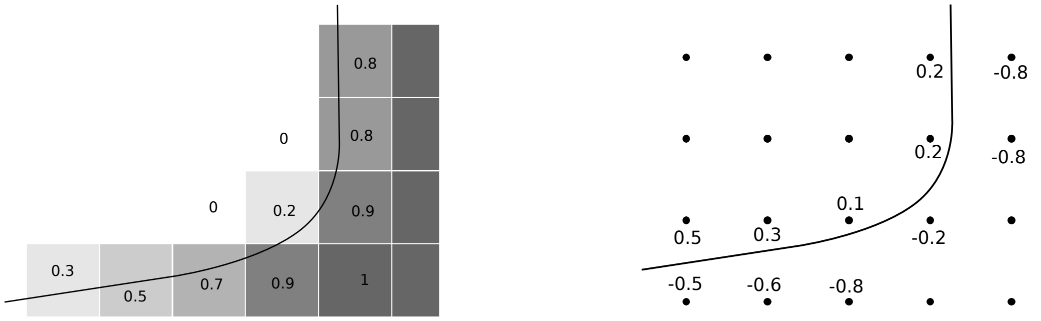
Another technique for storing volume information numerically is achieved when the simulation space is split into a rectilinear grid, similar to the grid used in the LS method. Therefore, the simulation space is now divided into squares in 2D and cubes in 3D. The simulation space in this method is thus split into cells of equal size, which is why it is usually referred to as a cell-based volume [78]. Each cell is then assigned a number denoting the material the cell represents and a filling fraction ff, denoting how much of the cell is filled with the respective material. The filling fraction is usually bound to the range $[0, 1]$ so that the sum of all filling fractions in a cell always equals unity.

Explicit Volumes

In the simplest case, the filling fraction is binary, meaning the cell is either entirely occupied by the material or empty. This way, there is no need to store a filling fraction. Storing only a material identifier specifying which material the cell belongs to is sufficient. Therefore, the material is described by a collection of squares or cubes, which form an explicit volume description, as mentioned above. This method is referred to as voxel-based volume representation [79] and leads to stepped interfaces between materials since the resolution is limited by the size of the voxels. Therefore, high resolutions are required in order to achieve satisfactory final geometries with precise contours. If the size of individual voxels is close to the size of physical atoms, this approach can be considered for the application in atomistic modelling which enables the simulation of surface reactions in great detail, as well as atomistic effects such as diffusion or ion implantation [80].

Implicit Volumes

If the filling fraction is a real number, this method is very similar to the LS method, since it represents the material boundary implicitly on a grid. Therefore, this type of cell-based representation is called cell set (CS) and a simple surface is shown in Fig. 2.13. Due to the strong similarities in the fundamental concepts, cell-based methods share the same shortcomings as level sets, such as lacking memory efficiency, although this can be solved employing techniques analogous to the sparse field method.



(a) Cell-based representation of a curved boundary of a material. The numbers in the cells represent the filling fraction stored in the respective cell.

(b) Sparse field LS for the same curved boundary from (a), storing the signed distances to the interface.

Figure 2.13: Comparison of a cell-based material and a sparse field LS. While the former is intrinsically associated with volume, level set representations describe an interface or boundary. However, both share common properties as they both represent a material boundary implicitly.

Algorithms associated with describing changing materials using cell-based methods are not as efficient as those developed for the LS method [81]. Although the filling fractions can be used for the calculation of explicit material interfaces [82], they often require complex algorithms to generate the explicit surface from filling fractions alone [83]. Since the exact position of a material inside a cell is not known, the explicit surface needs to be approximated using several neighbouring cells. Furthermore, the precise calculation of the surface normal requires a large number of neighbouring cells as there is no signed distance property as in the LS which could be used. It is therefore not possible to robustly determine the exact location of the surface, especially if there are several different materials present inside one cell. Therefore, the level set method is more appropriate for representing material boundaries.

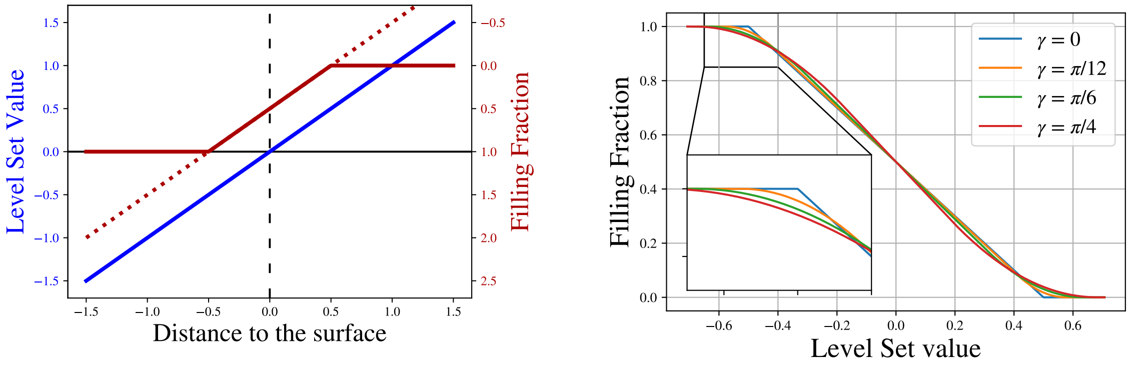
Due to their similarities, both implicit cells and LSs can be defined on the same grid, where the level set is used to store the exact location of the surface, while the cell-based volume representation is used to store volumetric data, such as doping concentration, strain, or the number of impurities. When the two representations are combined, it is sometimes necessary to convert between them. For example, the surface is advanced in the LS, resulting in the need to also update the filling fraction values. Considering the geometric nature of both functions, namely the distance from the surface for the LS and the volume of the cell inside the boundary for the CS, their underlying implicit functions are closely related. For the conversion to the cell-based representation, the explicit surface can be found using the level set values and the volume inside the cell at the interface is calculated as shown in Appendix A. In one-dimensional (1D) space, the filling fraction $\mathit{ff}(\vec{x})$ and the LS are related by

$$\mathit{ff}(\vec{x}) = \begin{cases} 0, & \text{for} \quad 0.5 < \phi(\vec{x}) \\ 0.5 - \phi(\vec{x}), & \text{for} \quad -0.5 \leq \phi(\vec{x}) \leq 0.5 \\ 1, & \text{for} \quad \phi(\vec{x}) < -0.5 \end{cases} \tag{2.28}$$

This simple relationship is shown in Fig. 2.14a. Note that the axis for the filling fraction is inverted and that, in the interval $[-0.5, 0.5]$, the LS value and the filling fraction only differ by a constant offset of 0.5. Outside of this interval, the two functions differ since the filling fraction cannot be less than 0 or more than 1.

In higher dimensions, the filling fraction is also dependent on the normal vector of the surface, as shown in Fig. 2.14b. The exact relationship is described in Appendix A, where the simple relationship in Eq. (2.28) is only valid if the normal vector is parallel to a grid axis. However, for non-zero angles to the grid γ, the function relating the filling fraction and the level set becomes smoother, as highlighted by the inset in Fig. 2.14b. The maximum difference occurs at an angle of $\gamma = \pi/4$, where the filling fraction differs by 0.04 in 2D. The 1D solution in Eq. (2.28) is therefore a simple approximation for any 2D case, but may lead to an error of up to 4%. Although this approximation might be sufficient for some applications, the error is too large to tolerate for sophisticated physical simulations. Since the surface normal of a level set can be generated straight-forwardly, as described in Section 2.3.2.3, there is no need to use this simple approximation and the accurate CS values can be calculated instead.

The inverse problem, converting a CS to a LS, is not always possible unambiguously. In 1D, a sparse field LS can be generated by simply inverting Eq. (2.28). However, the relationship between the filling fraction and the LS value is more complex in higher dimensions and depends on the surface normal, as shown in Fig. 2.14b. Since the filling fraction does not store information about the location of the surface, it is not possible to calculate the surface normal from the filling fractions alone. This can be visualised by considering that $\mathit{ff}(\vec{x})$ is not a straight line in Fig. 2.14b, meaning that it does not represent a signed distance. Therefore, a robust conversion from a CS to a LS is not possible in dimensions higher than 1.



(a) 1D level set and filling fraction.

(b) Level set versus filling fraction in 2D.

Figure 2.14: Relationship between the level set values and filling fractions for a) 1D and b) 2D representations. In 1D, the filling fraction (red) follows a straight line (dotted red) in the allowed interval $[0, 1]$, which is given by Eq. (2.28). In dimensions higher than 1, the relationship is dependent on the normal vector, where γ is the angle between the normal vector and a grid axis. In the interval $\gamma = [0, \pi/2]$ the function is symmetric around $\gamma = \pi/4$ and then repeats periodically.

2.4 Modelling Material Evolution

In order to describe a semiconductor fabrication process with sufficient accuracy, the most important feature of a simulator is the accurate description of the evolution of material surfaces and interfaces over time. For certain types of processes it might be sufficient to simply describe the final result of the process, while for others the precise physical behaviour of materials over time must be included in order to reach a satisfactory final structure. It is therefore necessary to describe simulated materials and their evolution over time with high accuracies in order to achieve satisfactory results. As with the numerical description of materials, the capabilities and limitations of the simulation strongly depend on the numerical method chosen to solve the evolution of material boundaries over time. Several different methods will be discussed here to solve the general problem of moving a numerical surface representing material interfaces.

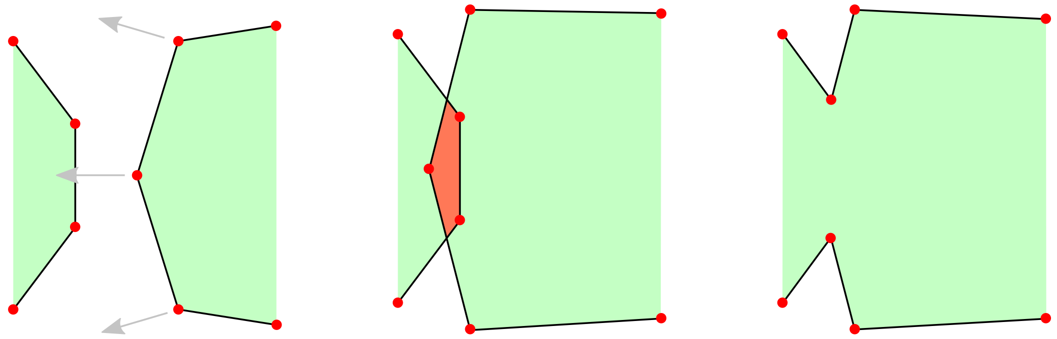
The movement of a surface in time is usually described by a vector velocity field $\vec{V}(\vec{x}_s)$ describing where each point on the surface $\vec{x}_s$ should move. This velocity field is obtained by physical process models or empirical geometric models which will be discussed in Chapter 5. The general problem of generating the final material $\mathcal{M}(t = t_{\text{final}})$ must be solved:

$$\mathcal{M}(t = t_{\text{final}}) \text{ given } \mathcal{M}(t = t_{\text{initial}}), \vec{V}(\vec{x}_s, t) \text{ for } t \in [t_{\text{initial}}, t_{\text{final}}] \tag{2.29}$$

Depending on the specific method employed to generate $\mathcal{M}(t = t_{\text{final}})$, $\vec{V}(\vec{x}_s, t)$ might only be required at specific times or only at t_{initial} and not during the entire time range $[t_{\text{initial}}, t_{\text{final}}]$. The aim of the numerical methods presented in the following is to apply this velocity field to the surface as accurately as possible with the highest achievable computational efficiency.

2.4.1 Evolution of Explicit Meshes

Explicit surfaces are evolved over time by shifting the nodes which define the surface in the direction given by the velocity vector $\vec{V}(\vec{x}_s, t)$ [84]. The surface elements, usually triangles, still connect the same nodes as before and the final surface is thus complete. However, this can lead to non-physical intersections of surface elements [43]. Such a self-intersection is depicted in Fig. 2.15, where two parts of the same material move towards each other and create an overlap between the surface elements.



(a) Valid explicit surface describing initial material.

(b) Broken surface due to a self-intersection.

(c) Repaired final surface through re-meshing.

Figure 2.15: When the nodes (red) of an explicit surface (black lines) describing a material (green) are moved, the result may include regions of non-physical self-intersections (red). These regions must be repaired using subsequent re-meshing steps requiring substantial computational effort.

In order to overcome such non-physical shapes, self-intersections must be found, which is computationally expensive, since every surface element must be checked. Even after identifying affected surface elements, complex re-meshing algorithms must be carried out in order to resolve the conflicting surface elements and to generate the correct physical structure depicted in Fig. 2.15c.

In addition to self-intersections, moving explicit surface representations results in a changing accuracy of the surface description. As an explicit interface is only defined by the nodes on the surface, moving the surface may result in closely spaced nodes in certain sections and widely spaced surface nodes in other sections of the surface, as shown in Fig. 2.16. The resolution may be high in one region, unnecessarily storing surface nodes which are very close together and thus leading to a memory misuse. Another region of the surface may be described by only a few points, leading to a low resolution and an inaccurate description of the interface. In order to mitigate the effects of the changing density of nodes, the surface must be re-meshed regularly to insert or remove nodes where necessary in order to achieve certain mesh quality criteria, such as equal area triangles or equal edge lengths [85]. Although algorithms exist for this type of problem, they require a substantial amount of computational effort and are thus undesirable for complex 3D simulations.

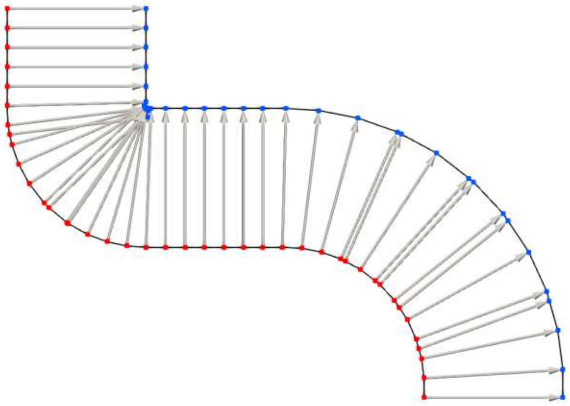


Figure 2.16: Nodes (red), defining the initial material interface (black), are shifted by the surface normals (arrows) to move the surface outwards isotropically as would occur during conformal deposition. The nodes of the final interface (blue) are spaced differently, leading to many overlapping points in areas of concave curvature and widely spaced points in area of convex curvature.

2.4.2 Iterative Level Set Advection

Moving an implicit surface is referred to as advection and is governed by a differential equation called the LS equation

$$\frac{\partial\phi(\vec{x},t)}{\partial t} + V(\vec{x},t)||\nabla\phi(\vec{x},t)||_2 = 0 \quad , \tag{2.30}$$

where $V(\vec{x},t)$ is a scalar velocity field denoting the speed of the propagation in the normal direction to the surface. Given $\vec{V}(\vec{x},t)$ and surface normal vectors $\hat{n}(\vec{x})$ calculated with Eq. (2.11), the scalar velocity field is given by

$$V(\vec{x},t) = \vec{V}(\vec{x},t)\cdot\hat{n}(\vec{x}) \quad . \tag{2.31}$$

Many process models give an expression for the surface speed already in the normal direction which can then be applied directly.

In order to find $\mathcal{M}_{\text{final}}$, Eq. (2.30) must be solved to find $\phi(\vec{x},t_{\text{final}})$. Since the LS equation is a Hamilton-Jacobi type differential equation, numerous numerical schemes are available to solve it, which differ in complexity, performance and assumptions about the velocity function. In order to apply such numerical schemes, Eq. (2.30) is rewritten as

$$\frac{\partial\phi(\vec{x},t)}{\partial t} + \hat{H}(\phi(\vec{x},t),V(\vec{x},t)) = 0 \quad , \tag{2.32}$$

where the Hamiltonian is simply given by $\hat{H}(\phi(\vec{x},t),V(\vec{x},t)) = V(\vec{x},t)||\nabla\phi(\vec{x},t)||_2$.

In order to solve Eq. (2.32), the first step is to discretise time into several time steps of size Δt. The solution is then computed iteratively for each time step Δt until t_{final} is reached. One of the simplest numerical schemes to solve a differential equation is the first-order Euler method [86] which requires the Hamiltonian to be evaluated only once per time step. Therefore, the evolution of the LS during one time step is given by

$$\phi(\vec{x}, t + \Delta t) = \phi(\vec{x}, t) - \Delta t \; \hat{H}(\phi(\vec{x}, t), V(\vec{x}, t)) \quad . \tag{2.33}$$

Therefore, the entire change of the surface during one time step is encapsulated in the second term. The new LS value for each point $\vec{x}$ is calculated by simply subtracting the Hamiltonian from the initial LS value. More complex schemes for the solution in time exist and lead to better results through additional evaluations of the Hamiltonian [87] or higher order approximations [88, 89]. Such complex schemes require more computational effort and thus increased simulation time. However, regardless of the chosen numerical scheme, certain conditions must be met in order to guarantee numeric stability. These will be discussed further in the following sections.

2.4.2.1 Surface Velocity Field

For a full grid level set, the velocity field $V(\vec{x}, t)$ must be defined in the entire domain in order to solve Eq. (2.30). Since space is discretised, this means that there must be a value $V(\vec{g}, t)$ for every grid point $\vec{g}$ in the domain. However, during process simulations, only material interfaces are modelled and thus velocities can only be calculated at the interface itself. Therefore, the surface velocity must be extrapolated to all other grid points.

The speed at which the implicit function should move is generated by considering the closest surface point and using its velocity value [90]. In order to achieve this efficiently, the values for the velocity are propagated from the zero level set outwards, which is referred to as velocity extension [49]. This results in the same numerical problem, as for generating neighbouring level set values discussed in Section 2.3.2.2 [91]. The previously mentioned methods can be employed analogously to the extension of the surface velocities to the domain. In the case of sparse field level sets, this step can be ignored since only the $\mathcal{L}_0$ layer is used for advection. Only the points closest to the surface are used to describe the implicit function and the velocity field must be defined directly for these grid points, hence there is no need for a velocity extension.

2.4.2.2 CFL Condition

Numerical time integration schemes are used to propagate information from the surface. Due to the time and spatial discretisation of the problem, there is a maximum distance information should propagate in one time step, which is given by the Courant-Friedrichs-Lewy (CFL) condition [92]. For the advection of a LS, this condition limits how much a single LS value is allowed to be changed in a single time step, before all other LS values must be updated in order to guarantee numerical stability. Hence, it sets a maximum distance the surface should be moved for one solution of Eq. (2.30), i.e., one time step. This maximum distance, according to the CFL condition, is given by:

$$\max |\hat{H}(\phi(\vec{g}, t))\Delta t| = \max |\phi(\vec{g}, t + \Delta t) - \phi(\vec{g}, t)| \leq 1 \quad , \tag{2.34}$$

where the difference in LS values is simply the product of the Hamiltonian and the time step, as defined in Eq. (2.33). Since the difference in $\phi(\vec{g}, t)$ values must be smaller than 1, effectively the surface must not be moved more than one grid spacing in a single time step in order to guarantee stable numerical solutions.

In the case of sparse-field level sets, only LS values with an absolute value lower than 0.5 are stored. If an active point $\vec{a}$ is advected according to the maximum set by Eq. (2.34), the minimum value the point may have is $\phi(\vec{a}, t + \Delta t) = 0.5$, while the maximum value is 1.5. This means that the surface will no longer be located between this point and one of its neighbours after advection, meaning there will be no valid zero level set layer $\mathcal{L}_0$. As this layer contains all the information about the surface, a valid LS cannot be formed robustly anymore. Although the $\mathcal{L}_0$ layer could be reconstructed from neighbouring layers, this may lead to large numerical errors and hence a stricter CFL condition must be chosen for sparse-field level sets [47]. The maximum change in $\phi(\vec{g}, t)$ during one time step is then given by:

$$\max|\hat{H}(\phi(\vec{g}, t))\Delta t| = \max|\phi(\vec{g}, t + \Delta t) - \phi(\vec{g}, t)| \leq 0.5 \quad . \tag{2.35}$$

In practice, this condition is satisfied by reducing the time increment Δt until the change in $\phi(\vec{g}, t)$ becomes small enough. Therefore, the time integration and the solution to $\hat{H}$ must be recomputed every time any part of the surface moves by more than half the grid spacing, or more than the predefined CFL condition. The advection must thus be repeated until the simulation time has advanced to the full process time of the simulation fabrication step. Note that Eq. (2.34) or Eq. (2.35) must be satisfied regardless of the numerical scheme chosen for the solution of the LS equation. Therefore, this type of advection must always be conducted by taking multiple time steps, which is why it is referred to as iterative advection.

The change of LS values for a time step depends on the Hamiltonian which is a differential equation in space and must be solved using appropriate spatial schemes. According to the definition in Eq. (2.6), the implicit function is stored on a rectilinear grid with grid points $\vec{g}$, which can be used to solve $\hat{H}$. In the following, several numerical schemes used to approximate the Hamiltonian will be discussed.

2.4.2.3 Engquist-Osher Scheme

Upwind schemes are one of the earliest-developed methods to numerically solve hyperbolic partial differential equations (PDEs) [93]. After Engquist and Osher proposed their finite difference numerical scheme for stable and entropy satisfying flow in 1980 [94], Osher and Sethian employed a version of this upwind scheme to solve the LS equation in 1988 [42]. Upwind schemes use one-sided approximations to a PDE based on the direction of flow of information of the system. A simple example is the one-dimensional LS equation

$$\frac{\partial\phi(x,t)}{\partial t} - V(x,t)\frac{\partial\phi(x,t)}{\partial x} = 0 \quad . \tag{2.36}$$

Depending on the sign of $V(x,t)$, the surface and thus the information propagates towards positive or negative x. The direction the information is propagating towards is called downwind, while the direction the information coming from is called upwind. In this scheme, the solution is approximated by only considering the information of the initial equation upwind of x giving this scheme its name.

Using the definition of the forward and backward difference from Eq. (2.12), the numerical derivative $\partial_x \phi(x,t)$ of the 1D problem in Eq. (2.36) is defined for the upwind scheme as

$$\partial_x^1 \phi(g,t) \approx \begin{cases} D_x^-(\phi(g,t)) & \text{if } \operatorname{sgn}(V(g,t)) = \operatorname{sgn}(D_x^-(\phi(g,t))) \\ D_x^+(\phi(g,t)) & \text{otherwise} \quad , \end{cases} \tag{2.37}$$

where $\operatorname{sgn}(x)$ is the signum function which returns -1 for negative x and $+1$ for positive x. The superscript 1 in ∂_x^1 denotes that this is the first order approximation to the partial derivative. Whether the forward finite difference D_x^+ or the backward finite difference D_x^- is chosen thus depends on both the sign of the scalar velocity field and the sign of the finite differences themselves. This is expected because a change in the sign of finite differences means that the surface is facing in the opposite direction. If this is the case, the direction of the advection and thus the propagation of information is also changed, meaning the other finite difference must be chosen in order to use the upwind side. The definition given here is valid only for the convention that the LS values inside the surface are negative. If the opposite convention is chosen, the forward and backward differences simply need to be exchanged to achieve the correct result.

In dimensions higher than 1, the spatial derivative is given by the ℓ_2 norm of all components of the gradient, as defined in Eq. (2.30), and the Hamiltonian is thus

$$\hat{H} \approx V(g,t) \sqrt{\sum_{i=1}^{D} \partial_i \phi(g_i,t)} \quad , \tag{2.38}$$

where g_i is the i^{th} component of the grid point coordinate $\vec{g}$ and ∂_i is the finite difference in the dimension i as defined in Eq. (2.37). The result of this numerical scheme is first order accurate in space. If higher precision is needed, the second order scheme can be applied by including an additional term in the numerical derivative:

$$\partial_i^2 \phi(g,t) \approx \begin{cases} D_i^-(\phi(g,t)) + \frac{\Delta g}{2} D_i^{--}(\phi(g,t)) & \text{if } \operatorname{sgn}(V(g,t)) = \operatorname{sgn}(D_i^-(\phi(g,t))) \\ D_i^+(\phi(g,t)) - \frac{\Delta g}{2} D_i^{++}(\phi(g,t)) & \text{otherwise} \quad . \end{cases} \tag{2.39}$$

The second order term for the approximation of the derivative is calculated by applying two one-sided numerical differences and choosing the appropriate result

$$D_x^{\pm\pm}(\phi(g,t)) = \zeta\left(D_x^{\pm}\left(D_x^{\pm}(\phi(g,t))\right), D_x^{-}\left(D_x^{+}(\phi(g,t))\right)\right) \quad , \tag{2.40}$$

where ζ is a switch function selecting the correct upwind side using the absolute values of its arguments, the differential approximations:

$$\zeta(a,b) = \begin{cases} a & \text{if } |a| \leq |b| \\ b & \text{otherwise} \end{cases} \tag{2.41}$$

The final value of the Hamiltonian is then computed in the same way as for the first-order approximation shown in Eq. (2.38).

Non-convex Hamiltonians

Due to the nature of the upwind scheme, stability is only guaranteed for convex Hamiltonians [44]. Convexity of the Hamiltonian is defined as

$$\frac{\partial^2 \hat{H}}{\partial\phi_i \partial\phi_j} = \frac{\partial^2 V(\vec{x},t)||\nabla\phi(\vec{x},t)||_2}{\partial\phi_i\partial\phi_j} \geq 0 \quad \text{for all } i,j \quad , \tag{2.42}$$

where ϕ_i is the partial derivative of $\phi(\vec{x},t)$ with respect to the spatial coordinate $i \in \{1,...,D\}$. If the Hamiltonian is non-convex, wave-like properties may distribute through the simulation domain and thus result in unphysical geometries [95].

As discussed in Section 2.3.2.3, the surface normals of the LS are found by evaluating the partial derivatives in all directions. Therefore, surface normal dependent velocity fields may result in non-convex Hamiltonians leading to problematic numerical errors when using upwind schemes [95]. Therefore, the suitability of the Engquist-Osher scheme for the solution of the specific velocity field must be guaranteed by ensuring that the condition in Eq. (2.42) is satisfied for all derivatives $\partial\phi_i, \partial\phi_j$.

2.4.2.4 Lax-Friedrichs Scheme

Upwind schemes take one-sided approximations to the Hamiltonian depending on the direction of information propagation. The Lax-Friedrichs scheme uses central differences to obtain the gradient and thus achieves more accurate results than upwind schemes [96]. The numerical derivatives used to solve the Hamiltonian are defined as

$$\partial_i^{\pm}(\phi(\vec{g},t)) = D_i^{\pm}(\phi(g,t)) \mp \frac{\Delta g}{2} D_i^{\pm\pm}(\phi(g,t)) \quad , \tag{2.43}$$

where $D_i^{\pm\pm}$ is the second order term defined in Eq. (2.40). If only the first order approximation is used, this term can be omitted. Using these numerical differences, the Hamiltonian is approximated using

$$\hat{H} \approx V(\vec{g},t)\sqrt{\sum_{i=1}^{D}\left(\frac{\partial_i^-(\phi(\vec{g},t)) + \partial_i^+(\phi(\vec{g},t))}{2}\right)^2} - D_{LF} \quad , \tag{2.44}$$

where D_{LF} is an additional dissipation term defined as

$$D_{LF} = \sum_{i=1}^{D} \alpha_i \left(\frac{\partial_i^-(\phi(\vec{g},t)) - \partial_i^+(\phi(\vec{g},t))}{2}\right) \quad , \tag{2.45}$$

where α_i are the dissipation coefficients. The numerical dissipation is non-zero in regions of high curvature, i.e. where the LS function changes abruptly. Especially at discontinuities, such as sharp corners of the surface, unphysical oscillations may be introduced into the LS, which are balanced by this dissipation term [97]. In order to achieve robust results, the dissipation coefficients must be chosen correctly. If they are too small, unphysical behaviour is not suppressed and oscillations still occur on the surface. If they are too large, sharp features are smoothed unnecessarily. The correct α_i values can be chosen by considering that the scheme is stable if it is monotone [98]. This means that the Hamiltonian must have a positive gradient along each direction of change for every point in space. Using the partial derivative $\phi_i = \partial\phi(x_i, t)/\partial x_i$, several inequalities must be satisfied to ensure a monotone Hamiltonian:

$$\left|\frac{\partial \hat{H}}{\partial \phi_i}\right| \leq \alpha_i \quad \text{for } i \in \{1, ..., D\}, \text{ for all } \vec{x} \tag{2.46}$$

$$\Delta t \left(\sum_{i=1}^{D} \frac{\alpha_i}{\Delta g}\right) \leq 1 \quad \text{for all } \vec{g} \tag{2.47}$$

In order to chose appropriate values for the dissipation without unnecessary smoothing, these inequalities can be equated, i.e. choosing the largest possible time step Δt and the smallest allowed values for α_i. Since α_i is the partial derivative of the Hamiltonian, the inequalities can be expressed as [48]

$$\alpha_i = \max_{\text{all } \vec{g}} \left|\frac{\partial \hat{H}}{\partial \phi_i}\right| \quad \text{for } i \in \{1, ..., D\} \tag{2.48}$$

$$\Delta t = \min \left[\left(\sum_{i=1}^{D} \frac{\alpha_i}{\Delta g}\right)^{-1}\right] \tag{2.49}$$

If the scalar velocity field $V(\vec{x}, t)$ can be expressed analytically, all necessary values can be found straight-forwardly by solving the partial derivatives analytically [95].

Dissipation for Spatially Constant Velocity Fields

Usually, the velocity field is not given analytically, but rather it is defined only on the discretised grid as $V(\vec{g}, t)$. In this case, the correct values for α_i must be found numerically. In the simplest case, the velocity is constant in the simulation domain and the partial derivative of the Hamiltonian is simply

$$\alpha_i = \left|\frac{\partial \hat{H}}{\partial \phi_i}\right| = \left|V(t)\frac{\phi_i}{||\nabla\phi(t)||_2}\right| = \max_{\text{all } \vec{g}} |V(t) n_i(\vec{g})| \quad , \tag{2.50}$$

where $n_i(\vec{g})$ is the i^{th} component of the surface normal vector which, according to Eq. (2.11), can be written as $\frac{\phi_i}{||\nabla\phi||_2}$.

Dissipation for Normal Vector Dependent Velocity Fields

If the velocity field does depend on the normal vector of the surface and cannot be expressed analytically, Eq. (2.50) cannot be used. In this case, the values for α_i must be generated by solving the partial derivative of the Hamiltonian numerically. For simplicity, the dependency of the velocity field $V(\vec{n}, t)$ on the normal vector and the time will not be written explicitly in the following. Since the velocity field is now dependent on ϕ_i, the Hamiltonian is differentiated according to the product rule:

$$\frac{\partial \hat{H}}{\partial \phi_i} = \frac{\partial V}{\partial \phi_i} ||\nabla\phi||_2 + V \frac{\partial ||\nabla\phi||_2}{\partial \phi_i} \quad , \tag{2.51}$$

where the second term is simply the result obtained in Eq. (2.50) for constant velocity fields. The velocity field may depend on all components of the normal vector n_j and thus on all ϕ_j. Therefore, the differential of the velocity field with respect to ϕ_i may be written as

$$\frac{\partial V}{\partial \phi_i} = \sum_{j=1}^{D} \frac{\partial V}{\partial n_j} \frac{\partial n_j}{\partial \phi_i} \quad . \tag{2.52}$$

Differentiating all terms then leads to

$$\frac{\partial V}{\partial \phi_i} = ||\nabla\phi||_2^{-3} \left[\frac{\partial V}{\partial n_i} \left(\sum_{j \neq i} \phi_j^2 \right) - \sum_{j \neq i} \frac{\partial V}{\partial n_j} \phi_i \phi_j \right] \quad . \tag{2.53}$$

Using the above results, the dissipation can be rewritten as

$$\alpha_i = \left| V n_i + \frac{\partial V}{\partial n_i} \left(\sum_{j \neq i} \frac{\phi_j^2}{||\nabla\phi||_2^2} \right) - \sum_{j \neq i} \frac{\partial V}{\partial n_j} \frac{\phi_i \phi_j}{||\nabla\phi||_2^2} \right| \quad , \tag{2.54}$$

where the first term is simply the result for spatially constant V, the second term is the derivative in the i direction proposed by Montoliu et al. [99] and the third term encompasses cross-directional derivatives as proposed by Toifl et al. [100] within the scope of this work. All terms can be found directly from ϕ, except the $\partial V / \partial n_i$ terms. These must be solved numerically, which can be achieved using a central difference scheme [99]. The derivative is then approximated as

$$\frac{\partial V}{\partial n_i} \approx \frac{V(n_i + \Delta n) - V(n_i - \Delta n)}{2\Delta n} \quad , \tag{2.55}$$

where all other components of the normal vector are kept constant. Δn is the finite difference, which is either given by the discretisation of the normal vector values or may be chosen freely. A stable numerical choice is

$$\Delta n = \epsilon^{\frac{1}{3}} V(\vec{n}, t) \quad , \tag{2.56}$$

where ϵ is the floating point accuracy of the simulator [101].

Dissipation for General Velocity Fields

A general velocity field may depend on the current time, the surface normal and the spatial coordinate and is thus defined as $V(t, \vec{n}, \vec{x})$. Due to the additional dependency on location $\vec{x}$, the derivative of V, with respect to ϕ_i in Eq. (2.52), must be extended by spatial terms:

$$\frac{\partial V}{\partial \phi_i} = \sum_{j=1}^{D} \left(\frac{\partial V}{\partial n_j} \frac{\partial n_j}{\partial \phi_i} + \frac{\partial V}{\partial x_j} \frac{\partial x_j}{\partial \phi_i} \right) \tag{2.57}$$

To the best of the authors' knowledge no general expression for the solution of the second term has been presented to date. However, for certain applications, where the spatial dependence of the velocity field is much smaller than the dependence on the normal vector, the second term may be ignored. In certain types of epitaxial growth, arriving molecules may diffuse along the surface for large distances before reacting with the substrate, meaning that there is no large change of growth rate with distance. This type of growth is, however, strongly dependent on the surface normal, such that most of the dissipation will stem from the normal vector dependence. As $\partial V/\partial x_j$ is much smaller than $\partial V/\partial n_j$, the spatial component does not contribute significantly to the dissipation. Hence, ignoring the spatial component may be viable for some simulations, although a general solution incorporating the spatial dependence of V would certainly be favourable. When there is a spatial dependence, but the velocity does not depend on the surface normal, the Hamiltonian is convex as defined by Eq. (2.42) and the Upwind scheme presented in Section 2.4.2.3 can be used instead.

Stencil-Based Schemes

Using the above definitions for α_i, global dissipation coefficients are chosen based on the maximum value encountered in the entire simulation domain, which is why this technique is referred to as Global Lax-Friedrichs scheme. However, this may lead to large values of α_i and thus to unnecessary smoothing in large parts of the surface, as high curvatures might only occur in small regions of the surface.

The gradient of the Hamiltonian is a local property of the implicit function and thus, the Global Lax-Friedrichs scheme overestimates the values for α_i and therefore leads to high smoothing [102, 103]. In order to reduce this smoothing effect, one may use only the neighbourhood of the current grid point for the calculation of the dissipation. Such a scheme was developed as a part of this work and is referred to as Stencil Local Lax-Friedrichs (SLLF) [96]. The dissipation is then defined as

$$\alpha_i(\vec{g}) = \left|\frac{\partial \hat{H}}{\partial \phi_i}\right|_S = \max_{\vec{g}\in S} |V(t) n_i(\vec{g})| \quad , \tag{2.58}$$

where S is the stencil made of the current point and neighbouring points. This method uses a 49-point stencil around the current grid point. However, smaller stencils are possible. For example, the Local Lax-Friedrichs (LLF) scheme proposed in [104] uses only the first neighbours, meaning 9 and 27 grid points in 2D and 3D, respectively. In the extreme case, only the current grid point is used [48], which is referred to as Local Local Lax-Friedrichs (LLLF). While LLLF requires the least computational effort, it usually underestimates the required dissipation coefficient and thus results in unphysical oscillations. LLF achieves satisfactory values for α_i, while keeping the computational requirements at a minimum. Therefore, this scheme is the most appropriate for the advection of general velocity functions $V(\vec{g}, t)$.

It is important to note that although stencil-based schemes are used to obtain spatially varying dissipation coefficients $\alpha(\vec{g})$, the size of the time step Δt still must be chosen based on the global α_i values, since the time step is a global property. Therefore, the correct size of the time step for stencil-based schemes is

$$\Delta t = \min_{\text{all } \vec{g} \in \mathcal{L}_0} \left| \left(\sum_{i=1}^{D} \frac{\alpha_i}{\Delta g} \right)^{-1} \right| \quad . \tag{2.59}$$

2.4.3 Geometric Level Set Advection

In Section 2.4.2, the evolution of a level set was modelled by considering the surface normal speed and solving the propagation of the surface iteratively. In each iteration, only a small time step can be applied, due to numeric restrictions of the underlying methods used to advance the surface. At most, the zero LS can be moved by half a grid spacing when using the sparse field approach. Hence, when a surface is represented at a high resolution, several hundred time steps might be necessary to advance the surface to its final location. Therefore, moving the interface using iterative advection commonly requires substantial computational effort, while introducing numerical errors [105]. Such errors stem from the approximations of partial derivatives, required to solve the level set equation, as discussed in the previous section.

If the propagation of time does not have to be modelled explicitly, meaning that a geometric relationship between the initial and the final surface is known, there is no need to move the surface in small steps. The resulting geometry can then simply be ”drawn” based on this geometric relationship, which is referred to as geometric advection or process emulation [106]. This way, perfectly isotropic deposition can simply be thought of as extending the surface outwards at every point by a constant distance. This method of emulating surface movement was first applied successfully using explicit voxel-based representations [107, 108] similar to those described in Section 2.3.3.2.

Within the scope of this work, the method of emulation was applied to the LS for the first time. Due to the higher resolution of the LS as compared to voxel-based surface representations, better results can be achieved with comparable computational effort. Additionally, this allows geometric advection to be combined with iterative advection to achieve a high flexibility with regard to the modelling capabilities of the implemented simulator. In this section, the underlying geometric advection algorithm, as developed during this work, is presented including considerations for practical applications, as well as fundamental limitations of this approach.

In order to familiarise the reader, the general approach of this method will first be explained using an explicit surface representation. Applying the same strategy to level sets, most algorithmic approaches remain the same and can be applied to the implicit surface.

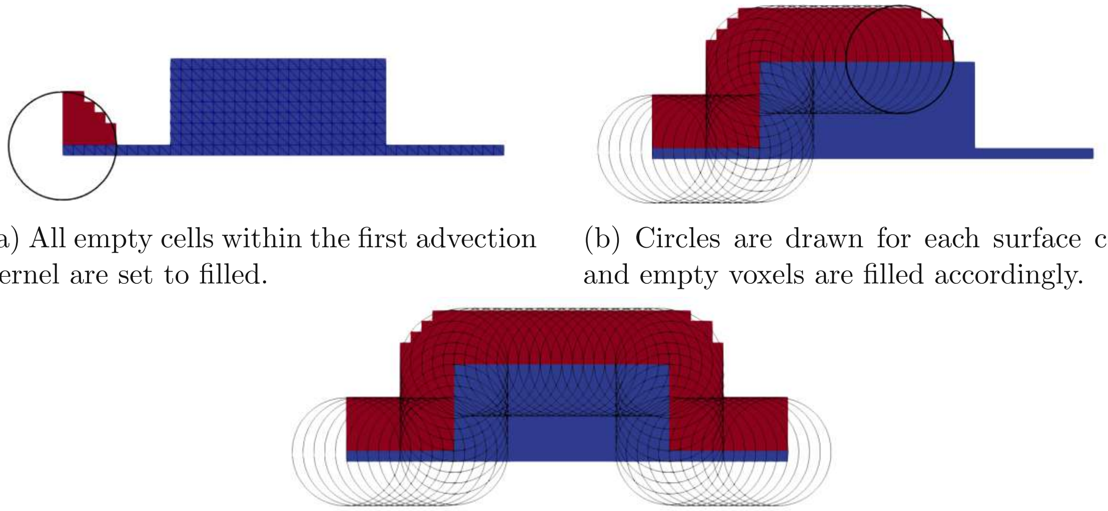
2.4.3.1 Geometric Voxel Advection

For voxel-based advection, the initial surface is described as a Cartesian grid of cells, where each cell is filled with an integer denoting which material it contains. Usually, partially filled cells are not permitted [109], since this would strongly increase the complexity of the algorithm and thus lead to increased computational effort [82]. For a perfectly conformal deposition process, the final surface can be straight-forwardly generated by considering the distance between the filled and the empty cells. Algorithmically, this is achieved by first identifying all cells at the surface by checking whether a voxel has a neighbouring empty cell. If this is the case, the voxel must lie on the surface.

Subsequently, as shown in Fig. 2.17a, a circle is drawn, centred at the interface between a filled voxel and its empty neighbouring voxel. This circle represents the so-called advection kernel or advection distribution, which encompasses all the geometric information of the emulated process. A circle corresponds to uniform surface movement in all directions and is thus used to emulate a perfectly isotropic deposition processes.

After the advection kernel has been placed at the interface, all empty cells within the kernel are set to filled, as indicated in Fig. 2.17a. Depending on the exact shape of the advection kernel, different processes can be emulated. Repeating this procedure for all points on the surface leads to the formation of a new material which is a geometrically accurate representation of an isotropically deposited film on top of the initial material.

Fig. 2.17 shows this procedure after the first kernel was drawn, at some point in time during the advection and the final result showing the entire new material. Some of the problems of the geometric advection method are also visible in this figure. Due to the limited resolution of the voxel-based material representation, sub-grid distances cannot be represented accurately [78]. This leads to the small gap between the advection kernels and the new material on the top surface and to stepped corners which should be smooth and round. The metric used to determine whether a voxel is inside or outside of the advection kernel also has a great impact on the final result. If the centre point is used, more than half of the volume of the voxel might actually be filled with the new material, but the voxel is left empty because the centre is not inside the new material. Additionally, the top surface in Fig. 2.17c is not advected as far as the advection kernels reach and thus the new surface is not moved far enough. These small inaccuracies can accumulate and ultimately lead to large discrepancies in the final result. Therefore, while the geometric voxel advection offers great computational efficiency and is easily implemented, the limited resolution and possible numerical issues make it unfavourable for many applications [79].



(a) All empty cells within the first advection kernel are set to filled.

(b) Circles are drawn for each surface cell and empty voxels are filled accordingly.

(c) Once all circles have been drawn, their union and thus the resulting geometry accurately represents the result of a perfectly isotropic deposition process.

Figure 2.17: Advection of a voxel-based explicit mesh for the emulation of an isotropic deposition process. The centre of each advection kernel is found by taking a filled voxel with a neighbouring empty cell and placing the centre of the kernel at their interface. Several deposition kernels (black circles) are used at the interface between the filled initial cells (blue) and empty cells to draw the new surface. All empty cells within a distribution are filled with the new material.

2.4.3.2 Geometric Advection Distributions

In the example discussed in the previous section, a geometric advection distribution was used to describe the geometric effect the modelled process has on the initial surface. The final surface, after isotropic deposition, was emulated using spherical advection distributions placed on the initial surface. The circle in 2D, or sphere in 3D, implies that the surface is moving at the same speed in all directions. Thereby, the radius of the sphere sets how far the surface has been advanced by the end of the modelled process.

In order to capture the specific geometric behaviour of each process, it must be expressed as a geometric distribution. Given the geometric characteristics of the modelled process, such a distribution must describe how a specific point on the initial surface $\vec{x}_s$ affects the final interface. In the case of isotropic deposition, it is most natural to express the effect on the final surface in terms of the spherical angles θ_{inc} and ϕ_{az}

$$D_{iso}(\vec{x}_s, \theta_{inc}, \phi_{az}) = R_{iso} \quad , \tag{2.60}$$

where R_{iso} is the deposition distance. Since this advection distribution is independent of $\vec{x}_s$ and the spherical angles, it is isotropic and thus has the same effect everywhere on the initial surface. If the growth distance of a deposition process does depend on some locally varying surface property $F(\vec{x}_s)$, but is isotropic nonetheless, the radius of the spherical deposition will simply be scaled by this property using

$$D_{iso}(\vec{x}_s) = F(\vec{x}_s) R_{iso} \quad . \tag{2.61}$$

If the process is not isotropic and therefore has some directionality, the geometric distribution must have a non-spherical shape. There are no general limitations on the shape or complexity of the spherical distribution, other than it being single valued in all $\vec{x}_s, \theta_{inc}, \phi_{az}$. This must be the case as it represents the distance by which the surface at $\vec{x}_s$ would move under the modelled process to form the final interface, if there were no other initial surface points. As the surface can only be moved by a single distance in each direction, the geometric advection distribution must be single-valued.

For certain distributions it may be more natural to express them in terms of other coordinate systems in which it is easier to represent the shape of the distribution. For example, perfectly directional processes which only move the surface in one spatial direction and not in any other are more naturally expressed using Cartesian coordinates. However, no matter what coordinate system is used to represent the geometric distribution, it must be single valued for all combinations of its arguments.

Signed Distance Calculation Using Geometric Distributions

By considering this definition of geometric advection distributions, the voxel-based approach presented in the previous section (see Fig. 2.17) can be extended to the LS method [106]. The simplest approach would be to generate the implicit surface of a geometric advection distribution, centre it at each initial surface point and then use Boolean operations to generate the final surface. This would entail iterating over the initial surface once for every grid point of the final surface, making it computationally inefficient. However, by considering the geometric nature of the SDF used to describe implicit surfaces, it is possible to construct the final surface directly.

Using the geometric approach, a point on the final surface does not evolve from its neighbouring points but is rather set directly. Therefore, the signed distance d_s of each active grid point $\vec{a}$ on the new surface may be calculated independently of all others. To calculate the signed distance value, first a possible new surface point $\vec{P}_{cand}$ referred to as candidate point is selected, as shown in Fig. 2.18.

Next, the signed distance value of this point is calculated using each initial surface point $\vec{P}_{cont}$, called contribute point, at a time. First, the geometric distribution is centred at $\vec{P}_{cont}$, shown in Fig. 2.18 for an isotropic deposition. In order to calculate the signed distance d_s, the distance vector $\vec{v}$ from $\vec{P}_{cont}$ to $\vec{P}_{cand}$ is calculated. The geometric advection distribution is then used to find the advection distance of the process in the direction of $\vec{v}$. The signed distance of the new surface to the point $\vec{P}_{cand}$ is then given by

$$d_s(\vec{v}, D(\vec{x}_s, \vec{v})) = |\vec{v}| - D(\vec{x}_s, \vec{v}) \quad . \tag{2.62}$$

In the case of isotropic deposition, given in Fig. 2.18, this simplifies to

$$d_s(\vec{v}) = |\vec{v}| - D_{iso} = |\vec{v}| - R_{iso} \quad . \tag{2.63}$$

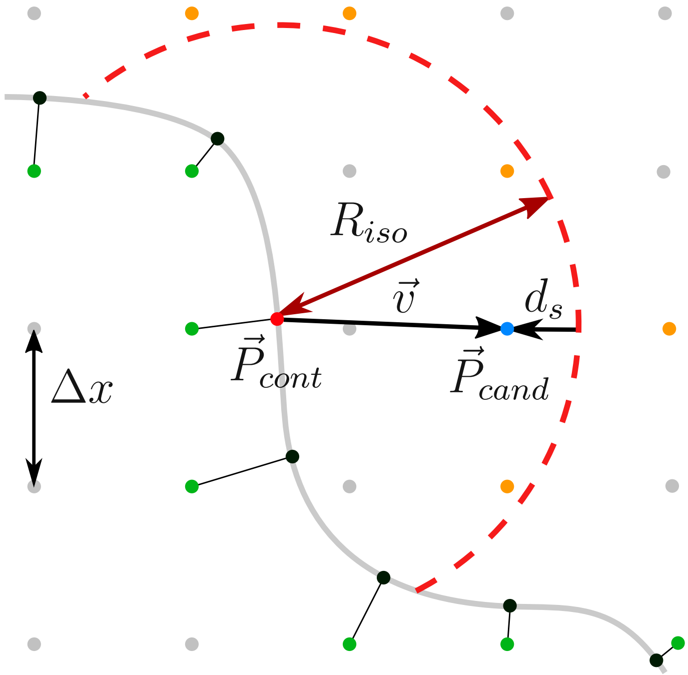


Figure 2.18: A geometric distribution (dashed red) describing isotropic growth, used to calculate the signed distance d_s for a candidate point $\vec{P}_{cand}$ (blue) from one point of the initial surface $\vec{P}_{cont}$ (red). The growth distance R_{iso} is equal for all directions of the distance vector $\vec{v}$. The set of all initial points are shown in black and the set of all candidate points in orange.

Hence, the signed distance d_s of the final surface only depends on $|\vec{v}|$, the distance to the initial surface, but not on direction or location, as one would expect for isotropic growth everywhere on the surface.
Since $\phi(\vec{x})$ is scaled to the grid spacing, the new LS value at $\vec{P}_{cand}$ is given by

$$\phi(\vec{P}_{cand}, t_{\text{final}}) = \frac{|\vec{v}| - D(\vec{x}_s, \vec{v})}{\Delta g} \quad , \tag{2.64}$$

which, for the isotropic case, simply becomes

$$\phi(\vec{P}_{cand}, t_{\text{final}}) = \frac{|\vec{v}| - R_{iso}}{\Delta g} \quad . \tag{2.65}$$

Once this calculation has been performed with all possible active points $\vec{a}$ of the new surface by setting them as $\vec{P}_{cand}$, the new LS values $\phi(\vec{a}, t_{\text{final}})$ represent a correctly initialised LS describing the final surface. The way in which all the possible combinations of contribute and candidate points are merged to form the final level set will be discussed in the following.

2.4.3.3 Geometric Advection Algorithm

The purpose of the geometric advection algorithm is to identify all necessary candidate points of the new surface and use the geometric advection distributions to find the correct LS value for each of these candidate points. Given the set of possible LS values $\phi(\vec{P}_{cand}, \{\vec{P}_{cont}\})$ for each candidate point, the algorithm must identify the correct final value for $\phi(\vec{P}_{cand})$. In the following, the algorithm will be presented using the isotropic deposition example of the previous section for simplicity. Differences in the algorithm for general geometric advection distributions will be discussed thereafter.

Condition for Choosing the Final LS Value

Isotropic deposition corresponds to a positive and constant surface velocity function. Therefore, the velocity field $V(\vec{x}, t) = R$ for $R \geq 0$ in Eq. (2.30) and the numerical LS equation may be written as

$$\phi(\vec{g}, t_{\text{final}}) = \phi(\vec{g}, t) - R||\nabla\phi(\vec{g}, t)||_2 \quad . \tag{2.66}$$

By definition, $||\nabla\phi(\vec{g}, t)||_2 > 0$ and R must be positive. Therefore, all level set values must decrease when advected under a velocity field describing isotropic deposition. Intuitively, this can be imagined as more and more grid points moving inside the surface and thus becoming negative, while grid points outside of the surface are closer to the surface than before and thus their LS value must be smaller.

As all LS values must decrease, the smallest value of a set of possible level set values for a point $\vec{P}_{cand}$ correctly describes the final LS since it implies the farthest movement of the surface:

$$\phi(\vec{P}_{cand}, t_{\text{final}}) = \min\left(\phi(\vec{P}_{cand}, \{\vec{P}_{cont}\}, t_{\text{final}})\right) \quad . \tag{2.67}$$

Note that this only holds for a deposition process, since all LS values decrease in size. The opposite is the case for an etching process, where the maximum value must be chosen, since every LS value must increase.

The Algorithm

Using the above definition for choosing the correct LS values, the algorithm in Fig. 2.19 is used to generate the final surface only from the initial LS and a geometric advection distribution.

The algorithm is described in detail in the following, with the most important steps depicted in detail in Fig. 2.20:

1. Generate Explicit Points: First, points on the initial surface must be generated so geometric advection distributions can be centred on them. The resulting point cloud should represent the surface with enough accuracy to not loose any features. However, since the final LS has a finite resolution, numerous additional points will not provide added benefit, while increasing the number of signed distance calculations and therefore computational effort. The ideal resolution of the initial surface is thus on the order of the grid spacing. As presented in Section 2.3.2.3, the LS can be used to quickly generate points on the surface using the closest point approximation [60]. Since grid points are used to generate these points, they are automatically separated by roughly one grid spacing. Therefore, centring one geometric advection distribution at each point will produce the expected final surface.

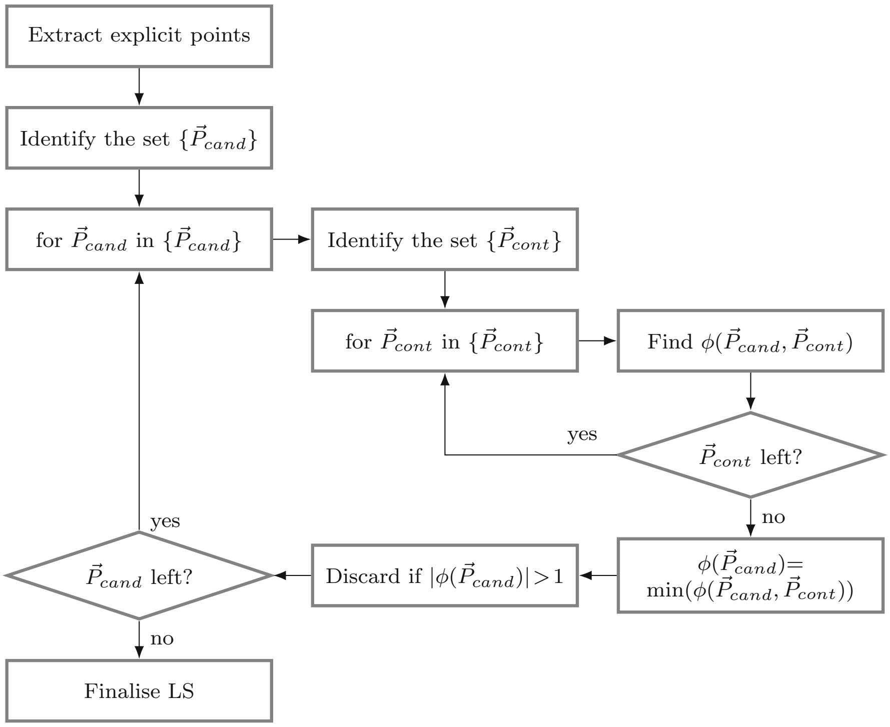

Extract explicit points
Identify the set $\{\vec{P}_{cand}\}$
for $\vec{P}_{cand}$ in $\{\vec{P}_{cand}\}$
Identify the set $\{\vec{P}_{cont}\}$
for $\vec{P}_{cont}$ in $\{\vec{P}_{cont}\}$
Find $\phi(\vec{P}_{cand}, \vec{P}_{cont})$
yes
$\vec{P}_{cont}$ left?
no
yes
$\vec{P}_{cand}$ left?
Discard if $|\phi(\vec{P}_{cand})| > 1$
$\phi(\vec{P}_{cand})=$ $\min(\phi(\vec{P}_{cand}, \vec{P}_{cont}))$
no
Finalise LS

Figure 2.19: Flowchart of the full geometric advection algorithm for a deposition process.

2. Identify Candidate Points: Find the grid points, which will likely be active grid points of the new level set and thus may be involved in describing the final surface. These are shown as orange and blue points in Fig. 2.20a and should represent the minimal set of new LS points. Ideally, the set of candidate points $\{\vec{P}_{cand}\}$ is exactly equivalent to the set of active points $\{\vec{a}\}$ of the final LS . While it is possible to treat all grid points as candidate points, keeping the number of candidate points low is key to an efficient algorithm, as it drastically reduces the number of time-consuming calculations of $\phi(\vec{P}_{cand}, \vec{P}_{cont})$. An efficient implementation for identifying the minimal set of candidate points is discussed in Section 4.2.7.1.

3. Iterate over Candidate Points:

   (a) Identify Contribute Points: For each candidate point, the set of contribute points which might affect the formers final value must be identified. Again, this could include all initial surface points, however this set should also be as small as possible in order to save computation time. Since active points of the new surface must lie on the boundary of the geometric advection distribution, a simple bounding box approach can be used to identify possible contribute points, as shown in Fig. 2.20b. Depending on the exact shape of the geometric advection distribution, there might be more efficient methods to identify contribute points given a specific candidate point. However, the bounding box approach can be used universally and thus provides a robust strategy while being reasonably efficient for most advection distributions encountered in process simulation, such as spheres, boxes, or ellipsoids.

(b) Calculate New Level Set Value: For each of the just-identified contribute points, the LS value which they would produce at the candidate point is calculated. This means that $\phi(\vec{P}_{cand}, \vec{P}_{cont}, t_{\text{final}})$ is evaluated as shown in Fig. 2.20c. Using Eq. (2.67), the lowest of these values is chosen as the final LS value $\phi(\vec{P}_{cand}, t_{\text{final}})$. This final value is then checked to see whether the current candidate point will be an active point on the final surface. If it is not, meaning $|\phi(\vec{P}_{cand}, t_{final})| > \Delta g$, the candidate point will be discarded.

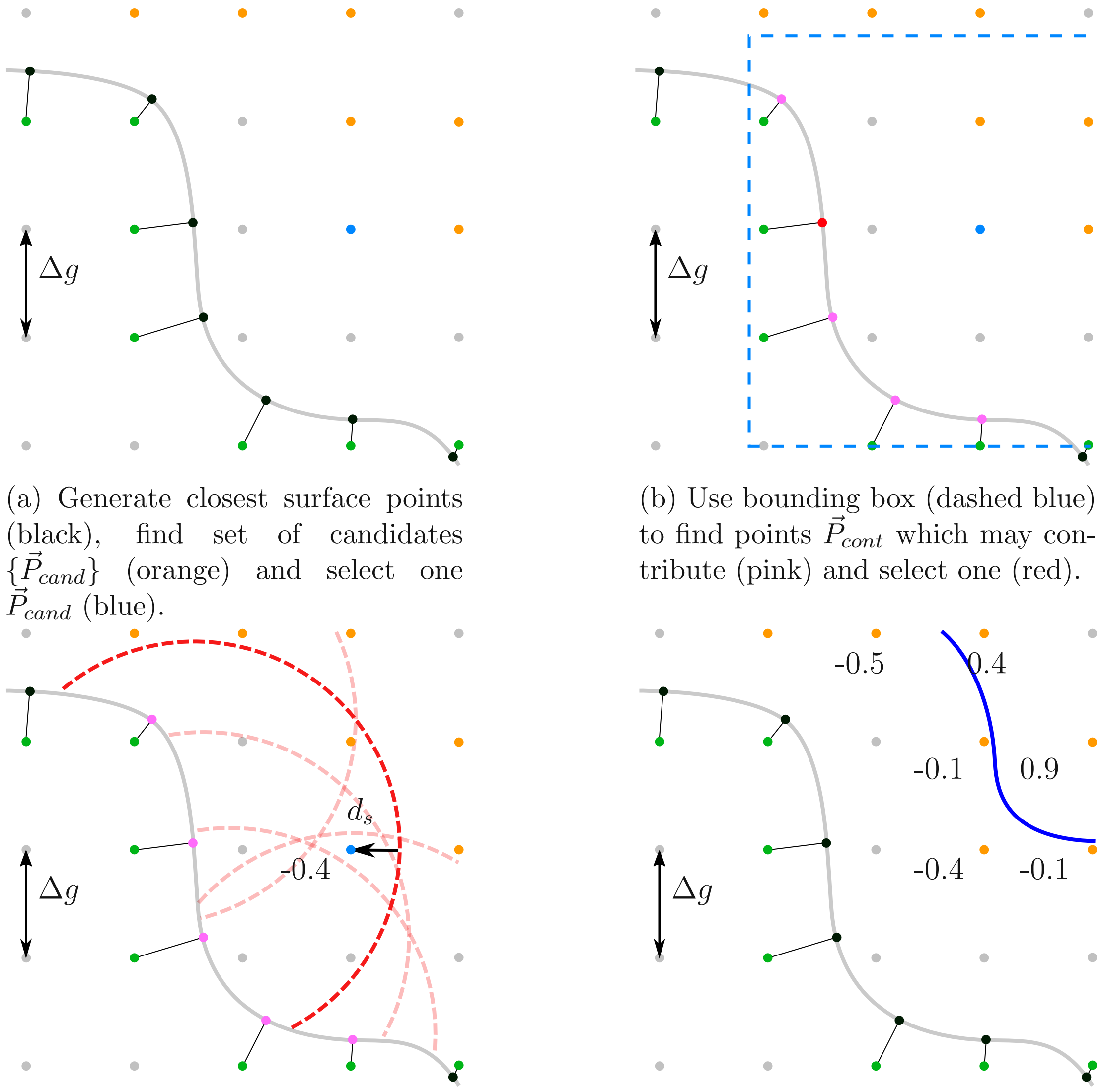
4. Finalise Level Set: Once all candidate points have been considered and their LS values have been calculated, they automatically form a valid new surface. The values of the candidates which were discarded are not required and are thus not considered in the final surface, forming an efficient sparse-field LS. However, the final surface still contains LS values larger than 0.5, which are not required for sparse-field LSs and could be removed in a final step. Nevertheless, in regions of high curvature it is beneficial to not only include the $\mathcal{L}_0$, but also the $\mathcal{L}_1$ layer and thus allow values up to 1 in order to achieve a more robust surface description. Since these values are still required for the algorithm and would have to be removed in an additional step which does not provide great benefits, it is recommended that these values are kept in the final result.

2.4.3.4 Adaptions for Specific Geometric Advection Distributions

The criterion for choosing the correct $\phi(\vec{P}_{cand}, t_{\text{final}})$ from the set $\phi(\vec{P}_{cand}, \{\vec{P}_{cont}\}, t_{\text{final}})$ was defined in Eq. (2.67). For a deposition process, the lowest value leads to the expected surface description, since all values in the LS must decrease. Choosing the minimum of the set of values gives the correct result because this value reflects the farthest movement of the surface under the advection, meaning the contribute point which contributes most to the deposition is kept.

However, this is not the case for etching processes, where all LS values must increase and thus the maximum value corresponds to the farthest movement. Therefore, the criterion for choosing the correct final value changes depending on the geometric advection distribution which must be known in advance, so the same criterion can be applied for all candidate points. Whether the geometric advection distribution describes deposition or etching, is denoted by its sign. In this work,the convention of negative values indicating etching is used.

Since the criterion for choosing the correct LS values must be known in advance, the modelled process cannot include simultaneous deposition in one region and etching in another, as is the case in modern plasma processes [110]. However, such a process can still be modelled by splitting it into two different geometric advection steps, one for etching, followed by a second geometric advection for deposition.



(a) Generate closest surface points (black), find set of candidates $\{\vec{P}_{cand}\}$ (orange) and select one $\vec{P}_{cand}$ (blue).

(b) Use bounding box (dashed blue) to find points $\vec{P}_{cont}$ which may contribute (pink) and select one (red).

(c) Calculate $\phi(\vec{P}_{cand})$ for all $\vec{P}_{cont}$ and choose the lowest as the final value (blue).

(d) When all $\phi(\vec{P}_{cand})$ are calculated, a valid new LS has been constructed automatically.

Figure 2.20: Visualisation of how the algorithm generates the LS describing the final surface from a point cloud of the initial surface.

2.4.3.5 Geometric Advection of Multiple Materials

In order to use the geometric advection algorithm in microelectronic simulations, it is essential that multiple materials and differing advection distributions based on the current material can be handled appropriately. Using the layer-wrapping approach described in Section 2.3.2.5, the initial surface must include all lower material and thus represents the union of all materials in the simulation domain. Given a point on the initial surface $\vec{x}_s$ and the distance vector to a candidate point $\vec{P}_{cand}$, the material for both can be deduced straight-forwardly. Since the geometric advection distributions are a function of both, the distribution may change according to the material of the initial surface as well as of the final surface at $\vec{P}_{cand}$. If, for example, a part of the initial surface is covered by a masking material, the geometric advection distribution at these points can be set to zero, leading to the same behaviour as if the material had been masked. Equally, if a material should be etched until an etch stop layer is reached, the geometric advection can be set to $\phi_{\text{etch stop}}$ if $\vec{P}_{cand}$ is part of this layer, thus leaving the etch stop material intact. Therefore, no special consideration or change in the algorithm is necessary to incorporate multiple materials since all the information is already included in the geometric advection distribution.

2.4.3.6 Time-varying Processes

Using geometric advection, the movement of the surface is only described by a distance rather than a velocity used for iterative advection. Therefore, velocity functions which vary in time cannot be modelled using the geometric advection approach without more detailed considerations. The most important limitation is that a geometric advection distribution must be single-valued in all its arguments. Thus, in order to represent this surface movement using a geometric advection, the integral of the surface movement over time must be single valued in every spatial direction from the initial surface. If this is not the case, the process can usually still be modelled using geometric advection, but must be split into several sequential steps of geometric advections. How many steps are necessary and how exactly the process should be split up must be considered for every modelled process individually.

2.4.4 Volume-Dependent Modelling

Several important manufacturing processes lead to changes deep within materials, instead of just on their surfaces and interfaces. Some diffusion processes, such as ion implantation [111] or thermal annealing [112], might not affect the interface but rather result in a change of properties inside the material, while others might lead to a change of the interface which is controlled by changing densities inside the material, such as oxidation [113]. The doping concentration may also heavily influence etch rates [114], which is why volume-dependent modelling may be important even when only topographical changes should be simulated.

In order to model these processes, a volume representation is necessary to store all the relevant information wherever it is needed. A simple and commonly applied solution is using a triangular [115] or tetrahedral mesh [113], where each triangle or tetrahedron stores all the necessary information. However, this means that the entire volume has to be represented, which is memory intensive. Since diffusive processes act by diffusing through the material from the surface, usually only the volume directly under the surface needs to be represented in order to model the process appropriately.

A cell-based representation, as discussed in Section 2.3.3.2, is compatible with the level set and can be used to store volume data. Therefore, a sparse approach was developed within this work, similar to the sparse-field LS, in which the volume is split into voxels, centred around a grid point, and information is stored only at required grid points. Using this cell-based representation on the same grid as the LS thus leads to an accurate and robust description of volume properties and their evolution in time. If the change in volume property, such as strain or doping, results in a topographical change, a velocity field $\vec{V}(\vec{g})$ can be generated from the cell-based representation and used to robustly move the LS surface. Converting the final LS-based result to filling fractions, as described in Section 2.3.3.2, allows to obtain the final cell-based volume description. This flexible approach allows any type of volume data to be stored and manipulated by different algorithms depending on the application. Therefore, this design allows for ion implantation, diffusion and mechanical stress to be modelled using the most appropriate algorithm, while still keeping the benefits of a LS surface description and minimizing the need for mesh conversions. Since both material representations only store sparse data, this approach is also highly memory efficient.

2.5 Summary

In this chapter, the continuum modelling approach for the description of wafer materials was found to be the most appropriate for the requirements of the developed simulator.

Different numerical representations for continuous materials were introduced and their relative advantages discussed. The sparse-field level set method was deemed appropriate for the description of material interfaces, due to its minimal memory requirements and the ability to extract geometric properties such as the surface normal and the curvature straight-forwardly. Additionally, Boolean Operations can be carried out with optimal efficiency, requiring every defined grid point to only be visited once. For the representation of multiple materials, the layer wrapping scheme was found to be the most appropriate to minimise numerical errors.

Volume-based material descriptions were reviewed and parallels between the level set method and cell-based methods were highlighted. A surface described by the LS can straight-forwardly be converted to cell-based filling fractions. However, due to the latter not storing enough information to generate surface normals, a conversion back to a level set is not possible unambiguously.

Next, the modelling of material evolution was discussed and the level set was deemed the most appropriate for the description of semiconductor fabrication processes. The merger and separation of level set surfaces leads to appropriate results without further considerations, as self-intersections are avoided entirely. Different numerical schemes for level set advection were presented, including the Stencil Local Lax-Friedrichs scheme for the robust advection of non-convex velocity fields. This numerical advection scheme was developed during the course of this work for strongly normal vector dependent material evolution, as encountered during anisotropic wet etching of silicon described in Section 5.1.3.

Next, geometric level set advection was presented and the underlying algorithm, which was developed wihtin the scope of this work, discussed in detail. In this work, for the first time, this efficient modelling approach for material evolution was applied to the level set method. The developed computationally efficient and robust algorithm was presented and limitations shown. It was found that the algorithm is the most appropriate for process emulation, were the geometric outcome of a processing step is known in advance, since it decreases computation time significantly when compared to iterative level set advection.

Finally, the general approach to volume-dependent modelling in the implemented simulator was discussed. A sparse cell-based volume data structure, defined on the same grid as the level set, was developed to achieve compatibility with the surface description. The volume description can then be used to model physical processes deep within materials and it can be combined seamlessly with the level set method to move material interfaces accordingly.

Chapter 3

Surface Rate Calculation

In order to move a surface, its evolution must be captured by a velocity field $\vec{V}(\vec{x}_s)$ defined at every point on the surface $\vec{x}_s$, as described in Section 2.4. In the case of the sparse-field level set method, this means that there must be a velocity field $\vec{V}(\vec{a})$ defining the surface movement for every active point $\vec{a}$ describing the interface. As all topographical changes of the surface resulting from a manufacturing process are captured in this velocity field, it is crucial that it is as accurate and as robust as possible. Analogously to physical growth or etch rates, these surface velocities are often referred to as surface rates, even when the physical behaviour is not considered directly. In order to capture these surface rates appropriately, process models are used to simulate how a fabrication process will affect the surface and thus give accurate approximations for the surface velocities. Two main approaches to modelling fabrication processes to find the surface rates are discussed in the following.

Firstly, empirical modelling, or process emulation, describes extracting geometric parameters, such as etch depth or isotropy from experimental data and using them to formulate simple models for the surface rates. When using the iterative advection described in Section 2.4.2, a velocity field can only be applied for a single advection step and then must be recalculated since the level set (LS) surface has changed and thus the velocity field must be adjusted as well. However, geometric process parameters may also be expressed as a geometric advection distribution, as described in Section 2.4.3, and the entire process could be emulated in a single step.

The second approach is chemical modelling and encompasses a rigorous description of all relevant physical processes leading to a change in topography by considering the underlying physical principles, including particle transport, surface kinetics and surface chemistry. Since chemical modelling requires the explicit simulation of process time, only iterative advection may be used to advance a LS surface, while geometric advection is not suited for this type of modelling.

Due to its simplicity, empirical modelling is usually computationally more efficient, while chemical modelling describes the underlying physics of a process and can thus be used for physical analysis or to simulate processes for which no experimental data is available.

3.1 Empirical Surface Rate Modelling

In this approach to finding the surface velocities $\vec{V}(\vec{x}_s)$ capturing the topographical changes, the result of a fabrication step must already be known and geometric parameters of the final geometry extracted. Once these parameters, such as etch depth, deposition distance, isotropy, or directionality are known, surface velocities are generated algebraically. Since no physical behaviour is modelled, but only geometric considerations are used to mimic the final surface, this approach is also referred to as process emulation [116]. A simple example is isotropic deposition of a material, which is modelled by setting the velocity field in the normal direction of the surface $V(\vec{x}_s)$ to the thickness of the deposited material. Applying iterative advection for a unit of time results in the final surface showing the expected deposition behaviour. If the surface is moved over large distances, this velocity field must be applied several times in order not to violate the Courant-Friedrichs-Lewy (CFL) condition.

However, for certain types of processes, it is possible to express the velocity field $\vec{V}(\vec{x}_s)$ as a geometric advection distribution. Then, for isotropic deposition, the initial surface is simply shifted outwards by a constant distance, resulting in a film of constant thickness, as if grown perfectly isotropically. There is no general limit on the complexity of a process modelled using geometric advection, although the algebraic descriptions required to capture the geometric properties of complex fabrication process may sometimes be described more naturally using a velocity field and applying it using iterative advection. Due to the simplicity and straight-forward calculations of the final surface, geometric advection is highly efficient and commonly employed for the generation of large structures in order to characterise entire microelectronic circuits[117, 118, 119]. Since time is not modelled explicitly, but just the initial interface and the final surface, the geometric advection algorithm discussed in Section 2.4.3 is the most appropriate method for simulating this type of model.

3.2 Chemical Surface Rate Modelling

The goal of this approach is to consider the physical processes which lead to the evolution of material interfaces. In order to obtain a surface velocity field $\vec{V}(\vec{x})$, the chemical reactions on the surface have to be modelled in order to express their topographical effect on the material through etch or deposition rates R. These reactions depend on the particles present at the considered surface site, expressed using surface coverages Θ_x denoting the fraction of the surface site covered by a molecule of type x. The distribution of coverages for each particle type is governed by the transport of this type in the gas phase through the reactor. Modelling this transport, the surface fluxes F_x for each particle type can be found in order to calculate the surface coverages Θ_x. Hence, as shown in Fig. 3.1, several different models have to be formed with the result of one model used as the input for the next. After all the physical models have been evaluated, the final result is the topographical change of the surface captured in the surface velocity field $\vec{V}(\vec{x})$.

This section will discuss these models and their physical footing in detail. The models will be presented in the order in which they would be performed during the simulation of a semiconductor fabrication process.

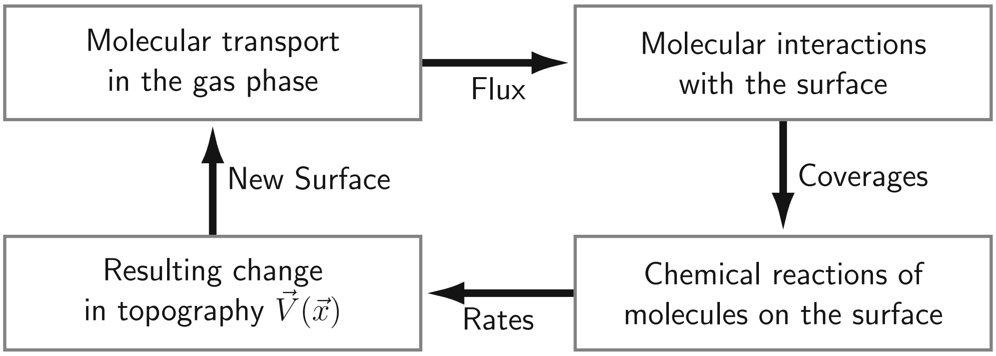

Molecular transport in the gas phase
Flux
Molecular interactions with the surface
Coverages
Chemical reactions of molecules on the surface
Rates
Resulting change in topography $\vec{V}(\vec{x})$
New Surface

Figure 3.1: Flowchart of the chemical models used to generate the surface velocity field $\vec{V}(\vec{x})$ and the information they provide for the next step (see arrows). First, the distribution of molecules impinging on the surface is modelled, which produces the incoming flux at every point on the surface. This flux is then used to model interactions with the surface in order to find how molecules are distributed leading to the surface coverage for each particle type. Using the number of molecules present at each surface site, their chemical reactions with the substrate are modelled, finally leading to the surface rates and thus the velocity field describing the topographical change. Applying these velocities to the initial surface using a single iterative advection step results in the final surface and the cycle starts anew.

3.2.1 Molecular Transport in Plasma Environments

The first step in the chemical description of a fabrication process, is calculating the rates at which different atoms, molecules, or ions impinge on the surface. In the context of process technology computer aided design (TCAD) modelling, these rates are referred to as particle flux and depend strongly on the different geometrical effects and transport phenomena inside the reactor [120, 121]. A thorough description of the thermodynamics of the fabrication process is necessary to capture all relevant phenomena. As already mentioned in Section 2.1.1, the reactor space is divided into a reactor scale and a feature scale region. Although reactor scale modelling is not the main scope of this work, the fundamental concepts are discussed in the following, focusing on the implications it has for feature scale modelling.

3.2.2 Reactor Scale Transport

As shown in Fig. 3.2, the simulation domain is split into reactor scale and feature scale regions, separated by the plane $\mathcal{P}$, referred to as source plane. The reactor scale thus encompasses all space inside the reactor, whose dimensions are at the very least on the order of the wafer size, i.e. 300 mm for modern processes. A simple parameter used to determine the transport regime of particles is to consider their mean free path in an ideal gas:

$$\bar{\lambda} = \frac{k_B T}{\sqrt{2}\pi d^2 P} \quad , \tag{3.1}$$

where k_B is the Boltzmann constant, T is the temperature, P is the pressure inside the reactor and d is the collision diameter of the gas molecule [122], which for the most commonly encountered particles is around 0.4 nm [123].

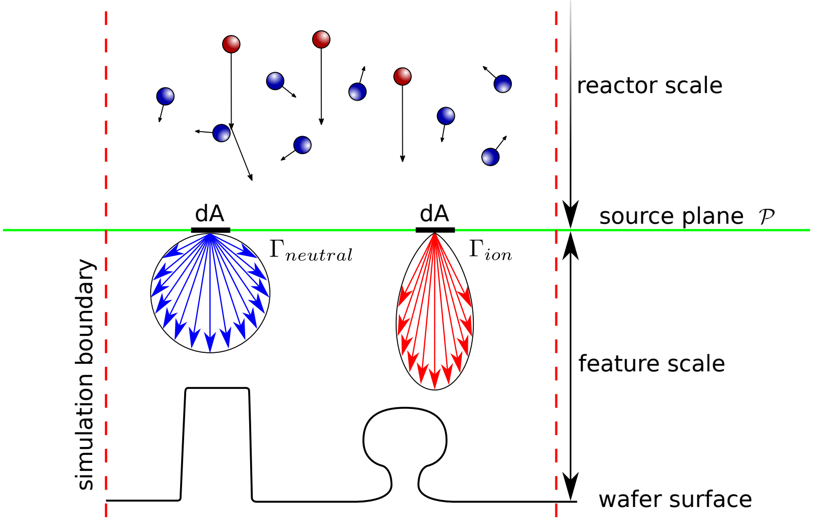


Figure 3.2: Schematic representation of the traversal of neutral molecules and ions through the reactor and feature scale regions. While the motion through the reactor scale is dominated by random collisions with other molecules, the path through the feature scale region is dominated by ballistic transport. The directional distribution of neutral atoms and molecules $\Gamma_{neutral}$ (blue) and ions Γ_{ion} (red) entering the feature scale region is shown as blue and red arrows, respectively. The molecular entities will then traverse the feature scale region in straight lines, only colliding with the surface.

In extreme cases, processes may operate at temperatures up to 1500 K and pressures down to 0.1 Pa [124], leading to a mean free path of $\bar{\lambda} \approx 290$ mm. For these types of processes, the exact reactor geometry and operation would have to be considered in order to generate accurate particle fluxes on the surface [125]. However, most microelectronic fabrication processes do not operate above 500 K or below 10 Pa, so the mean free path does not exceed $\bar{\lambda} \approx 10$ mm, meaning it is much smaller than the reactor region. Therefore, molecular transport is within the continuum regime and the directional distribution of particle motion is uniform, while the velocity follows the Maxwell-Boltzmann distribution [126]. Since particles are equally likely to move in any direction, the directional distribution of the particle flux crossing the source plane $\mathcal{P}$ towards the wafer is described by a cosine [60]

$$\Gamma_{neutral}(\theta) = F_{neutral} \cos(\theta) \quad , \tag{3.2}$$

where θ is the angle to the source plane normal and $F_{neutral}$ is the neutral particle flux towards the wafer, usually found through experiments or reactor-scale simulations taking into account the properties of the specific reactor. In general, the flux Γ_x for any particle of type x may also include angular distributions for the energy of the particles or the location on the source plane. For neutral species, the kinetic energy is usually too low to significantly impact interactions with the surface. As this work focusses on structures well below the micrometre regime, it is also reasonable to assume that neutral flux distributions do not change significantly across the length scale of the simulation domain. Hence, the flux of neutral particles through the source plane can be described appropriately without considering the energy distribution of particles or the location on the source plane.

In numerous fabrication processes, biased plasmas [127] are used to accelerate ions towards the wafer and achieve focused particle trajectories. Essentially, the distribution of the ions becomes more directional as the plasma sheath potential increases and ion movement is not only governed by the random collisions with other particles, but also by the electromagnetic field in the reactor. Hence the particle flux for ions has a more focused directional distribution which can be approximated by considering the ion flux F_{ion} and a power cosine [128]:

$$\Gamma_{ion}(\theta, \vec{x}_{\mathcal{P}}, E) = \frac{n+1}{2\pi} F_{ion}(\vec{x}_{\mathcal{P}}) \epsilon(E) \cos(\theta)^n \quad , \tag{3.3}$$

where n is the exponent of the cosine, $\epsilon(E)$ is the ion energy distribution giving the fraction of all ions with energy E and $\vec{x}_{\mathcal{P}}$ is the position on the source plane.

Depending on the type of plasma process to be simulated, different simplifying assumptions can be made. In a direct current (DC) plasma, if the thermal energy is small compared to the ion energy, all ions have the same energy:

$$E = eV \quad , \tag{3.4}$$

where e is the elementary charge. Therefore, the ion energy distribution follows a Dirac delta function

$$\epsilon(E) = \delta(E - eV) \tag{3.5}$$

and the exponent n can be approximated with [129]

$$n = \frac{2E}{k_B T} \quad . \tag{3.6}$$

In focused ion beam (FIB) plasmas, it can be assumed that all ions have the same energy and direction and only depend on $\vec{x}_{\mathcal{P}}$ as the ion flux is distributed according to a Gaussian [130] around the centre of the beam $\vec{x}_0$

$$\Gamma_{ion}(\vec{x}_{\mathcal{P}}) \propto \exp\left(||\vec{x}_{\mathcal{P}} - \vec{x}_0||_2^2\right) \delta(E - E_0) \quad , \tag{3.7}$$

where E_0 is the ion energy of the beam.

The source distribution in molecular-beam epitaxy (MBE) reactors is more complex, as a ring of point sources is used to generate the source flux [131]. It is also possible that these point sources emit different materials, further complicating the arrangement [132]. It is important to consider the exact setup of the reactor in order to find the spatial and directional source flux for the feature scale which forms a small part of the wafer, since the directional distribution of the incoming flux changes drastically across the wafer [133].

Radio frequency (RF) plasmas have complex energy, spatial and directional dependencies and therefore plasma sheath simulations should be used to find an accurate expression for Γ_{ion} [134] in these types of plasmas.

3.2.3 Feature Scale Transport

By definition, the size of the feature scale is much smaller than the reactor scale region, namely on the length scale of the manufactured features, i.e. smaller than 10 µm. The mean free path of particles $\bar{\lambda}$ defined in Eq. (3.1) is much larger than the size of the feature scale and thus the movement of particles is governed by ballistic transport, which can lead to shadowing due to the straight path of particles. Physically, each infinitesimally sized source plane element dA in Fig. 3.2 can be considered an individual particle source with properties governed by its source distributions Γ_x. Due to the relatively slow movement of the wafer surface compared to the molecules in the gas phase, the surface can be treated as constant during the modelling of the particle transport. Practically, this means that it can be assumed that there is a constant flux from the source plane throughout the calculation of incoming particle fluxes on the wafer surface.

3.2.3.1 Boundary Conditions

Since it would be computationally unfeasible, and often impossible, to simulate the entire wafer, only a small portion of interest, such as a single feature, device, or circuit, is simulated. In order to still properly encompass the influence of neighbouring sections of the wafer, appropriate boundary conditions must be used for the simulation. Usually, structures would be repeated across the wafer, e.g. static random access memory (SRAM) cells in the memory region of a microprocessor. Therefore, when a particle leaves the simulation domain and hits a boundary, it must be treated accordingly. Using reflective boundary conditions, as shown in Fig. 3.3, the particle would simply be reflected off the domain boundary and therefore stay within the simulation domain. Periodic boundary conditions refer to the particle re-entering the domain through the other side, thus leading to the same description, as if the simulation domain was repeated periodically outside of the domain boundary.

3.2.3.2 Modelling Approaches

The two main approaches to modelling particle transport are numerical modelling using ray tracing methods and analytical calculations which require certain assumptions about the process to be made. In analytical models, certain properties of the initial geometry can be used to find a solution for the particle flux on the surface.

One example for this approach is using Knudsen diffusion to model particle transport along straight trenches, which has been used successfully for specific processes [135, 136, 137]. However, deep knowledge of the modelled process is required and certain assumptions have to be made, so analytical models are not applicable to all processes and geometries. Furthermore, their general approaches may differ considerably, which is why they will not be discussed in detail here.

Ray tracing methods follow the ballistic trajectories of molecules in the feature scale region. However, due to the large number of molecules, not all can be simulated explicitly, but their trajectories are approximated using Monte Carlo (MC) methods. Each ray then represents potentially thousands of molecules, drastically reducing simulation runtime. Since the transport is simulated directly, the resulting distribution of molecules represents the expected physical result, given that the number of simulated particles is large enough. Since the transport simulation encompasses the trajectories of particles and the geometry directly, they are not limited to certain processes or initial geometries. Therefore, ray tracing is more appropriate for the modelling of general fabrication processes, where simplifying assumptions cannot be made easily.

The following sections will describe the numerical methods employed during ray tracing in detail.

3.2.3.3 Top-Down Flux Calculation

As suggested by its name, in the top down modelling approach, particles are represented using rays originating from each source plane element dA. In the simplest case, all particles are adsorbed on the surface on impact and do not reflect or re-emit. Thus, the propagation of each ray ends somewhere on the surface. After all rays have been traced, the number of rays terminating at each surface element is counted, which is equivalent to the particle surface flux $F(\vec{x}_s)$, as discussed in Fig. 3.1. Typical source plane fluxes for semiconductor manufacturing processes are on the order of $10^{16}\,\mathrm{cm}^{-2}\,\mathrm{s}^{-1}$ [110]. Therefore, in order to simulate a wafer area of $1\,\mu\mathrm{m}^2$ during one second of a fabrication process, roughly 10^8 rays must be traced. As most fabrication processes last substantially longer than one second, the large number of rays which would have to be traced makes it unfeasible to actually simulate each molecule individually. Hence, each ray usually represents thousands of molecules rather than just one. The particle surface flux $F(\vec{x}_s)$ is then simply the product of the number of rays terminating at the surface element $\vec{x}_s$ and the number of molecules per ray. Although representing multiple molecules with one ray leads to averaging of the surface fluxes, the result can be considered a good approximation as long as the total number of rays is much larger than the number of surface elements:

$$N_{\text{rays}} \gg N_{\text{surface elements}} \quad . \tag{3.8}$$

To achieve a desired accuracy in the final result, the number of rays originating from the source plane should thus be related to the number of surface elements the rays may terminate on:

$$N_{\text{rays}} \propto \frac{1}{\sigma_{\text{ray}}} N_{\text{surface elements}} \quad , \tag{3.9}$$

where σ_{ray} is the relative error compared to the analytical result for the surface flux, obtained by integrating over all source plane distributions Γ_x. Intuitively, this can be verified by considering that, in order to achieve a numerical error of $\sigma_{\text{ray}} \to 0$, an infinite amount of rays would have to be traced.

Given an expression for the source plane distribution $\Gamma_x(\theta, \vec{x}_p, E)$, as in Eq. (3.2) and Eq. (3.3), MC methods are employed to generate the initial properties of each ray [138]. The starting position of each ray on the source plane can either be generated pseudo-randomly or chosen from a rectilinear grid of points on the source plane with a regular grid spacing. The latter is computationally more efficient and leads to comparable results, as long as the grid spacing on the source plane is comparable to the resolution of the surface the particles are incident on. Once all the necessary properties of the ray are known, its path to the surface can be traced using algorithms which have been extensively applied in computer graphics [139, 140, 141, 142]. The general idea of these ray tracing methods are shown in Fig. 3.3 for two particle source distributions, as they would be used in the simulation of a semiconductor fabrication process. The figure also highlights boundary conditions of the simulation domain, as well as possible reflections of molecules if they are not fully adsorbed by the surface.

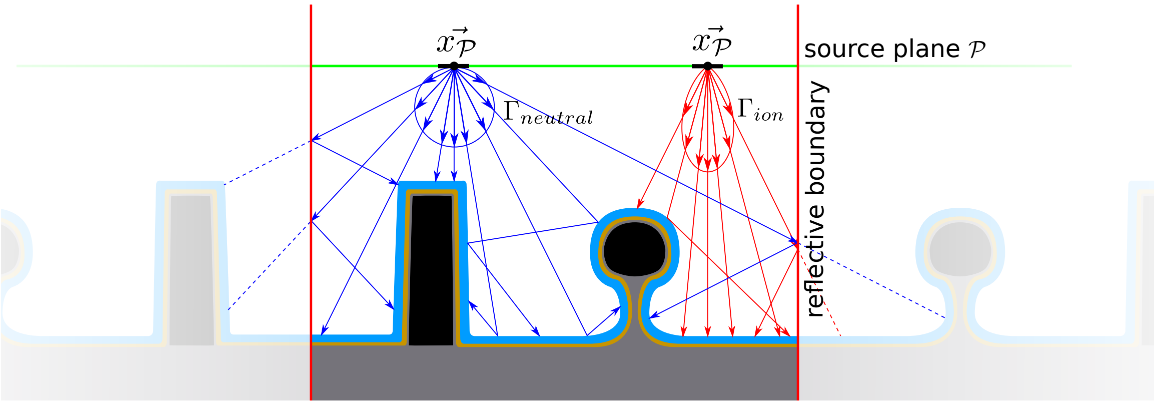


Figure 3.3: Schematic depiction of rays being traced from the source plane to the surface using reflective boundary conditions. Each ray describes either neutral molecules (blue) or ions (red), governed by the source distributions $\Gamma_{neutral}$ and Γ_{ion}, respectively. Diffuse and specular reflections are shown for neutral species and ions, respectively.

The probability with which the particle will stick to the surface is referred to as the sticking probability S and may vary for each combination of molecule and material it is incident on. Since each ray represents numerous molecules, S can be used as a probabilistic factor rather than simply deciding whether the ray should be reflected or not. This means that the ray is always traced further as if the reflection occurred, with the flux carried by that particle reduced by a factor of $1 - S$. When the flux reaches a pre-defined low threshold, the tracing of the particular ray is stopped. How particles are reflected or re-emitted from the surface depends on the specific chemistry and is governed by a reflection distribution Γ_{refl} describing the angle, flux and energy of the reflected ray, analogous to the source distribution Γ_x.

Several algorithms have been proposed in order to perform ray tracing directly on implicit surfaces [143, 144, 145]. However, many of these algorithms stem from the field of computer graphics and are tailored towards visualisation applications. Hence, they are not necessarily applicable to the simulation of molecular transport. Furthermore, performing ray-surface intersection tests on implicit surfaces is computationally more expensive than on explicit surfaces [146], which is why molecular transport simulations are usually performed using explicit surface representations, such as triangulated surfaces [147]. However, the generation of a triangulated surface from a level set is expensive and thus undesirable. Hence, a different type of explicit surface should be used for the simulation of molecular transport using ray tracing.

In order for ray-tracing methods to work robustly, the explicit surface must be closed. There is no requirement for the mesh, however, to not contain self-intersections or other usually undesirable inconsistencies. Therefore, the explicit surface can be approximated using spheres of radius Δg centred at each grid point [148], or more accurately by discs forming tangential planes to the surface [60], as shown in Fig. 3.4. Spheres or discs can be extracted highly efficiently while still providing a robust, closed explicit interface for fast intersection tests of commonly used ray tracing algorithms. Since each sphere or disc is associated with a grid point, the results of the ray tracing step can easily be related back to the section of the surface described by the specific grid point.

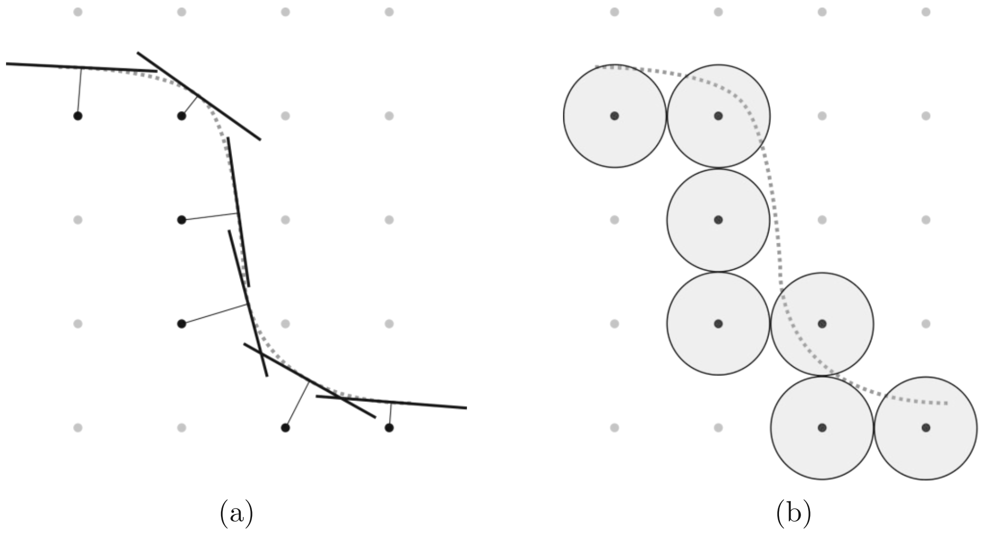


Figure 3.4: Two ways to approximate an implicit surface efficiently by explicit shapes on active grid points (black), in order to simplify intersection tests for ray tracing. (a) Tangential line segments used to form an explicit approximation of the surface, as described in [60]; (b) surface approximated by explicit circles centred at active grid points, as described in [148]. The line segments and circles are replaced by discs and spheres in three dimensions.

In the top-down ray tracing approach, the physical behaviour of molecules moving through the feature scale is simulated and thus, all physical properties may be included in the simulation. Hence, it is also possible to include phenomena such as charged molecules in the feature scale being repelled by surface charges [149]. Furthermore, even complex reflective properties can be modelled using the top-down approach straight-forwardly. Specular reflections, for example, can be realised by considering the incoming angle of each ray with respect to the surface, which is found easily from the intersection test. Diffuse reflections employing any type of reflection distribution may also be used straight-forwardly, as discussed above. Crystal-direction dependent effects on the reflection or re-emission of molecules can also be included straight-forwardly by including the surface orientation at the intersection point [150]. Additionally, even particle-particle collisions can be modelled if the simulation domain is too large for the ballistic transport assumption to hold, such as high aspect ratio geometries like deep vias or trenches.

However, the inclusion of additional physical properties in the description of a process may result in a strong increase in the required computational effort. Although the top-down approach is often more computationally intensive than other forms of modelling molecular transport in the feature scale region, it is very closely linked to the physical behaviour inside the reaction chamber. Therefore, it is the most commonly encountered method in physical process simulations.

3.2.3.4 Bottom-Up Flux Calculation

The bottom-up approach to calculating the surface flux can be considered the reverse of top-down flux calculation, described in the previous section. Instead of tracing rays along the physical paths of the molecules, an element on the surface is considered and the paths of molecules are traced back to their origin. This is achieved by considering how much of the source plane is visible from a point $\vec{x}_s$ [151, 152], as shown in Fig. 3.5. Numerically, the visible parts of the source plane $\mathcal{P}$ are found by considering one surface element $\vec{x}_s$ and iterating over all discretised points on the source plane. If the distribution is visible from $\vec{x}_s$, its contribution towards the surface element is added to the total flux, according to each source flux distribution Γ_{src}.

The sum of all visible point fluxes results in the total molecular flux incident on the surface element at $\vec{x}_s$

$$F_{\mathrm{direct}}(\vec{x}_s) = \sum_{\vec{x}_{\mathcal{P}}} \Gamma_{src}(\vec{x}_s, \vec{x}_{\mathcal{P}}) Y(\vec{x}_s, \vec{x}_{\mathcal{P}}) \quad , \tag{3.10}$$

with the total flux being referred to as direct flux, since it does not include any reflections from other parts of the surface. The visibility function $Y(\vec{x}_s, \vec{x}_{\mathcal{P}})$ is unity if the point $\vec{x}_{\mathcal{P}}$ is visible from $\vec{x}_s$ and zero otherwise. Since the exact flux emitted from Γ_{src} depends on the angle of emission, the distribution depends on the surface point and source plane point. This is indicated by the black arrow in Fig. 3.5. The angles at which source plane points are visible from $\vec{x}_s$ are indicated by the green arc. All distributions on the source plane within the cone indicated by the dashed grey line thus contribute to the direct particle flux incident on the considered surface element.

In order to identify which points are visible and thus to find a description for Y, adaptive visibility sampling can be employed. In this method, rays are launched

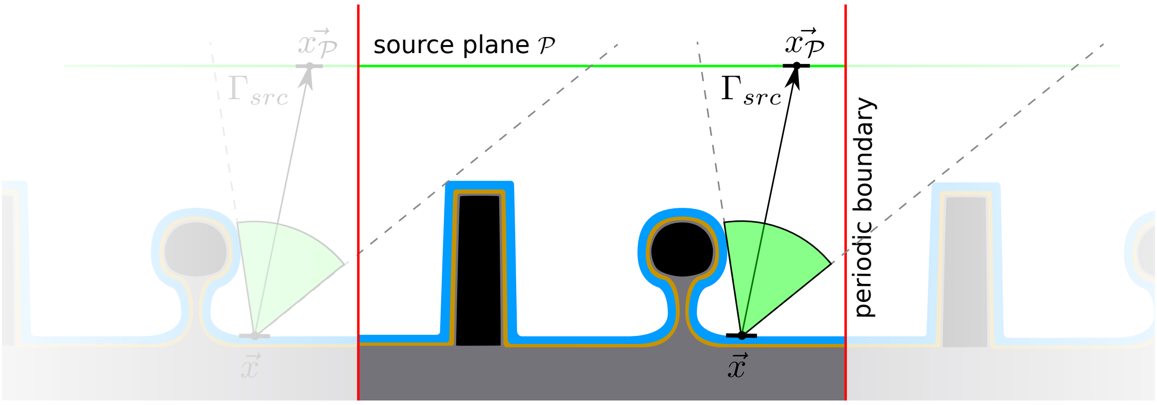


Figure 3.5: Schematic representation of the bottom-up flux calculation for modern transistor gate structures, using periodic boundary conditions, meaning the entire simulation domain is repeated at the boundaries. The black arrow indicates the direction used to find the direct flux incident on $\vec{x}$ from a single particle source at $\vec{x_{\mathcal{P}}}$ with a source distribution of Γ_{src}. The flux of all visible source plane elements, indicated by the green arc, is summed to give the total direct flux on $\vec{x}$.

towards the source plane in spatial directions spaced by pre-defined angles. If a ray hits the source plane, this part of $\mathcal{P}$ is visible to the surface point. If the ray hits another part of the surface first, the source plane is not visible from $\vec{x}_x$ in that specific direction.

The achieved resolution is given by the angle between two consecutive rays, since a transition from a visible to a shadowed section can only be resolved at this angle. The resolution can be increased adaptively, by launching additional rays in the direction of the observed transition [147]. Once a specific angular resolution has been achieved, all source distributions can be summed and the final flux can be calculated. If only a fixed resolution is required, numerous rays can be launched simultaneously, which can be performed efficiently on graphics processing units (GPUs) [153]. Therefore, sampling visibilities can be much more efficient than actually simulating the physical paths of molecules as in the top-down approach.

However, modelling reflections using the bottom-up method requires additional considerations, as they cannot be modelled directly from the source distributions. In order to find the reflected or re-emitted fluxes, an iterative method must be applied, which first solves for the direct flux F_{direct} which is then used to find the reflected and re-emitted fluxes. Each surface element is then considered as a particle source with distributions Γ_{refl} and Γ_{reem}, as shown in Fig. 3.6. The source distributions include all the reflection properties based on the material and the type of the described molecule, such as energy, sticking probability, or angular dependence. Similarly to the direct flux defined in Eq. (3.10), the reflected and re-emitted fluxes incident on a surface element at $\vec{x}$ are found by summing all contributions from other surface elements at $\vec{x}'$.

For simplicity, the fluxes in the iterative method are named according to the current iteration, meaning the direct flux is labelled F_0, the flux resulting from the

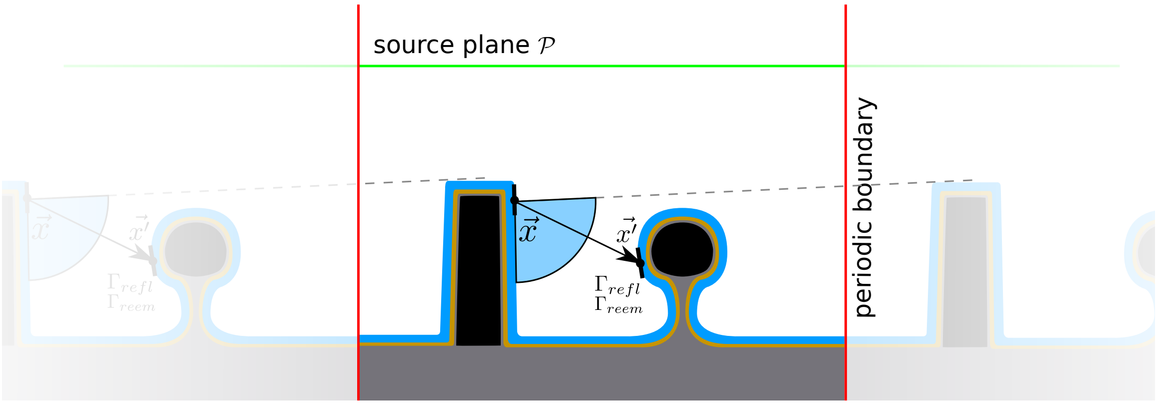


Figure 3.6: Calculation of reflected or re-emitted fluxes using a bottom up technique with periodic simulation boundaries. The black arrow indicates the direction used to find the reflected and re-emitted flux incident on $\vec{x}$ from a particle source at $\vec{x}'$. The source distributions Γ_{refl} and Γ_{reem} define the flux emitted towards $\vec{x}$. The total indirect flux at point $\vec{x}$ is found by summing the flux from all visible surface points, highlighted by the blue arc.

first iteration of reflection and re-emission is labelled F_1 and so forth. The i-th iteration of the secondary fluxes is given by

$$\begin{aligned} F_{i+1}(\vec{x}) = F_i(\vec{x}) - F_{\text{i,refl}}(\vec{x}) - F_{\text{i,reem}}(\vec{x}) \\ + \sum_{\vec{x}'} Y(\vec{x}, \vec{x}') \left[\Gamma_{\text{i,refl}}(\vec{x}, \vec{x}', F_i) + \Gamma_{\text{i,reem}}(\vec{x}, \vec{x}', F_i)\right] \quad , \end{aligned} \tag{3.11}$$

where $F_{\text{i,refl}}(\vec{x})$ and $F_{\text{i,reem}}(\vec{x})$ are the total outgoing fluxes from the surface element $\vec{x}$. The arriving flux at the considered surface element $\vec{x}$ is thus reduced by the total reflected or re-emitted flux and is increased by the arriving fluxes from other parts of the surface. This step is repeated until the desired accuracy has been reached or all particles have been absorbed, meaning that $F_{\text{i,refl}}(\vec{x}), F_{\text{i,reem}}(\vec{x}) \to 0$ for all $\vec{x}$. The number of iterations necessary to arrive at appropriate results strongly depends on $\Gamma_{\text{i,refl}}$ and $\Gamma_{\text{i,reem}}$. If they do result in a large portion of the arriving flux leaving the surface again, meaning that the sticking probability is low, more iterations will be necessary as more reflections or re-emission take place.

Since the flux at the surface element $\vec{x}$ is preserved after each step of the iteration rather than individual particles and their properties, modelling specular reflections is not possible straight-forwardly. This type of reflections can only be approximated by assigning an average incoming and outgoing direction [154]. This leads to shortcomings when modelling ion-enhanced etching processes, which are heavily influenced by energetic ions and their reflections off the surface. Nevertheless, for processes with little influence from ions, the bottom-up approach can be used as a more efficient alternative to the top-down method.

3.2.4 Molecular Interactions with the Surface

The modelling of feature scale transport, discussed in the previous section, results in the incoming flux values on the surface for all molecular species involved in the fabrication process. The next step is to model how the particles which impinge on the surface interact with the substrate material to be adsorbed on the surface and thus influence the coverage of different molecular species on it [155]. If a material is covered by a species, it means that the specific molecules are adsorbed on the surface long enough to engage in chemical reactions leading to a topographical change on the surface. This is in contrast to reflected particles, which leave the surface immediately and are not able react with the substrate chemically.

The most general type of process, which encompasses physical and chemical surface interactions, is the ion-enhanced etching of a substrate. This type of process is used here as an example for the procedure of modelling interactions with the surface, as it may involve many different types of molecules and simultaneous deposition and etching. For simplicity, the particles which impact the surface may be grouped together according to the chemical effect they have on the substrate [156]. This results in four abstracted types of particles affecting the surface topography:

1. Etchant: Leads to the removal of substrate material

2. Passivation Species: Deposits on the surface to form a passivating layer

3. Passivation Etchant: Removes passivating layer and possibly substrate

4. Energetic Ions: Physically impacts the surface delivering kinetic energy

In order to appropriately model the process, the incoming flux of etchant F_e, passivation F_p, passivation etchant F_{pe} and ions F_i must be known to find the corresponding coverages θ_e, θ_p, θ_{pe} [157].

The coverages θ_e and θ_p are normalised such that for each point on the surface, their sum cannot be greater than unity, while θ_{pe} is defined as a fraction of θ_p. The coverages can therefore also be interpreted as the probability of the specific type of particle being present on the considered surface site. There is no coverage of ions, since ions do not deposit on the surface, but rather deliver kinetic energy for immediate physical reactions on the surface.

Since the transport of molecules is fast compared to the surface movement, it is assumed here that the coverages on the surface will reach a steady state immediately after the fabrication process is started [158, 159]. Based on this assumption, a linear equation can be set up for each particle type's coverage [160], describing all relevant factors which contribute to the coverage.

<u>Etchant Coverage</u>

The etchant coverage is expressed as:

$$\frac{d\theta_e}{dt} = F_e S_e (1 - \theta_e - \theta_p) - k_{ie} F_i Y_{ie} \theta_e - k_{ev} F_{ev} \theta_e \approx 0 \quad , \tag{3.12}$$

where S_e is the sticking probability of etchant on the surface. k_{ie} and k_{ev} are the stoichiometric factors for ion-enhanced etching an chemical etching respectively, describing how much of one material, compared to its reactant, is needed in the chemical reaction. Y_{ie} is the ion-enhanced etching yield, which has a square root dependence on ion energy and incoming angle to the surface. It is usually modelled as [161]

$$Y_{ie} = \left(\sqrt{E} - \sqrt{E_{th}}\right) cos(\omega) \quad , \tag{3.13}$$

where E_{th} is the threshold energy for ion-enhanced etching to occur and ω is the angle between the incoming ion and the surface normal.

F_{ev} is the evaporation flux describing how much of the chemically etched material is transported from the surface through desorption and may be expressed as:

$$F_{ev} = K_s F_e e^{-E_s/k_B T} \quad , \tag{3.14}$$

where K_s is a dimensionless reaction rate constant which defines how likely a reaction is to take place once the etchant comes into contact with the substrate and E_s is the binding energy between molecules of the substrate.

Passivation Coverage

The passivation coverage is given by:

$$\frac{d\theta_p}{dt} = F_p S_p - F_i Y_p \theta_p \theta_{pe} - F_{depo} \approx 0 \quad , \tag{3.15}$$

where S_p is the sticking probability of passivating molecules on the substrate and Y_p is the etching yield for ion-enhanced polymer etching. F_{depo} is the flux describing at which rate molecules are incorporated into the substrate during deposition in order to advance the surface.

Passivation Etchant Coverage

The coverage of passivation etchant is written as:

$$\frac{d\theta_{pe}}{dt} = F_{pe} S_{pe}(1 - \theta_{pe}) - F_i Y_p \theta_{pe} \approx 0 \quad , \tag{3.16}$$

where S_{pe} is the sticking probability of this type of particle on the passivation.

Physical considerations

The first terms in Eqs. (3.12), (3.15) and (3.16) describe the main mechanism with which the coverage is increased, namely through particle transport in the gas phase. Since only the molecules which actually stick to the surface without being reflected can engage in surface reactions, this term is also dependent on the respective sticking probabilities. The subsequent negative terms describe loss mechanisms which for etching occur only through the removal of the substrate through energetic ions or evaporation. In the case of deposition, the main loss mechanisms are ion-enhanced etching and the incorporation of molecules into the material through deposition.

Several additional phenomena which occur in specific processes could also be added. If surface diffusion of reactants play a role, a term describing this flux can be included. This flux is then simply obtained by considering the fluxes at close surface points and allowing for diffusive flow to the currently considered site.

In the simplest case, the particle fluxes do not depend on the coverages on the surface. In this case, ray tracing can be performed to find the fluxes which, in turn, are used to calculate the coverages for each particle type. However, in several processes, the reflective properties of particles may change drastically with changing surface coverages. This includes physical sputtering of the substrate through energetic ions, which is strongly dependent on etchant coverage and may be modelled as a type of re-emission. Sputtering, therefore, has to be considered during the modelling of molecular transport.

In order to resolve this cyclic dependency, an initialisation step is necessary for the coverages to settle into a steady state, as required for Eqs. (3.12), (3.15) and (3.16) to be valid. Once the coverages have converged to a value, the surface velocities can be calculated and the level set advanced. The coverages at the surface locations before the advection serve as a good approximation for the next particle transport modelling step [162], since the level set cannot be moved by large distances due to the CFL condition discussed in Section 2.4.2.2. Hence, this initialisation step is usually only necessary at the very beginnning of a simulation. However, it is crucial to perform this step for each different physical surface rate model applied.

3.2.5 Chemical Reactions of Molecules on the Surface

Once the fluxes and coverages on each surface point are known, they can be used to identify how the surface will evolve over time. In order to identify this evolution, the main processes which lead to a change in the surface must be identified. In the example of an ion-enhanced etching process, discussed in the previous section, there are several types of etching and deposition mechanisms which lead to topographical changes:

1. Deposition of passivation
2. Ion-enhanced etching of passivation
3. Chemical etching of the substrate
4. Ion-enhanced etching of the substrate
5. Physical sputtering of the substrate

These reactions are depicted in Fig. 3.7, highlighting how the different fluxes and coverages discussed in the previous sections contribute to the topographical changes.

Passivation

The first two mechanisms only affect the passivation layer deposited in regions of low ion flux. This passivating layer therefore protects vertical features of the surface

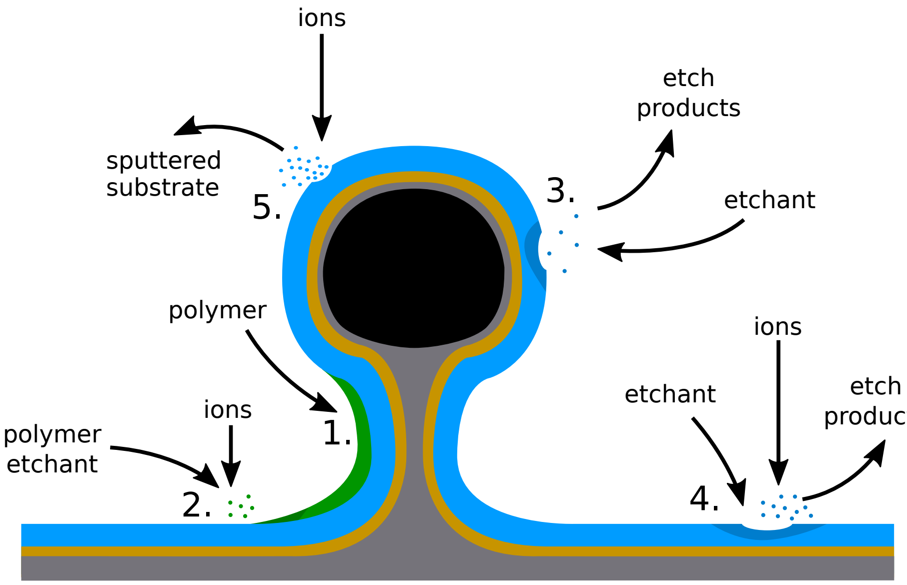


Figure 3.7: The five mechanisms leading to a topography change of the surface in the ion-enhanced etch process example. 1. Passivating molecules form polymers on sidewalls (green), which may be removed again through 2. ion-enhanced etching. 3. During chemical etching the substrate reacts with an etchant forming volatile etch products which thermally desorb from the surface through evaporation. 4. Ion-enhanced etching increases the chemical etch rate by breaking up bonds in the substrate, enhancing the formation of volatile etch products. 5. Energetic ions collide with the substrate and break the surface bonds to remove material physically without a chemical etchant through ion sputtering.

from being etched chemically. Eq. (3.15) can be rearranged to give the deposition flux used to advance the surface, which is simply the number of particles available for deposition on the substrate. The deposition rate of the passivation at a point $\vec{x}_s$ on the surface is therefore given by

$$v_{depo}(\vec{x}_s) = -\frac{F_{depo}}{\rho_p} = \frac{1}{\rho_p}\left(F_p S_p - F_i Y_p \theta_{pe}\right) \quad , \tag{3.17}$$

where ρ_p is the number density of the passivation material, meaning the number of particles per volume. The minus sign for the deposition rate stems from the convention of negative LS values denoting the inside of a material. The passivation coverage θ_p is not included in the second term as deposition means that the surface is covered in passivating particles and therefore $\theta_p = 1$. If there is a large flux of passivating particles at the surface, a passivation layer will be deposited, while a large ion flux results in the suppression of growth. If there is a previously deposited passivation layer beneath, the negative value means that this layer is etched. If the surface point $\vec{x}_s$ is on the original substrate rather than on the passivation layer, then v_{depo} must not be negative, as passivation etchant cannot remove the substrate. In this case, v_{depo} has no physical meaning and can be ignored.

Substrate

The substrate can be modified through all of the above mechanisms with the exception of the ion-enhanced etching of the passivation layer. If deposition dominates and $v_{depo} > 0$, no further considerations are necessary and v_{depo} can simply be applied at the specific point on the surface. If, however, $v_{depo} < 0$ then etching of the substrate takes place. Each of the three etch mechanisms is expressed by one term in the final surface velocity:

$$v(\vec{x}) = \frac{1}{\rho_{sub}} \left[\underbrace{F_{ev}\theta_e}_{\text{chemical etching}} + \underbrace{F_i Y_{ie}\theta_e}_{\text{ion-enhanced etching}} + \underbrace{F_i Y_s (1-\theta_e)}_{\text{ion sputtering}} \right] , \qquad (3.18)$$

where ρ_{sub} is the number density of the substrate. Since all coverages affect θ_e, it must be found by solving Eqs. (3.12), (3.15) and (3.16). Chemical and ion-enhanced etching consume etchant and therefore result in a decrease of etchant coverage, while ion sputtering removes the substrate without affecting any of the coverages.

Encapsulating a large number of physical etch or deposition mechanisms, a single model could, in theory, describe a wide array of different fabrication processes. The choice of stoichiometric factors, threshold energies and other coefficients for each mechanism is crucial in order to properly capture the properties of a fabrication process. Since these physical constants entail all differentiating properties between two different processes, they must be considered with great rigour. Usually, these constants are found from experiments, reactor-scale or ab-initio simulations, or a combination of both. By adding more terms to the above model, any physical effect affecting surface movement could be included, although simple models, such as the one described here, are frequently enough to describe even complex processes [110]. It is therefore important to identify dominant etch or deposition mechanisms in order to ensure that the main properties of the fabrication process are modelled adequately.

3.3 Summary

Methods for finding the velocity field which describes the movement of a level set surface were discussed. Empirical modelling, or emulation, can be applied when the geometric effect of a process on the initial surface is known in advance and can simply be applied. Therefore, these types of processes can be modelled highly efficiently using the geometric advection algorithm presented in Section 2.4.3.

Chemical modelling is applied when intricate physical effects need to be included in the description. This includes the modelling of molecular transport through the feature scale region, which can be performed in a variety of ways. The most flexible and appropriate method for modelling complex fabrication processes was found to be top-down flux calculation employing Monte Carlo ray tracing. A highly efficient explicit data structure was presented in which discs tangential to the surface can be extracted directly from the level set. This allows for molecule-surface interactions to be recorded for all active grid points.

The modelling approach for molecular interactions with the surface, as used in the developed simulator, was presented. Using abstracted particle types for groups of molecules, this approach allows for a maximum of flexibility in the implementation of entirely different physical processes.

Finally, the modelling of chemical reactions on the surface, leading to the evolution of material interfaces, is discussed using the example of a modern ion-enhanced plasma etch process. For the appropriate modelling of this process, it is crucial to identify dominant physical mechanisms leading to material evolution. Therefore, if important physical behaviours are not encompassed by the model, process specific features cannot be generated, while the inclusion of unnecessary mechanisms will lead to computationally inefficient simulations of processes with only minor effects on the final structure.

Chapter 4

Software Implementation

This chapter describes the specific software implementation of the numerical methods presented in Chapter 2, implemented within the scope of this work. The C++ code for all numerical algorithms was implemented in smaller self-consistent libraries depending on the specific functionality of the respective implementation. Five individual C++ libraries were created, incorporating all functionality required for a process simulator with their dependency hierarchy shown in Fig. 4.1:

- ViennaHRLE [163]: This library provides the underlying sparse data storage for the level set and cell set representations. The data type stored is not limited to numeric types, so the data container can also be used for the storage of volume data.

- ViennaLS [164]: All level set specific algorithms, including conversions to other surface descriptions, advection algorithms, and geometric properties are part of this library which uses the data structure provided by ViennaHRLE to store the level set values.

- ViennaCS: This library contains the cell-based data structure described in Section 2.3.3.2. It uses the data structure of ViennaHRLE to store custom volumetric properties of the simulated materials.

- ViennaRay [165]: A high-performance ray tracing library which is used to model molecular transport in the feature scale region, supporting custom particle types in order to model different sources, reflection types, and particle properties.

- ViennaPS [166]: This library ties together all other software implementations by providing a modelling framework which allows for the straight-forward creation of process models. It provides the necessary interfaces between all other libraries and makes them accessible in a simple fashion.

All the software libraries depicted in Fig. 4.1 are discussed in detail in this chapter starting with ViennaHRLE and then moving on to the dependent libraries. Finally, the interplay of all these software elements within ViennaPS will be discussed by considering the program flow of a typical microelectronic device fabrication simulation, highlighting important interfaces between the different libraries.

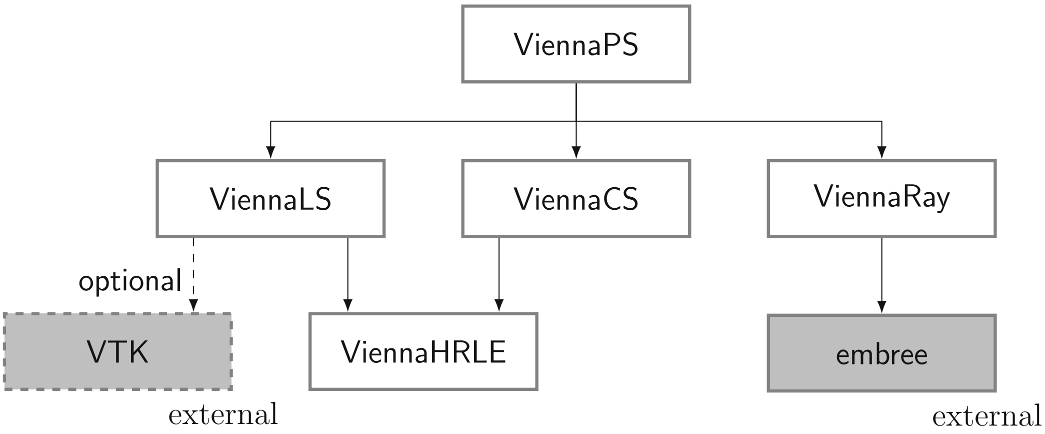


Figure 4.1: Dependencies of the ViennaPS process simulation library on other software libraries for core functionalities. Additional features, which do not provide essential utilities but rather convenient additional functionality are connected using dashed lines. External libraries which were not developed in the course of this work are filled with a grey background and are labelled as external.

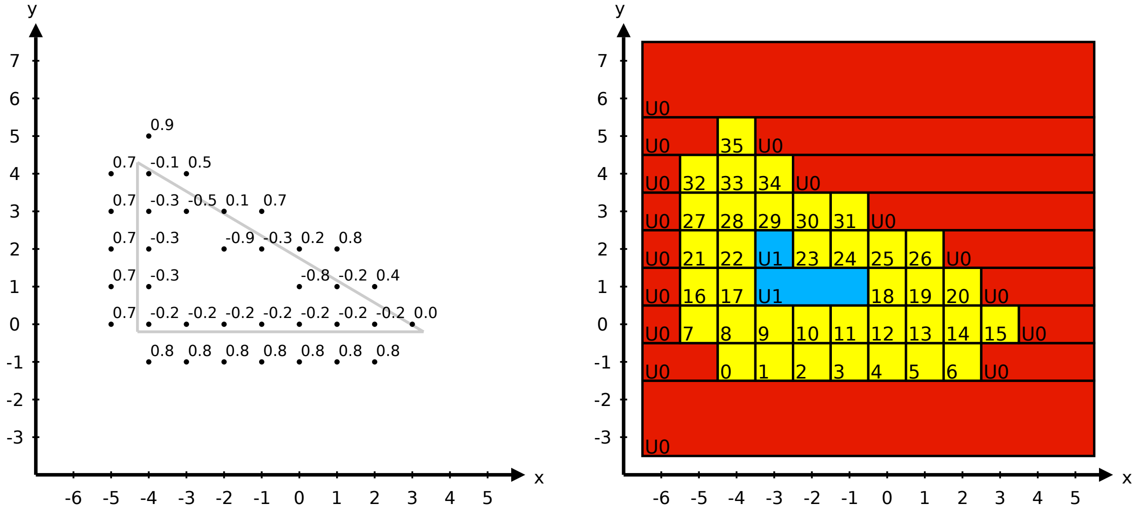
4.1 ViennaHRLE - Sparse Data Container

ViennaHRLE provides a storage container for sparse spatial data, as required for the description of a sparse field level set (LS). Due to the specific requirements of level set algorithms, a specialised data structure is needed for storing the LS values. Algorithms operating on the LS do not require random access, but rather fast sequential access as they operate on the entire surface in one pass. Furthermore, all values are defined on a rectilinear grid. However, not all points on the grid are required for the surface description but only a small set of grid points. A data structure which is optimised for this type of data access is the hierarchical run-length encoded (HRLE) data structure [167].

4.1.1 Data Structure

In this data container, the grid is segmented in each direction and run-length encoded. This means that certain neighbouring points can be described as one entity if they have similar values. Such a collection of points in an HRLE structure is referred to as a run. In a sparse field LS stored in this structure, grid points with a signed distance value larger than 0.5 can be represented by a large negative or large positive number, depending on whether their location is inside or outside of the material volume, respectively. Since their exact LS value is not known, these grid points, or runs, are referred to as undefined points or undefined runs, respectively.

The way this data structure is constructed is illustrated in Fig. 4.2. The simple triangle in Fig. 4.2a is represented using a sparse field LS. In order to store the LS values efficiently, the grid is segmented into defined and undefined runs, as shown in Fig. 4.2b. Blue areas in the segmentation represent negative undefined runs, in which all points have no oppositely signed neighbour. Therefore, these points do not contribute additional information to the surface description and thus their exact value is not important. However, the sign of these points may still be required in simulations and can be found straight-forwardly by using the sign of the undefined value referenced by the run. The same applies to the red areas in Fig. 4.2b, which refer to positive undefined runs and therefore incorporate all grid points with positive values and no oppositely signed neighbours. Once the grid containing the LS is segmented, it can be run-length encoded and stored in the HRLE structure.



(a) Triangular surface represented by a sparse field LS.

(b) Segmentation of the sparse point data of the LS in the HRLE data structure.

Figure 4.2: Surface of a triangle stored in the HRLE data structure, showing how the LS values are segmented into defined (yellow), undefined negative (blue), or undefined positive (red) points.

In order to generate the data structure shown in Fig. 4.3, the segmented grid has to be projected in each dimension. In the case of the segmentation shown in Fig. 4.2b, the grid is first projected into the y direction. This is done by considering one index on the y-axis and marking it as a defined run if there is a defined run for any x-value in this grid line. Taking Fig. 4.2b as an example, the grid line at $y = -2$ corresponds to an undefined positive run in the y-direction, as there are no defined runs at $y = -2$. At $y = -1$ however, there are defined runs in this grid line, meaning $y = -1$ will be a defined run. For the structure shown in this figure, the projection onto the y-axis thus results in defined runs from $y = -1$ to $y = 5$ and positive undefined runs everywhere else. This projection is then run-length encoded to give three separate runs: One undefined positive run from $y = -3$ to $y = -2$, one defined run from $y = -1$ to $y = 5$ and another undefined positive run from $y = 6$ to $y = 7$. This gives the set of run types for the y dimension shown in Fig. 4.3. The defined run is given an identifier (ID) denoting which value is the first of this run. Since there is still a dimension below the current one, this refers to the start index for the lower dimension at which this run starts. The first defined run therefore always has the ID 0.

In Fig. 4.3, directly below the run types, the corresponding run breaks are shown. These values store the coordinate at which the next run starts, meaning the defined run starts at $y = -1$ and the last undefined run starts at $y = 6$, as discussed above. The beginning of the first run is defined as the lowest grid extent and the end of the last run is the upper grid extent, meaning that these runs extend to the edges of the simulation domain.

The start index, shown above the run types, defines which run is the first in the next grid line. As the entire structure was projected onto the y direction first, there is only one grid line for this dimension and thus only one start index, namely 0. Since all indices start at 0 by convention, the first start index must always be 0.

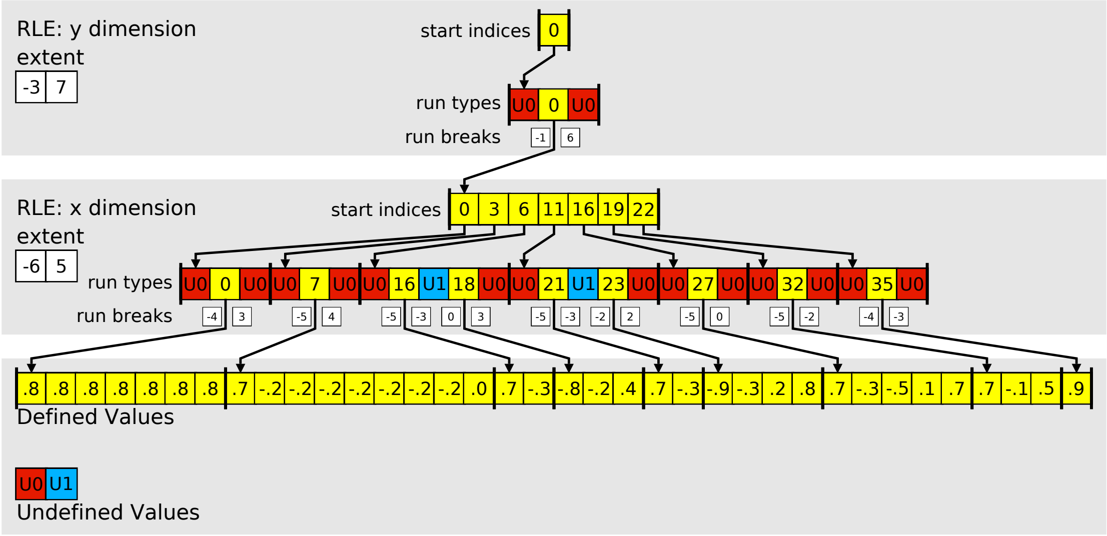


Figure 4.3: Internal representation of the HRLE data structure when storing a sparse field LS. Each dimension is run-length encoded separately and referenced into the next higher dimension using start indices. The defined values saved in the structure are stored sequentially in memory, allowing for fast sequential access.

Now the next lower dimension x is considered and the structure is filled accordingly. Since the only defined run in the y dimension extends over several grid lines, the first run of each grid line must be saved in the start indices. The first start index thus refers to the first run in the first grid line, which must always have the run type 0. The n$^{\text{th}}$ start index then refers to the first run in the n$^{\text{th}}$ grid line. The run types are then filled as before, but since x is the lowest dimension, the grid is not projected but run-length encoded as it is.

For example, the grid line at $y = -1$ in Fig. 4.2b is encoded first, resulting in three separate runs: Undefined positive from the beginning of the domain to $x = -5$, one defined run from $x = -4$ to $x = 2$ and one undefined positive run from $x = 3$ to the edge of the domain. There are seven defined runs in this grid line with the run type denoting the first of these. Since each defined run has its own run type ID, they are described by the IDs from 0 to 6. Hence, the next defined run will start with the run type ID of 7. In the lowest dimension, the run type IDs are simply the indices into the array which stores the defined LS value for each grid point. Therefore, all defined points are close to each other in storage, allowing for fast memory access. Especially during iterative advection, where the location of the LS values is not needed, the scalar velocity field can be applied to the array directly, allowing for highly efficient advection. There are three separate runs in the grid line at $y = -1$, which means that the next grid line must start with the 4^{th} run, corresponding to a start index of 3. This is repeated for each grid line corresponding to a defined run in the y dimension, resulting in a full description of the structure.

In order to describe a LS, only two undefined values are required: negative and positive. However, for the description of volume information, it is useful to define several additional background values. Hence, the HRLE structure allows for any number of undefined values. These are values, which are shared by several grid points and are therefore perfectly suited to be used for the storage of background values.

The implementation of the HRLE data structure in ViennaHRLE also includes the possibility for straight-forward parallelisation by automatically splitting the data structure according to the number of central processing unit (CPU) threads available for computation. This is performed by marking specific segmentation run types which signify the start and end of a parallel segment. The splitting is achieved through the dynamically balanced approach presented in [168], so that the optimal parallelisation can be achieved straight-forwardly.

4.1.2 Initialisation

First, the simulation domain is defined by setting its extent and boundary conditions. The most efficient way to fill this domain with the LS values describing a surface is to use a helper function provided by the ViennaHRLE library.

Listing 4.1: Helper function to fill an HRLE structure with LS values from a list of point-value pairs.

```
template <class T, int D>
void hrleFillDomainWithSignedDistance(
  hrleDomain<T, D> &newDomain,
  std::vector<std::pair<hrleVectorType<int, D>, T>> pointData,
  const T &negValue,
  const T &posValue,
  const bool sortPointList = true)
```

This function takes a list of points with their respective LS values, sorts them in lexicographical order and then inserts them into the HRLE structure. The sorting is necessary to reach the optimal performance by inserting values only at the end of all the arrays used to store the internal HRLE representation. Internally, the helper function shown in Listing 4.1 uses the two methods provided by `hrleDomain` to insert the points in the most efficient manner.

Listing 4.2: Functions to insert LS values directly into the HRLE structure.

```
template <class V>
void insertNextDefinedPoint(
  const V &point,
  hrleValueType distance)

template <class V>
void insertNextUndefinedPoint(
  const V &startPoint,
  const V &endPoint,
  hrleValueType value)
```

The first of the two functions provided in Listing 4.2 is used to insert a defined run into the structure by copying the value `distance` into the defined values array and building the required start indices and run breaks. The second function is used to insert undefined runs by specifying their start and end coordinates, as well as the value they represent. If this value is already stored somewhere in the undefined values array, there is no need to store it a second time, so the undefined run type of the first value will simply be used to represent this undefined run.

4.1.3 Data Access

Due to the complexity of the structure, random access is not efficient, as a value has to be found by traversing through each dimension and identifying the correct run by the run breaks. Therefore, finding one value takes on average $\mathcal{O}(\log N)$, where N is the number of LS values stored. Traversing the entire structure would thus require $\mathcal{O}(N \log N)$ operations. However, since many algorithms used in level sets can be performed by iterating over the entire structure once, efficient sequential data access can be used. Since all defined points are simply elements in an array, sequential access to each element is constant in time and the entire structure can be traversed in $\mathcal{O}(N)$ [169]. For ease of use, iterators are implemented in the library, which allow the user to access the current coordinate, the LS value, and several additional container-specific properties such as the current run. The ViennaHRLE library provides several types of iterators for accessing the values stored at the grid points:

1. hrleSparseIterator: Stops once at every defined and undefined run.

2. hrleDenseIterator: Stops once at every grid point.

3. hrleSparseMultiIterator: Iterates over several domains at the same time and stops at every run. If there are two defined runs at the same coordinate in multiple domains, it only stops once.

The last of the above iterators is useful for combining several `hrleDomain` structures into one according to a specific metric.

In order to find numerical derivatives, often required for topography simulations in a LS framework, fast neighbour access is required. However, finding the neighbouring grid points of any point requires random access and is therefore inefficient. A more efficient approach is to use several sequential iterators, each separated by one grid point and advancing them collectively. Therefore, the entire structure can be traversed in optimal time. ViennaHRLE provides fast neighbour access through specialised iterators. The neighbours to which these iterators provide access is given in terms of the spatial indices $i, j, k \in [x, y, z], i \neq j \neq k$ for a point at index (x, y, z) in the following. The order n of each iterator describes the distance of the farthest neighbour which can be accessed.

1. hrleSparseStarIterator: Access to all neighbours within $(i \pm n, j, k)$
2. hrleCartesianPlaneIterator: Access to all neighbours within $(i \pm n, j \pm n, k)$
3. hrleSparseBoxIterator: Access to all neighbours within $(i \pm n, j \pm n, k \pm n)$
4. hrleSparseCellIterator: Access to all neighbours within $(i + 1, j + 1, k + 1)$

Since access to each neighbour requires its own additional iterator, it is important to choose the most efficient one for the specific application. If only neighbours in grid directions are required, the hrleSparseStarIterator is sufficient, while grid diagonal neighbour access requires the hrleSparseBoxIterator. The hrleSparseCellIterator is specialised for efficient access to a unit cell of the rectilinear grid, which is important for the conversion of LSs to explicit surfaces using the marching cubes algorithm [170].

4.2 ViennaLS - Level Set Library

This high-performance C++ library uses the HRLE structure discussed in the previous section to store a LS surface and provides all the necessary algorithms to initialise the surface with a geometry, manipulate the LS values according to a velocity field, analyse features of the surface, and convert the LS to other material representations commonly used in device simulators. It therefore provides all the necessary tools to conduct topography simulations with level set surfaces and provides meshing algorithms to convert these results to be compatible with a device simulator. All algorithms are optimised for the HRLE data structure and thus provide optimal computational efficiency operating on the stored data by moving over all data sequentially and performing all computations in one traversal of the data structure. This library does not include volume information and can thus only be seen as a topography simulation library. However, it forms the basis for the process simulator ViennaPS by providing an accurate description of material interfaces and their evolution over time.

4.2.1 Software Design

Each surface is described by an instance of the class lsDomain which contains all the necessary information about the simulation domain, including the extent and boundary conditions of the simulation space, and the spacing and orientation of the grid. The sparse-field level set values themselves are stored inside the HRLE structure provided by ViennaHRLE within the lsDomain.

All algorithms are given an instance of lsDomain and operate on the data of this object once they are executed with a call to the algorithm member function apply(). This allows for all properties of algorithms to be initialised without data necessarily being available, followed by the execution of algorithms at a later point, when the data in lsDomain is available for manipulation.

In order to easily distinguish between algorithms and data, all classes holding data are named using nouns, while algorithms operating on data are named as verbs.

4.2.2 Geometry Creation

In order to quickly generate initial geometries for topography simulations, ViennaLS provides the algorithm lsMakeGeometry to fill a LS with one of four simple shapes:

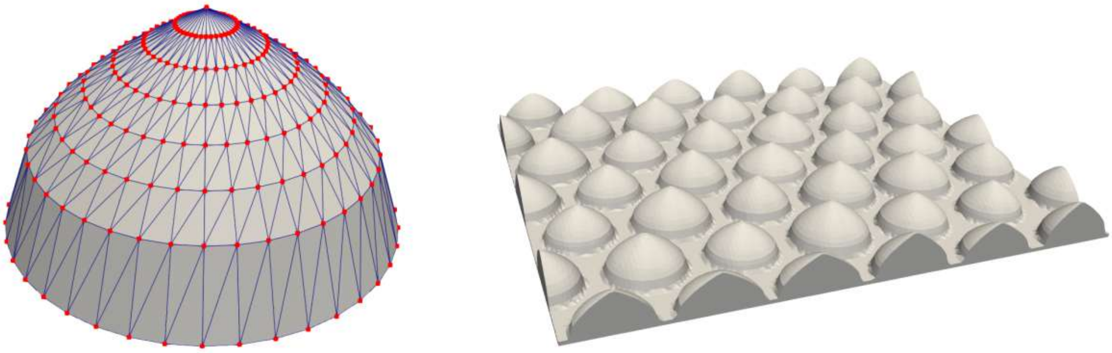
1. Sphere
2. Plane
3. Box
4. Cylinder

Only the surface of the sphere is generated directly in the LS due to its simplicity, while all other shapes are generated by first creating a triangulated mesh of the shape, translating, and rotating it as necessary, and converting it to a LS representation using the conversion algorithm described in Section 4.2.3.

Once a shape has been generated, it is stored in an lsDomain object, which can be combined with other shapes using Boolean operations, as described in Section 2.3.2.4. For these operations, ViennaLS uses the efficient implementation presented in [169]. The class lsBooleanOperation requires the user to pass a comparator function which takes the LS value from two lsDomain objects at the same point in space. This comparator should then return the LS value for the final surface at this point in space. For convenience, the most commonly used operations, presented in Section 2.3.2.4, are already implemented and can be used directly.

For the generation of more complex, user defined shapes, ViennaLS also offers custom geometries to be created using lsMakeGeometry. This is done by passing a point cloud to the algorithm, which will be converted to a convex hull surface mesh using the gift wrapping algorithm described in [171]. Therefore, more complex shapes, such as the rounded cone shown in Fig. 4.4a, can be created straight-forwardly. The red points shown in the figure are passed to the convex hull algorithm and converted to a triangulated surface mesh with the triangle edges shown in blue. This mesh is guaranteed to be closed, but does not necessarily satisfy other quality criteria, such as equal area triangles or equal edge lengths. However, the algorithm converting this mesh to a LS representation is robust to such meshes and only requires that the mesh is closed, meaning there are no edges which are not part of two triangles. Another restriction is that the resulting mesh is always convex, so in order to generate concave features, several convex structures must be combined using Boolean operations.

The resulting LS can be combined easily with other LSs using Boolean operations to produce large structures, such as the cone-shaped patterned sapphire substrates [172, 173] shown in Fig. 4.4b, used for enhancing the light extraction efficiency of light-emitting diodes [174]. Large and complex structures can be generated efficiently and straight-forwardly in the ViennaLS library due to the efficient algorithms for Boolean operations in the LS.



(a) Rounded cone triangulated from a point cloud (red) using the gift wrapping algorithm.

(b) Cone-shaped patterned sapphire substrate created by combining numerous level sets, each describing one cone. A final Boolean operation with a plane LS surface results in the final substrate.

Figure 4.4: Creation of a complex patterned substrate by (a) creating the convex hull of a point cloud to generate a cone and (b) combining numerous different cones with a plane surface using Boolean operations.

4.2.3 Conversion from Explicit Representations

The ViennaLS library offers algorithms for converting several different explicit surface representations to a sparse-field LS. All conversions follow the same general workflow. First, the explicit material is converted to a line-segmentation in two-dimensional (2D) or triangulation in three-dimensional (3D). Then the sparse-field LS values are found using a closest point transformation and are lexicographically sorted. Finally, the list of sorted points is inserted into the HRLE structure using the efficient initialisation described in Section 4.1.2. If the material interfaces are already given as line-segmented or triangulated surface meshes, they can be converted directly using the algorithm described in the following section.

4.2.3.1 Surface Meshes

This type of explicit surface representation only describes material interfaces using line-segments or triangles, referred to as mesh cells. Therefore, the LS value at each grid point $\vec{g}$ is given by:

$$\phi(\vec{g}) = \pm \min ||\vec{g} - \vec{x}_{s,g}||_1 \quad , \tag{4.1}$$

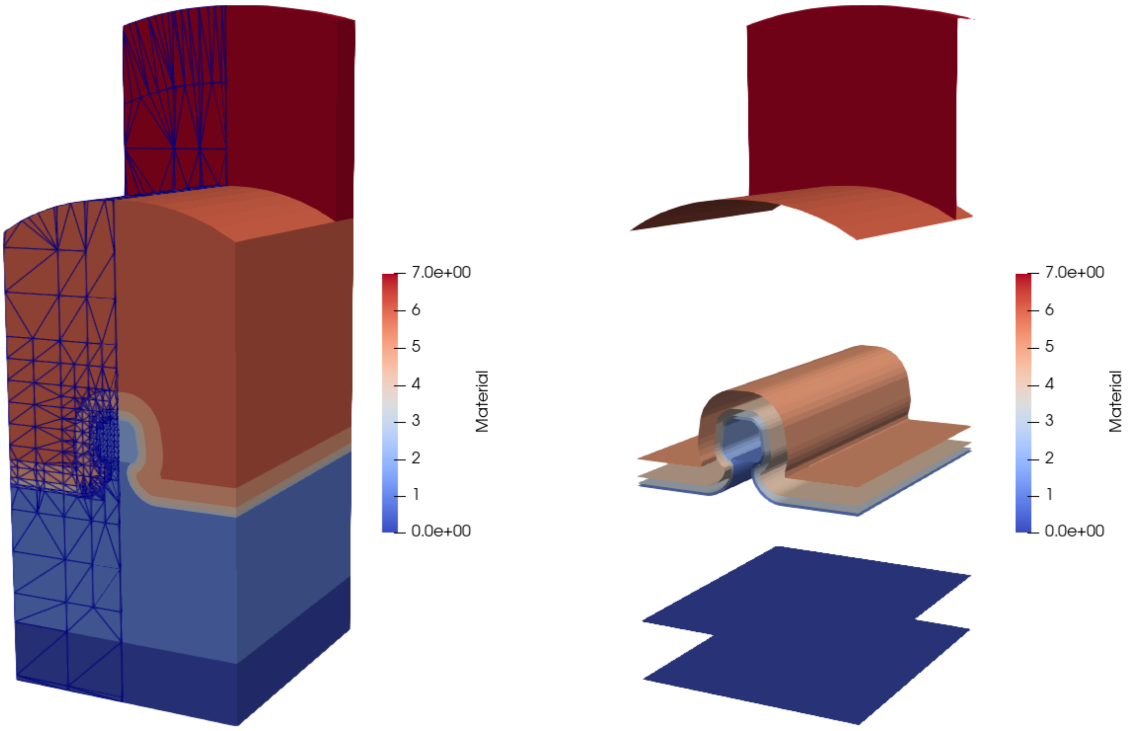
where $\vec{x}_{s,g}$ is a point which lies on the explicit surface as well as on a grid line of the LS grid. Only surface points which lie on a grid line are considered, since a Manhattan normalisation is used in the sparse-field method as described in Section 2.3.2.2. Therefore, in order to identify the LS values for the set of active points of the level set, each mesh cell is considered separately. For each cell, intersecting grid lines are identified and the LS values are set for all active grid points satisfying $|\phi(\vec{g})| \leq 1.0$. If a grid point has already been set using an earlier mesh cell, the smaller absolute value of the two is chosen for the grid point. Once all relevant grid points have been set using all mesh cells, the point list is sorted and used to initialise the HRLE data structure. The algorithm currently implemented in ViennaLS is an adaptation of the closest point transformation algorithm described in more detail in [169].

4.2.3.2 Volume Meshes

ViennaLS also supports the creation of level sets from triangle meshes in 2D and tetrahedral meshes in 3D. Each cell therefore represents a small volume of a certain material which is denoted by a material identifier for each cell in the mesh. In order to convert these types of meshes to LSs, they are first converted to several surface meshes, one for each distinct material. Each cell is first deconstructed into its surface representation, namely triangles into three line-segments and tetrahedrons into four triangles. These surface elements are then stored in a sorted container and if two surface elements are identical, i.e. contain the exact same nodes, but have opposite normals, they are removed from the list. For volume meshes, containing several materials, surface elements are only removed if the identical but opposite element has a material ID lower than the currently considered one. This automatically results in a surface mesh for each material, with the surfaces automatically following the layer wrapping approach required for the multi-material LS representation described in Section 2.3.2.5, as shown in Fig. 4.5. Finally, each surface is transformed to one LS using the closest point transform described in Section 4.2.3.1.

4.2.4 Conversion to Explicit Representations

It is often necessary to obtain an explicit representation of a surface for tasks such as visualisation [175], device simulation [176], or transport modelling using Monte Carlo ray tracing [177]. Therefore, the ViennaLS library allows for the conversion to several different types of explicit surfaces. In the following, each supported explicit representation is discussed in detail, including the required conversion algorithms and the most common applications for these representations in the context of microelectronic simulations are highlighted.



(a) Initial tetrahedral volume mesh.

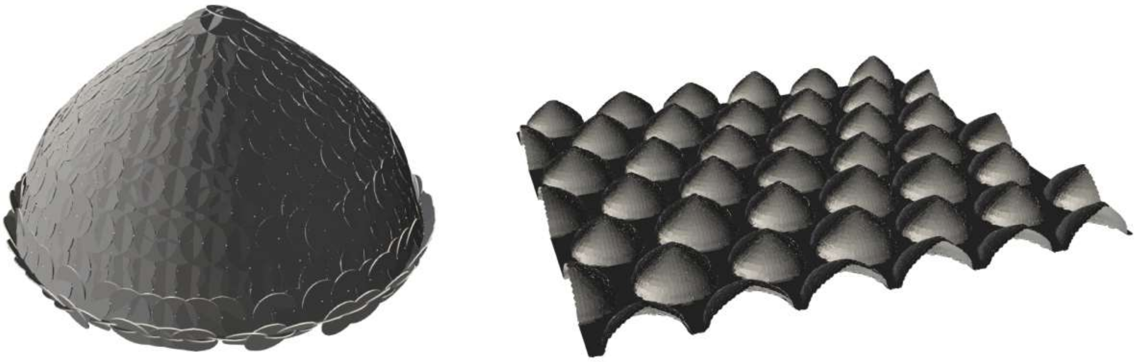
(b) Surfaces extracted from volume mesh.

Figure 4.5: Conversion of a tetrahedral volume mesh into several surface meshes representing the future LS surfaces, respecting the layer wrapping strategy of Section 2.3.2.5.

4.2.4.1 Disc Mesh

The most efficient way to extract an explicit surface from the LS grid is to shift all active points in the surface normal direction by the distance given by the LS value, as described in Section 2.3.2.3 and Eq. (2.15). This results in a point cloud of the explicit surface described by the level set. As discussed in Section 3.2.3.3, this type of mesh can be used to model molecular transport if a disc is centred at each point of this mesh, hence its name. The radius of the discs needs to be chosen carefully, so there are no holes in the mesh and the discs do not overlap unnecessarily, leading to large smoothing. The mesh itself does not contain this radius explicitly, but it is set during ray tracing. Fig. 4.6 shows that this type of explicit surface contains self-intersections and overlaps which are usually undesirable. However, for the application of Monte Carlo ray tracing, these properties do not have a major effect and thus can be ignored. The disc normal is set to the surface normal at the point around which it is centred, so that each disc forms a tangential plane to the surface at its centre. As can be seen in Fig. 4.6a, the high curvatures at the bottom of the cone result in the back of discs being exposed. Although this is not a problem for the modelling of molecular transport, it must be considered with great care when designing a ray tracing algorithm.

Generating this type of surface mesh can be performed in a single sweep across the HRLE data structure and is thus highly efficient. The only requirement is that the surface normals are known, in order to shift the grid points onto the explicit surface, as described in Section 2.3.2.3. Therefore, the LS has to be extended to include the necessary values required for the calculation of the surface normals.



(a) Level set created from the convex hull of a single cone, exported as a disc mesh.

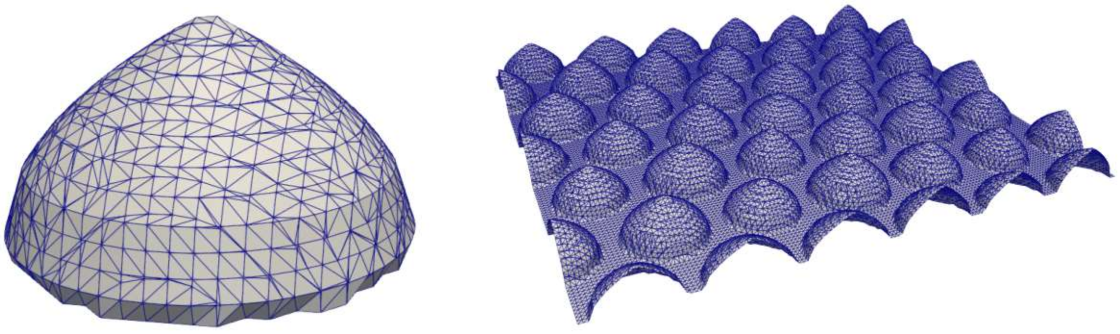
(b) The entire substrate shown in Fig. 4.4b exported as a disc mesh, as it would be used for Monte Carlo ray tracing.

Figure 4.6: Disc mesh used predominantly for efficient ray tracing. The mesh itself only contains points in space and the surface normals at these points. The discs are visualised to show that they form a closed surface appropriate for ray tracing.

4.2.4.2 Segmented Surface Mesh

Traditional ray tracing approaches employ conformal triangulated meshes in order to guarantee the properly normalised counting of surface ray intersections [178]. Therefore, triangulated surface meshes can be used for the modelling of molecular transport using Monte Carlo ray tracing, as well as visualisation applications. The marching cubes algorithm [170] is commonly used to generate a line-segmented or triangulated surface from implicit data. It works by considering one grid cell at a time, consisting of 2^D neighbouring LS values. Hence, it is called marching squares in 2D and marching cubes in 3D. Depending on the signs of the grid points and their LS values, different predefined surface elements are chosen from a lookup table. Iterating over all grid cells through which the material interface passes results in a properly extracted explicit surface. Therefore, the algorithm is linear in time with respect to the surface area and thus scales as $\mathcal{O}(N)$, where N is the number of LS values. However, since surface elements are added by just considering the current grid cell, the list of meshing nodes has to be checked in order to avoid duplicates. Using an associative data container, such as a hash, for storing the vertices, duplication can be avoided in constant time. The implementation uses the `hrleSparseCellIterator` discussed in Section 4.1.3 to access the LS values of a grid cell. Vertices are then inserted uniquely into a hash and connected using the lookup table of the marching cubes algorithm. All operations required to extract a triangulated surface can thus be executed in linear time representing the optimal computational efficiency.

Although the triangulation generated by the marching cubes algorithm is sufficient for some visualisation applications, it does not satisfy certain important quality criteria. For example, as shown in Fig. 4.7a, the triangles describing the cone vary strongly in size and some of them are thin and long, which can lead to catastrophic problems in some numerical algorithms used during device simulation. Hence, the meshes generated using this algorithm are not suitable for subsequent device simulation without additional re-meshing steps.



(a) Triangulation extracted from a cone LS using the marching cubes algorithm.

(b) The entire substrate shown in Fig. 4.4b converted from its LS representation to a triangulated surface mesh.

Figure 4.7: The marching cubes algorithm creates a conformal triangulation of the implicit data. Nevertheless, it leads to thin and sharp triangles which are unfavourable for certain applications. The edges of the mesh triangles are shown in blue.

4.2.5 Geometry Analysis

In order to enhance the understanding of a process and its effects on the simulated structures, it might sometimes be necessary to analyse the surface and its exact geometry. Due to the properties of the LS, certain types of analysis can be carried out efficiently and provide useful functionality for automation of process calibration or the efficient realisation of process emulation models. This section discusses the tools that the ViennaLS library provides for the analysis of the geometries represented by the LS.

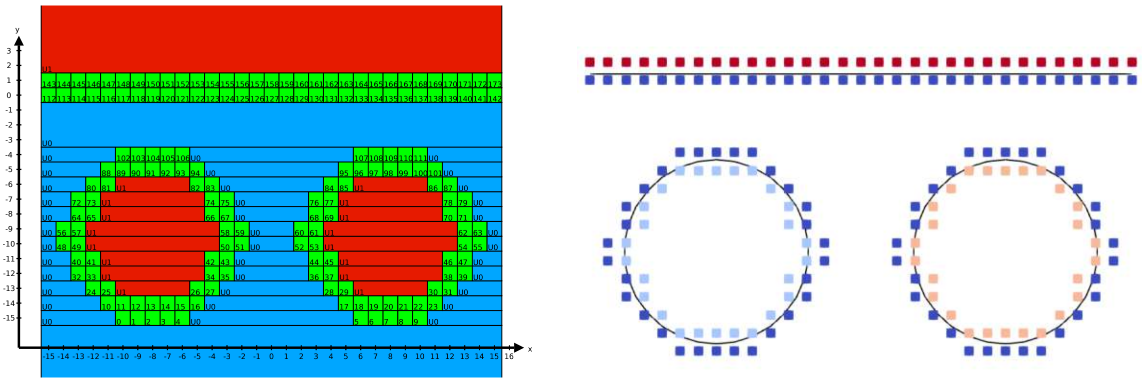
4.2.5.1 Connected Components

When dealing with complex surfaces, it is often useful to investigate how many disjoint regions of the surface are present in the simulation domain. If two defined grid points are connected by any number of direct first neighbours, they are part of the same region or component. If this is not possible because there are undefined grid points between these points, they are not connected and thus belong to two separate regions of the surface. In semiconductor process simulations, disjointed material regions usually mean that there is a void somewhere inside a material or that there is a material region which was separated from the substrate. In order to identify such material regions, the HRLE structure is traversed once and the connectivity information is built for every run as described in [169]. This connectivity information is represented by a component ID for each run, denoting to which surface region the respective grid point belongs. During one traversal of the HRLE structure using the `hrleSparseStarIterator` (see Section 4.1.3) the component IDs are assigned, as described in Algorithm 4.1. Hereby, it is important that every run in the segmentation of the HRLE grid is considered and not just the defined LS values.

```
Result: Connectivity information for LS grid
highestID = 0;
for point in hrleSparseStarIterator do
    if componentID not defined then
        if neighbours with sgn(neighbour) = sgn(point) then
            if 1 distinct neighbourComponentID then
                componentID = neighbourComponentID;
            else
                store connection between different componentIDs;
                componentID = any neighbourComponentID;
            end
        else
            componentID = highestID;
            highestID = highestID + 1;
        end
    end
end
```

Algorithm 4.1: Building connectivity information for a LS surface.

As a simple example, the segmentation of a material with two separated voids is shown in Fig. 4.8. Component IDs are assigned based on the neighbouring runs in the HRLE structure. All negative values have the same component ID in this example, as they are all connected through the undefined negative runs `U0`. Therefore, the disjoint material regions are given by the component IDs of positive values. Intuitively, this makes sense as there is one single material (negative values) and there are several holes denoting lack of material or being outside of the surface (positive values). If all LS values are inverted, then there is one substrate with two additional disjoint material regions. This usually occurs during etching when a geometry is separated and therefore generates two independent material regions. However, numerical inaccuracies can also lead to a small number of stray LS points remaining above the surface after advection. Such grid points are referred to as stray points and can be identified using the connected components algorithm.



(a) Segmentation of a material with two voids and infinite y directions.

(b) Defined points of the material showing the three component IDs denoting to which section of the surface each point belongs.

Figure 4.8: Using the segmentation of the HRLE structure, connected surface regions can be identified considering first neighbours. This allows for the identification of voids inside a material.

4.2.5.2 Void and Stray Point Detection

Having identified the connected components, as discussed in the previous section, it is straightforward to identify voids inside or stray points above the material. First the actual substrate must be identified, which can either be the material region with the most points or simply the material region situated in the most negative or positive direction. In the example of Section 4.2.5.1, the latter would be the case and the component ID set at the defined grid point with the most positive y-value would simply be chosen as the substrate ID. All grid points with differing component IDs must therefore be voids or stray points and can be treated as such.

In this example, only positive points were assigned differing component IDs. Therefore, negative points are not set based on their own component ID but rather based on the component ID of their nearest neighbours. If a negative point has at least one neighbour with the component ID associated with the substrate, it must be part of the substrate. Therefore, another sequential iteration over the HRLE structure is required in order to properly identify different material regions encompassing negative as well as positive values. The results of this identification, conducted on the example structure of the previous section, is shown in Fig. 4.9. When identifying stray points rather than voids, the signs of all of the above considerations must simply be inverted to correctly identify the different surface regions.

4.2.5.3 Feature Detection

For the analysis of simulated structures, the appropriate identification of geometric features is important in several applications, such as denoising [179] and automated image segmentation [180]. In order to identify features, the determination of the exact curvature of the surface is crucial. As described in Section 2.3.2.3, the curvature of

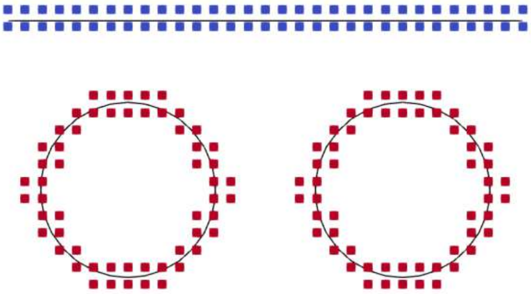
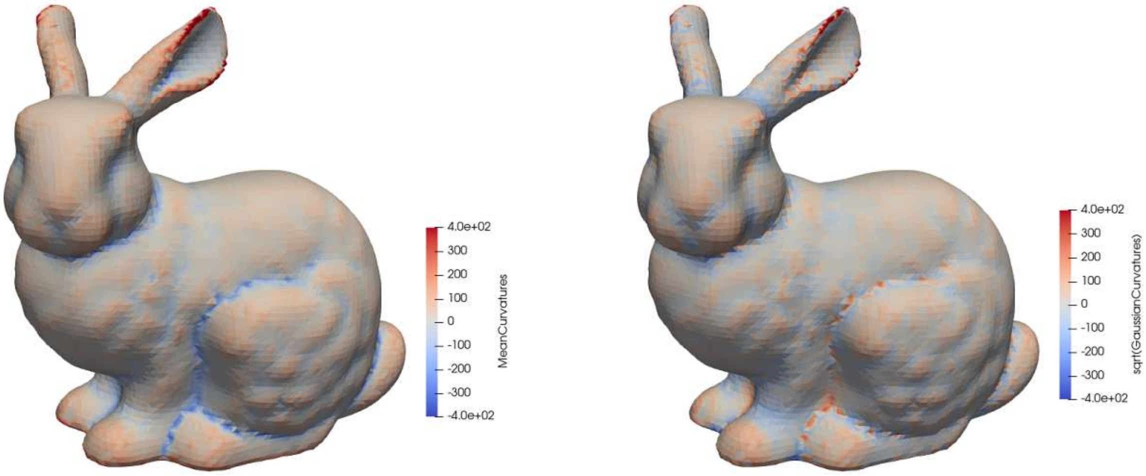


Figure 4.9: Voids inside a material properly marked by the presented algorithm. Red points refer to void points, while blue points refer to the substrate.

a surface described by a LS can be calculated directly without conversion to other surface representations. The curvature for the Stanford bunny test model is shown in Fig. 4.10 for the two types of curvature in 3D.



(a) Mean curvature for the defined points in the LS.

(b) Square root of the Gaussian curvature for the Stanford bunny LS with the original sign of the curvature.

Figure 4.10: Curvature calculated directly in the LS for all defined values of the Stanford bunny geometry. The root of the absolute Gaussian curvature is shown in (b) to compare to the mean curvature in (a), where a negative sign denotes concave curvature.

In the simplest case, these curvature values can be used directly for the detection of features. If the curvature exceeds a certain threshold value, the point of the surface is considered a feature. In 3D, only the mean curvature is used to identify features at first. However, the mean curvature may be zero at saddle points or other minimal surfaces, since the two principal curvatures carry opposite signs and may cancel. Therefore, when the mean curvature is below the threshold curvature for features, the Gaussian curvature is evaluated to test if the current point is a minimal surface and thus does in fact represent a feature. As only one threshold is set, it is compared to the square root of the absolute value of the Gaussian curvature, as the latter is equivalent to the square of the principal curvature on minimal surfaces.

This simple scheme for detecting features of the surface already leads to satisfactory results, as shown in Fig. 4.11. Features are highlighted in red and essentially form a binary cut off around a certain curvature value. Points of the surface which are part of a feature can thus be used for subsequent analysis of the structure.

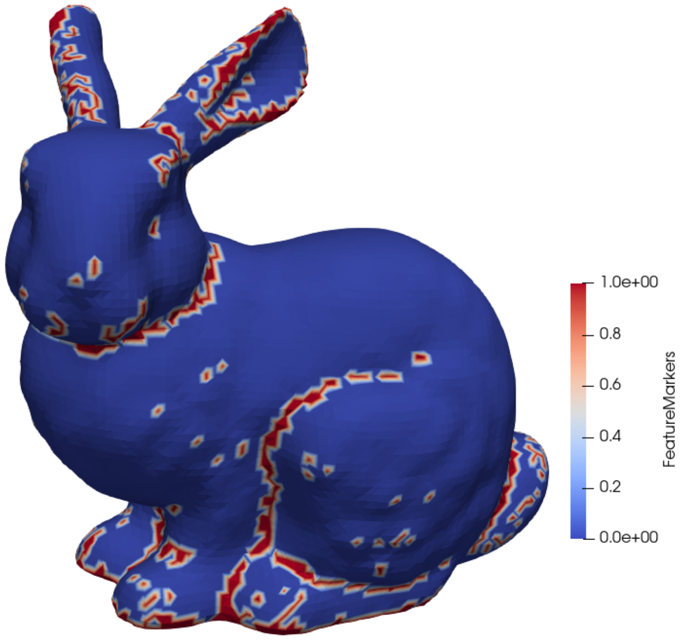


Figure 4.11: Features of the Stanford bunny LS highlighted in red with the feature detection threshold set to $\kappa = 110$.

4.2.6 Iterative Advection

Iterative advection can be implemented very efficiently using the HRLE structure, since all LS values are stored contiguously in memory. The implementation of the iterative advection algorithm consists of three main steps:

1. Calculating the surface velocity
2. Updating LS values
3. Rebuilding a valid sparse field LS

In the first step, the necessary change $\Delta\phi(\vec{g})$ in the LS value at the grid point $\vec{g}$ is calculated using one of the numerical integration schemes presented in Section 2.4.2. These values are then used in the second step to find the maximum time step which can be taken without violating the Courant-Friedrichs-Lewy (CFL) condition. The product of the time step and the change in LS value is then applied to all active LS values. Finally, in the third step, grid points are inserted, deleted, or adjusted in order to form a valid sparse field LS again.

In the following, each of these steps will be discussed in depth, highlighting implementation details and computational considerations.

4.2.6.1 Velocity Calculation

The way in which the surface should move is usually captured in a vector velocity field $\vec{V}(\vec{g})$. Using an appropriate numerical scheme, this velocity field must be converted to the change in LS value $\Delta\phi(\vec{g})$ which should be applied at the grid point $\vec{g}$. This change is equivalent to the Hamiltonian $\hat{H}$ of the numerical LS equation, presented in Section 2.4.2:

$$\phi(\vec{x}, t + \Delta t) = \phi(\vec{x}, t) - \Delta t \ \hat{H}(\phi(\vec{x}, t), V(\vec{x}, t)) \quad . \tag{2.33 revisited}$$

If there is only one LS surface this step is quite straight-forward, as the numerical integration scheme will return the correct value and the LS can simply be updated. However, if multiple materials are present, further considerations are necessary.

Multiple Materials

Using the layer wrapping approach for multiple materials, presented in Section 2.3.2.5, the current material at $\vec{g}$ has to be identified first. In the presented implementation, several `lsDomain` objects are passed to the advection algorithm in the wrapping order. The material ID denoting which LS represents which material in the simulation domain, is simply the array index to the respective LS. Therefore, a LS with a higher material ID must wrap all LSs with lower material IDs. Due to the way in which the layers are wrapped using Boolean operations, the material present at the grid point $\vec{g}$ is described by the LS with the lowest value at this point. Hence, the correct material is found using the following algorithm:

```
Result: Material ID of current point $\vec{g}$
materialID = maxID;
i = 0;
while $i < maxID$ do
    if $\phi_i(\vec{g}) \leq \phi_{maxID}(\vec{g})$ then
        materialID = i;
        break;
    end
    i = i + 1
end
```

Algorithm 4.2: Identification of the material at the grid point $\vec{g}$ from an array of LSs.

Once the correct material has been identified for each active grid point, the velocity field is constructed using the material information.

Iterative advection can be computationally expensive, due to the numerical derivatives which need to be solved in order to find a solution to the LS equation. Therefore, when advecting several materials, only the top most layer which encompasses all other LSs is advected. This is enough for a deposition process, since a new material is simply added and the initial materials do not change. Hence, advecting only the top LS is enough to describe the physical process.

However, when etching a material or when growing a material around a feature, further considerations are necessary in order to encompass the evolution of several LSs. In this discussion, we will consider the etching of a material around a mask. First, when advecting a material which is masked in certain areas, the top surface can only move to the mask and not further. Therefore, when the top layer would move inside the mask material, it must not be advected further. The simplest way to realise this is to find the LS value of the layer below and to use it to set the advected LS value, in case the top surface would advect further. However, the lower layer might also have a non-zero etch rate if the mask is worn away by the etch process. In this case, the new value for the advected level set $\phi(t+\Delta t)$ must be found by calculating the Hamiltonian for the top layer $\hat{H}_0$ and applying it for the time Δt_0 until the lower surface is reached. The time Δt_0 is found by dividing the difference in level set value of the top and bottom layer by the Hamiltonian of the top layer. From then on, the Hamiltonian $\hat{H}_1$ for the material below is found and applied until the maximum time Δt for this time step is reached, usually determined by the CFL number. Therefore, the change in LS value is given by:

$$\Delta\phi = \hat{H}_0(V,t)\Delta t_0 + \hat{H}_1(V,t)\Delta t_1 \quad \text{where} \quad \Delta t = \Delta t_0 + \Delta t_1 \tag{4.2}$$

An algorithm using the above approach for any number of materials is presented in Algorithm 4.3. Using this approach the different rates for each material can be represented appropriately although only the top LS surface is advected.

Once the above algorithm has concluded for all active grid points, the LSs which were not advected are adjusted by applying a Boolean intersection operation. This means that a material which was removed during the advection of the top LS is also removed for all other level sets. Hence, a set of correctly wrapped LSs is created, including the specific etch rates of all materials.

4.2.6.2 Updating LS values

The CFL condition given in Eq. (2.35) governs how far a sparse field LS can be moved robustly without encountering numerical instabilities, and may be rewritten as:

$$\max_{\text{all } \vec{g}} |\Delta\phi(\vec{g})\Delta t| \leq 0.5 \quad . \tag{4.3}$$

Since $\Delta\phi(\vec{g})$ is already given from the previous step, the maximum permitted time step Δt has to be chosen so that the above condition is satisfied for all points $\vec{g}$. Since the time step is defined globally, it must be the same and therefore the smallest time step must be chosen for the entire simulation domain:

$$\Delta t = \min_{\text{all } \vec{g}} |0.5/\Delta\phi(\vec{g})| \tag{4.4}$$

```
Result: Change in LS value $\Delta\phi$
cflNumber = 0.5;
topIndex = number of LSs - 1;
i = topIndex;
$\Delta\phi = 0$;
for $i >= 0$ do
    calculate $\hat{H}_i$;
    diff = $\phi_{i-1}(t)$ - $\phi_i(t)$;
    if $diff \leq cflNumber$ then
        $\Delta t_i$ = cflNumber - $\sum_{j=i+1}^{topIndex} \Delta t_j$;
    else
        $\Delta t_i$ = diff / $\hat{H}_i$;
    end
    $\Delta\phi = \Delta\phi + \hat{H}_i \, \Delta t_i$;
    i = i - 1
end
```

Algorithm 4.3: Algorithm for the change in LS value for a grid point close to several material interfaces.

The new LS values at each grid point are then set as the product $\Delta\phi(\vec{g})\Delta t$, which can be carried out highly efficiently as all defined LS values are stored contiguously in memory using the HRLE data structure.

4.2.6.3 Rebuilding a Valid Sparse Field LS

In the previous step, each LS value may have been changed by a maximum of ± 0.5, meaning that some grid points may not be a part of the $\mathcal{L}_0$ layer anymore and must therefore be removed, while other grid points may have moved into the $\mathcal{L}_0$ layer. In order to generate a valid sparse field LS, three rebuilding steps are required:

1. Add grid points which have previously not been in $\mathcal{L}_0$
   and now satisfy $|\phi(\vec{g})| \leq 0.5$

2. Remove grid points which were part of $\mathcal{L}_0$
   and are not active points anymore

3. Renormalise two oppositely signed neighbours
   if both have values larger than 0.5

The last step is not part of the original algorithm and was proposed in [169] in order to avoid invalid LSs for diverging velocity fields. If this step was not considered, two grid points of opposite signs with the surface passing between them could be advected in such a way that neither was active anymore. Although this is only possible if the negative one was reduced and the positive one increased, there is no general limitation on the velocity field disallowing such movement. Hence they could both be removed, although the surface passes between them. In order to avoid this problematic outcome, this third step was introduced, essentially redefining their values so that they are active points again.

All of the above steps can be carried out simultaneously by iterating over the HRLE structure once. During this iteration, a new data structure is built, so the old LS data stays intact. An algorithm providing the functionality to rebuild the sparse field LS after advection using only first neighbours is provided in Algorithm 4.4.

```
Result: Valid LS from advected LS
newLS = empty LS;
for point in hrleConstSparseStarIterator do
    oldValue = LS value at point;
    if point is active then
        if point has oppositely signed neighbour then
            if neighbour not active then
                newValue = $\mathrm{sgn}(oldValue)\,0.5$;
            else
                newValue = oldValue;
            end
            insert newValue into newLS at point;
        end
    else
        if point has oppositely signed, active neighbour then
            neigbourValue = LS value of neighbour;
            if $oldValue > 0$ and $|neighbourValue + 1| < 1$ then
                newValue = neighbourValue + 1;
            end
            if $oldValue < 0$ and $|neighbourValue - 1| < 1$ then
                newValue = neighbourValue - 1;
            end
            insert newValue into newLS at point;
        end
    end
end
```

Algorithm 4.4: Adding, removing, and updating grid points after advection to form a valid sparse field LS.

After this algorithm has concluded, all grid points with absolute LS values smaller than 1 will be defined, and all others undefined. Although this is not the smallest set of LS points required to describe the surface, including points up to unity rather than up to 0.5 leads to better surface descriptions in areas of high curvature. Therefore, the smallest possible set of grid points robustly describing the surface is created.

4.2.7 Geometric Advection

The geometric advection algorithm implemented in the `ViennaLS` library is an efficient implementation of the algorithm introduced in Section 2.4.3. Two parts of this algorithm are essential for high performance: Identification of candidate points, and identification of contribute points. The former requires fast random access in order to iterate over candidate points in the most efficient manner, while the latter requires the fast identification of close points within a certain distance to the current candidate point. Therefore, the HRLE structure is not well suited for this application, although it can be generated quickly from the set of new LS points resulting from the algorithm.

4.2.7.1 Efficient Identification of Candidate Points

As described in detail in Section 2.4.3, candidate points are grid points which will be active points in the final LS surface after advection. The simplest approach to finding these points is to check every grid point in the simulation domain, and calculate its final level set value from its specific contribute points. However, this algorithm scales as $\mathcal{O}(N^D)$, where N is the number of grid points in each dimension D of the simulation domain.

Better scaling can be achieved by considering the properties of the LS surface. Since every grid point in the simulation domain contains at least the information regarding whether it is inside or outside of the surface, namely its sign, there cannot be any holes in the surface. Since any surface represented in the LS must be closed, every active point must have at least one neighbour which is also an active point. For added robustness, the set of active points can be extended to LS values up to unity rather than 0.5, as described in Section 2.4.3. Therefore, if one candidate point of the final surface can be identified, its LS value can be calculated and its neighbours checked to see which one of them is also an active point. If a neighbour is not an active point of the final LS, it is ignored. If it is an active point, then its neighbours are checked and the above procedure is repeated, as presented in Algorithm 4.5. The algorithm effectively marches over the surface, saving active points and their corresponding LS values in an associative container, such as a hash map. If an active point is found which was calculated previously, it is ignored and therefore treated as a non-active point.

The first candidate point required to start the algorithm can be identified by considering any initial surface point, centring the geometric distribution around it, and marching through all grid points within the distribution to find one point with an active value. Therefore, the computational effort to identify the first active point is limited and can thus be ignored for large surfaces. Since every active grid point of the final surface only has to be visited once, the algorithm scales as $\mathcal{O}(N^{D-1})$, corresponding to optimal complexity.

If the surface consists of several disconnected components, as discussed in Section 4.2.5.1, the above algorithm would only apply to the disconnected region to which the first point belongs. Hence, the algorithm requires the identification of the

```
   Result: Set of active points of the new surface
1  pointMap = empty hash map;
2  neighbourSet = empty set;
3  a = first active point;
4  add a to hashMap;
5  add neighbours of a to neighbourSet;
6  while neighbourSet is not empty do
7      c = pop point from neighbourSet;
8      if c is not in hashMap then
9          value = find LS value for c;
10         if $value < 1.0$ then
11             add c to hashMap;
12             add neighbours of c to neighbourSet;
13         end
14     end
15 end
16 construct new LS from hashMap;
```

Algorithm 4.5: Efficiently identify candidate points by marching over the surface by looking at direct neighbours of active points.

connected components prior to execution. For each component, an initial point must be found and added to the *hashMap* in Algorithm 4.5. When all disconnected components contain a starting point, the algorithm can commence as above, taking into account all disconnected components of the LS surface.

In order to identify whether a candidate point is an active point of the final surface, all its corresponding contribute points must be evaluated. An efficient algorithm for the identification of contribute points is described in the next section. In Algorithm 4.5 this algorithm is referred to as finding the LS value of the candidate point c (line 9). The calculated LS value is then used to identify whether the point is active.

Finding the neighbour points of any active point is highly efficient, since by definition, all LS values are defined on a rectilinear grid and thus all neighbours are separated by one grid spacing in all Cartesian directions.

4.2.7.2 Limiting the Number of Contribute Points

The final LS value of each candidate point is calculated from the set of contribute points, which consists of the points close enough to the current candidate point in order to influence its value. Therefore, contribute points only form a small subset of all initial surface points. The fast identification of this subset of initial points is essential for the improved performance of the algorithm. However, surface points are represented explicitly and therefore are not structured in any particular way. In order to quickly access nearest neighbours, a k-d tree data structure is established by sorting the initial points according to their coordinates. The data structure can be queried for all stored points in the neighbourhood of a passed coordinate, returning an iterator over the range of points close to the passed coordinate. By setting the size of the queried neighbourhood to the bounding box of the geometric advection distribution, the minimum number of contribute points can be used for the algorithm.

4.3 ViennaCS - Cell Set Volume Data

This data storage library builds upon the ViennaHRLE library in order to provide an easy to use interface for storing and manipulating volumetric data. It supports conversions from a LS surface into filling fractions which can be set by the user as binary, meaning whether the cell is empty or filled, or as a continuous value from 0 to 1 representing how much material is found within the cell. Therefore, ViennaCS can be used to store an explicit volume representation using voxels, as well as the implicit location of the surface using filling fractions. The implicit filling fraction values for two intersecting spheres of different material are shown in Fig. 4.12. The volume inside the material, where the majority of cells are filled entirely and thus have a filling fraction of 1, are not stored explicitly in the HRLE data structure. They are represented using undefined runs and thus do not occupy additional memory. Therefore, the volume representation is as memory efficient as the sparse-field LS since it only stores the filling fractions for cells close to an interface.

The data stored in each cell is defined by the user, allowing them to specify the values according to which defined cells should be represented. In order to identify undefined runs in the HRLE data structure, the user may define background cells with specific values. A cell will then be represented as an undefined run if the data stored inside it compares as equal to a background cell or as a defined run otherwise. As the equality check is also implemented by the user, fine control over the exact type and ranges of data which should be stored explicitly is possible. This permits the user to increase memory efficiency by specifying which combination of values should compare equal to which background cell. For example, if the cell set data structure is used to store implanted ions inside a material, the user can define a lower threshold of implantation concentration below which a cell compares equal to an empty background cell. Furthermore, a user could implement any number of background cells representing all allowed values and only store undefined runs, describing the entire volume data using only these discretised background cells. Hence, depending on the application, accuracy can be traded for memory efficiency in order to achieve the most suitable results at the user's discretion and depending on the application.

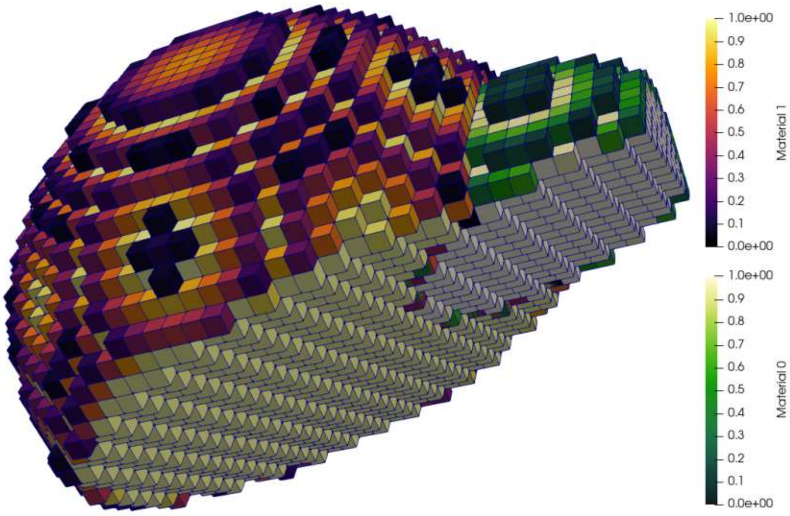


Figure 4.12: Filling fractions for two different materials, represented using implicit volumes. The cells which contain both materials have two filling fractions to describe the percentage of the cell volume each material occupies.

4.4 ViennaRay - Transport Modelling

The ViennaRay ray tracing library is based on the physical modelling approaches presented in Section 3.2.3.3. Rays are launched from a source plane above the geometry and their paths are traced until they intersect the surface which is defined using the disc mesh described in Section 4.2.4.1. The discs are not passed explicitly to the ray tracer, but are rather only represented as a point cloud and the corresponding surface normals. The discs are then implicitly generated by the ray tracer using a predefined value for the disc radius. The minimum disc radius which leads to a closed surface mesh is $\frac{1}{2}\sqrt{D}\Delta g$ which is used in this work since it provides the smallest robust surface area and thus avoids unnecessary smoothing during ray tracing.

In order to adequately describe interactions with the surface, which may influence the particles behaviour, such as reflection angle, energy loss during reflection, and sticking probability, the ray tracing library allows the user to implement their own particle type. As shown in Fig. 4.13, the user-defined particle type object must provide three callbacks:

1. `initNew`: Sets up all initial values for the ray, such as the initial energy for modelling ions.

2. `surfaceCollision`: Called when the particle intersects the surface. Used to model the interaction with the surface based on particle specific properties, such as incoming angle, energy, and sticking probability. The results are stored in the thread local data of the ray tracer. Since several discs may overlap and therefore create more than one intersection, this function may be called multiple times for one surface hit.

3. `surfaceReflection`: Called only once for the first intersection with a disc. It is used to set the reflection probability and the direction of the reflected particle, based on surface properties. Since a particle cannot change the surface upon reflection, this function is not allowed to change the thread local data of the ray tracer. However, in order to properly model surface dependent reflections, surface properties assumed to be constant during the ray tracing, such as different particle coverages, are accessible.

For convenience, a simple particle type is provided as a base implementation. This particle does not implement any reflections, but simply sticks or adsorbs on the surface. The disc, which is hit by the particle is recorded by the ray tracer and used to generate a hit counter for each surface element. For user defined particles, this counting of hits is unnecessary, as the particle type should implement the functionality for capturing the incoming particle flux in the thread local data of the ray tracer. Therefore, properties of the ray, such as incoming angle, energy, or initial flux can be used for the calculation of the surface flux directly.

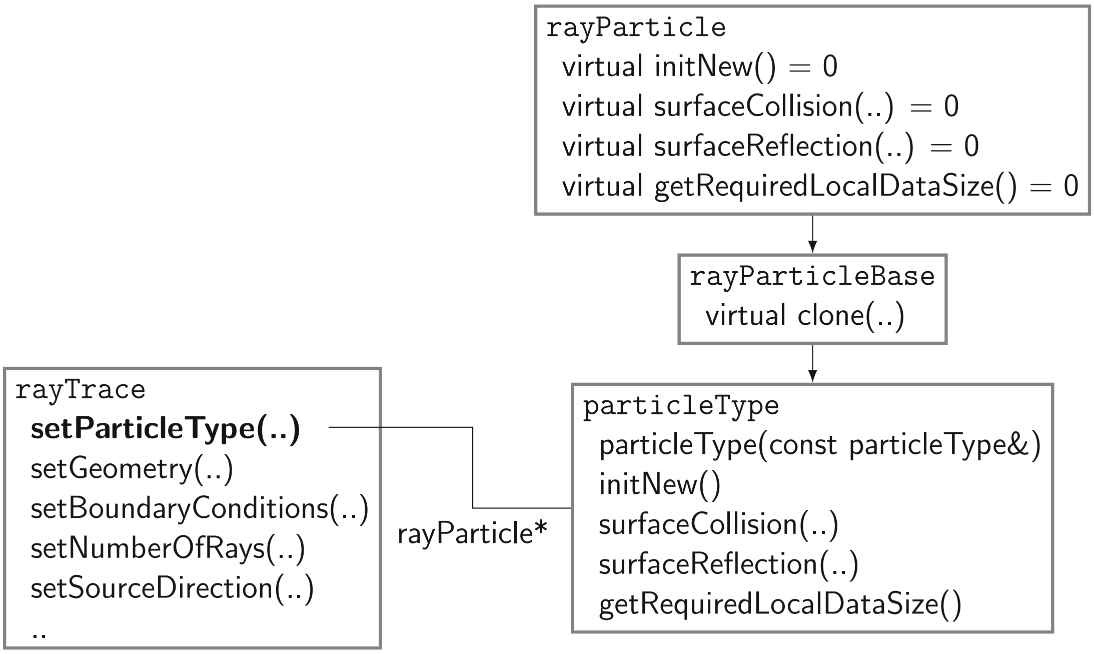


Figure 4.13: Interface of the ViennaRay library for defining the properties of the simulation domain. The particle type is passed using the abstract base class `rayParticle`, so the particle type can be assigned dynamically during runtime. Custom particle types should always be inherited from `rayParticleBase`, which contains the functionality for accessing the copy constructor of the user-defined particle type.

The particle type is then defined by deriving from the `rayParticleBase` class and passing it to the ray tracer, as shown in Fig. 4.13. As this particle type is used to trace each ray, it must have a defined copy constructor, so it can be duplicated via the `clone` member function of the base class. The two most common reflection types, diffuse and specular reflection, are provided as free functions. Hence, a user may apply them efficiently in the implementation of the custom particle type. These functions simply return the direction of the outgoing ray, given the direction of the incoming ray, as well as any information about the surface and the intersection point. Since all parameters of the ray tracer can be set during runtime, including the user defined particle type, the library does not need to be recompiled when changing particle models and can thus be used in the most flexible manner possible.

4.5 ViennaPS - Process Simulation

The ViennaPS software library ties all of the above mentioned tools together and aims to provide a straight-forward and simple to use interface to users without requiring in-depth knowledge of the mathematical concepts and numerical intricacies of the underlying simulation tools. It contains the necessary interfacing code to connect the different software libraries as well as a comprehensive process modelling interface and intuitive analysis tools.

In order to allow for the greatest possible flexibility, the entire simulation domain is encompassed within the `psDomain` class, which contains one LS for each material and one single cell set (CS) for storing the volume data of all materials. When the modelling framework is used to simulate a process, the entire domain is passed, so each model may either use the LS surface, the CS volume data, or both as needed to describe the changes introduced by the process. This modelling framework is the heart of the `ViennaPS` library and is discussed in detail in the following.

4.5.1 Modelling Framework

In order to allow for a full description of a semiconductor fabrication process, several steps are required to form a comprehensive chemical model, as discussed in Section 3.2. These steps are highlighted in Fig. 3.1, where the first step of modelling the molecular transport in the gas phase is carried out by the `ViennaRay` library using the user-defined particle type. The resulting particle flux can then be applied to find the surface coverage by modelling the molecular interactions with the surface, which is encompassed in a surface model in the presented library. The final step of modelling the chemical reactions of molecules on the surface results in the etch or growth rates and is also incorporated in the surface model. This surface model thus defines how the surface will advance, given the incoming particle fluxes and the chemical properties of the substrate, as described in Section 3.2.4 and Section 3.2.5. If surface coverages are needed in order to find the final surface rates and thus the velocities for LS advection, they have to be calculated inside the surface model.

Additionally, in order to simulate volume processes, a volume model may be implemented, which can access the LS and CS data structures to generate a velocity field for LS advection from the stored data. After advection, the CS is updated using the new level set values, adding or removing cells accordingly. The software design and interplay of different objects in the modelling framework is visualised in Fig. 4.14.

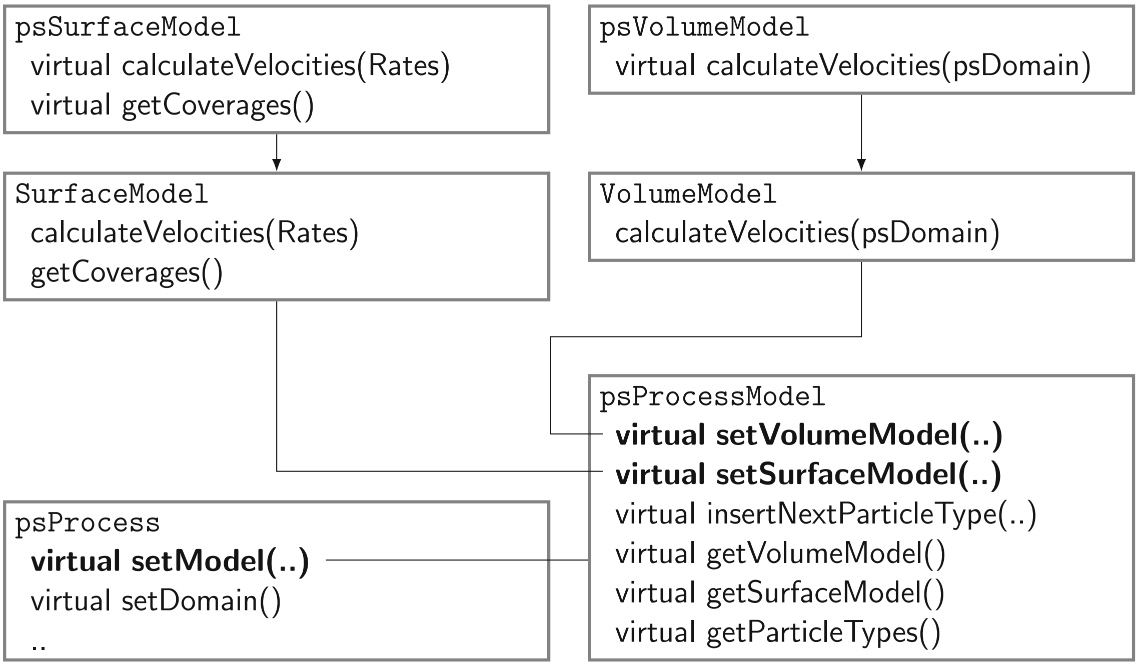

psSurfaceModel
virtual calculateVelocities(Rates)
virtual getCoverages()
psVolumeModel
virtual calculateVelocities(psDomain)
SurfaceModel
calculateVelocities(Rates)
getCoverages()
VolumeModel
calculateVelocities(psDomain)
psProcessModel
virtual setVolumeModel(..)
virtual setSurfaceModel(..)
virtual insertNextParticleType(..)
virtual getVolumeModel()
virtual getSurfaceModel()
virtual getParticleTypes()
psProcess
virtual setModel(..)
virtual setDomain()
..

Figure 4.14: Interface and inner workings of the modelling framework of the `ViennaPS` library. An object of the type `psProcessModel` is passed to the `psProcess` class which simulates the implemented model on the passed `psDomain`. This includes performing the ray tracing, executing the surface model and the volume model if they are defined. None of these are required, so emulation models can be carried out by, for example, only defining a surface model.

Therefore, all the information about the process is encompassed in the class `psProcessModel`, as it includes all the particle type information required for ray tracing, the surface model, as well as the volume model describing chemical processes inside the material. This class may also be used as a base class for the definition of process models for specific chemistries, where all particle types, the surface model, and the volume model may be defined inside the derived class rather than setting them using objects defined elsewhere. This design gives software developers greater flexibility for the implementation of specific chemical models, while still providing the possibility of mixing predefined particles types with surface and volume models using the `psProcessModel` class directly.

Additionally, each part of the model, or the entire model itself, does not have to be known at compile time, so they can be compiled and integrated in the simulator separately, avoiding the recompilation of the entire library when a new model is introduced. Furthermore, the fact that models must not be known at compile time means that the library is suitable to be packaged for interpreted languages, such as Python. Especially Python is widely used in the scientific community due to its intuitive design and wide range of easily accessible scientific tools. Therefore, the `ViennaPS` library can benefit greatly from being easily accessible to the scientific community as a Python package.

The flexible design of the ViennaPS modelling framework allows for the implementation of many different process models building upon numerous numerical concepts and computational techniques. Many common processes of interest can be simulated or emulated with ease without requiring in-depth knowledge of the underlying numerical methods used for the simulation. In the next chapter several process models developed within the ViennaPS framework are presented. The modelling approaches and input parameters are discussed in detail and the resulting geometries are compared to expected analytical results and experimental measurements.

4.6 Summary

Several independent libraries were implemented during the course of this work, culminating in a complete process simulation library, ViennaPS.

ViennaHRLE provides an implementation of a hierarchical run-length encoded data structure for the efficient storage of sparse data. It provides sequential data access with constant time complexity and random access with $\mathcal{O}(\log N)$. In addition to defined values, any number of undefined values can be stored, which occupy more than one grid point and can therefore be used conveniently to store background values describing large regions of the simulation domain. Several iterators were implemented, giving highly efficient access to neighbouring grid points, the number and arrangement of which depends on the specific iterator.

ViennaLS is a high-performance level set library based on the ViennaHRLE data structure. It provides all the necessary features for topography simulations, such as fast geometry creation, and conversions from and to several other material representations, such as triangular or tetrahedral meshes. An algorithm for the extraction of disc meshes for efficient transport modelling, as described in Section 3.2.3.3, has been developed and is provided with the library. Several algorithms for the analysis of the stored level set surface have been developed. These include the analysis of connected regions of the level set used for the detection of voids inside the surface or disconnected stray points outside of the surface. Feature detection directly on the level set was developed and is implemented using different underlying detection algorithms.

An efficient algorithm for iterative level set advection of multiple materials has been improved from the original ViennaTS framework and implemented in ViennaLS. The geometric advection algorithm developed during the course of this work is implemented using efficient algorithms for candidate and contribute point identification. This feature can be used highly efficiently for process emulations of multiple materials.

ViennaCS provides an efficient wrapper for storing volumetric data using ViennaHRLE. Monte Carlo ray tracing for top-down particle transport modelling is provided within the ViennaRay library, which provides an intuitive interface for the modelling approach presented in Section 3.2.4.

Finally, ViennaPS contains the implementation of a modelling framework for the emulation and simulation of semiconductor fabrication processes. This library employs all of the above software tools to provide the functionality required to combine emulation and simulation models within an intuitive interface.

Chapter 5

Process Modelling

The precise tuning of process parameters is the most crucial step in the simulation of semiconductor fabrication steps. Since the general setup of any empirical or physical model remains largely as described in Section 3.2, the only differentiating property between models is the set of input parameters. The specific values in each model, such as threshold energy, substrate density, or stoichiometric factors, must therefore be determined experimentally or derived from first principles. This is not only true for sophisticated physical simulations, but also for empirical emulation models, which rely heavily on experimental observations.

In this chapter, several models describing semiconductor fabrication process steps are presented as they have been developed or implemented in the simulation framework developed over the course of this work. Several simulations are based on previously published models and have only been implemented to show the broad capabilities of the simulator, while many other models have been developed over the course of this work and are presented for the first time. The input parameters, as well as their physical source are discussed for each model, highlighting values of highest importance. The results of these models are then analysed and compared to expected analytical results and empirical measurements. Finally, process flows developed during this work for entire devices are presented, which lead to accurate process-aware descriptions of the final geometry. The results of these simulations can be used to extract electrical properties of the device in order to conduct a circuit simulation including variations introduced during the manufacture.

5.1 Fabrication Steps

During the course of this work, several models describing single fabrication steps have been developed. In this section, these models are presented and critical features of the resulting geometries discussed. This includes an explanation of the underlying physics as well as particle transport and surface mechanics. The modelling approaches for emulation and simulation are thereafter presented for each model, highlighting their differences and the resulting features observable in the final geometry.

5.1.1 Chemical Vapour Deposition

Chemical vapour deposition (CVD) is one of the most commonly used processing steps in the manufacture of semiconductor devices. It is used to deposit a layer of material on top of another substrate and is commonly applied for the deposition of polysilicon [181, 182], silicon dioxide [183], silicon nitride [184] and tungsten [185]. In the simplest case, the feed gas can be simulated by only considering one precursor as the primary source of deposition, as is the case for CVD using silane [186]. In this case, only one particle type needs to be considered in the model and the deposition rate on each surface element is proportional to the incoming flux of this particle type [187]. However, for more complex feed gases or for the combination of several different feed gases into the reactor, more complex behaviours have been observed.

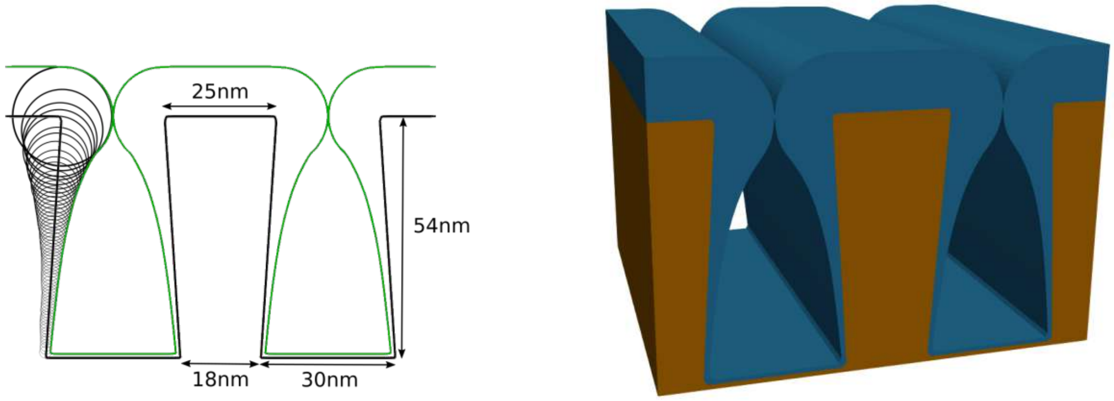
One such example is the deposition of silicon dioxide through tetraethyl orthosilicate (TEOS), where the initial feed gas dissociates into several precursors with highly different properties [188]. It is assumed that TEOS is the predominant precursor, but it only has a very low sticking probability and therefore does not contribute much to the deposition rate, while another molecule which dissociated from TEOS is very reactive and therefore has a high sticking probability. Since this other molecule only makes up a small fraction of all particles in the reactor, it does not completely dominate the deposition rate, rather it ensures subtle non-isotropy on the substrate [188]. Similar models have been previously developed and applied for the CVD of silicon nitride [189] and tungsten [190].

5.1.1.1 Empirical Model

As mentioned in the previous section, in the simplest case, there is only one precursor species responsible for deposition on the substrate. The deposition rate is therefore proportional to the incoming particle flux at each point on the surface. A crude approximation to the flux is achieved by considering the view factor of the source plane for each surface point. If the simplifying assumption of a sticking probability of 1 is made, meaning that particles only hit the surface once and then deposit at that location, the deposition rate is directly proportional to the incoming flux. This model will thus only appropriately represent processes with very high sticking coefficients and additional isotropic deposition terms will have to be included to account for lower sticking probabilities. If properties of the initial geometry are known, an analytical expression can be found for the incoming particle flux inside trenches and vias [191]. For an infinite trench, the ideal analytical solution is given by [192]:

$$F(d) = \frac{1}{2} + \frac{cos(\theta) - \frac{d}{D}}{2\sqrt{1 - 2\frac{d}{D}cos(\theta) + \left(\frac{d}{D}\right)^2}} \quad , \tag{5.1}$$

where $F(d)$ is the view factor of the source plane for a point a distance d down the trench sidewall which is tapered by the angle θ. In [106] a pinch-off CVD process for air-gap creation in back-end of line (BEOL) copper lines was emulated using this approach and satisfying results were achieved highly efficiently. Fig. 5.1a shows how each value for the view factor was used as a spherical geometric distribution kernel to achieve the resulting geometry. The generated three-dimensional (3D) geometry shown in Fig. 5.1b is an analytically correct representation of the model described by Eq. (5.1). Despite the model ignoring the build-up of material over time and the resulting change in trench diameter at the top, it produces the expected results exactly and is computationally efficient, due to the geometric advection used to simulate the process.



(a) Schematic showing how the described model creates the resulting geometry (green) using spherical geometric distributions (black circles). The outline of the circles thus creates the final geometry, based on the view factor calculated using Eq. (5.1). Reproduced from [106].

(b) Geometry resulting from the three-dimensional geometric advection model for a pinch-off CVD process for air-gap creation on top of copper lines (orange) depositing a dielectric (blue).

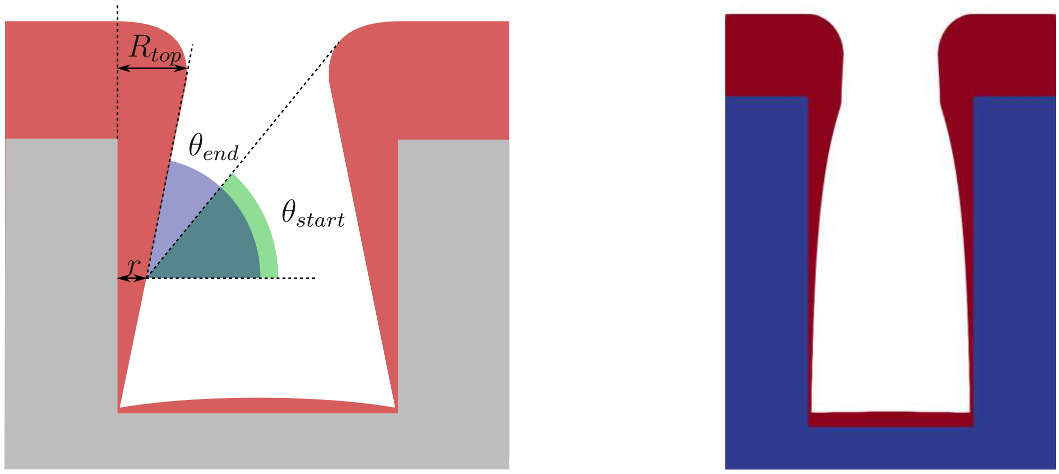
Figure 5.1: Resulting geometry of the geometric advection of a pinch-off CVD process used to generate insulating air-gaps in copper lines through the deposition of a dielectric.

5.1.1.2 Physical Model

Extending the model from the previous section, intricate physical effects specific to the modelled process can be included in the description. One such effect is the closing of the top of the trench during the process, leading to a smaller and smaller opening though which particles can pass to reach the bottom of the trench. Again, if the initial geometry is known, namely a trench, an analytical model can be used to find the deposition width r at each point on the initial surface [193]:

$$r(x_s) = \frac{R_{top}}{2} \int_0^{t_{total}} dt \int_{\theta_{start}}^{\theta_{end}} \cos(\theta) d\theta \quad , \tag{5.2}$$

where θ is the angle from the current initial point on a sidewall, R_{top} is the deposition thickness at the top surface and t_{total} is the total time of the processing step. The integrals in this equation can be solved numerically using the forward Euler method which is highly efficient and robust, while providing sufficiently accurate results for this application. The relevant values are depicted in Fig. 5.2a, showing how the closing trench influences the deposition distance at the sidewalls. As the closing rate of the top of the trench can be found analytically, the deposition width at all points of the trench can be also be found analytically, without the need for computationally expensive ray tracing methods. The final 3D geometry is shown in Fig. 5.2b, where the deposited material on top of the trench can be seen to behave differently to the simple model shown in Fig. 5.1b. In this more sophisticated model, the smallest opening of the trench actually occurs high above the initial top surface since the deposited material builds up while growing to the inside. This crucial behaviour could not be modelled using the simple approach discussed in the previous section as deposition was assumed to be constant in time, which is certainly not the case for this type of CVD process. Hence, the time dependence of the deposition and the resulting change in particle traversal through the feature scale region must be taken into account in order to achieve satisfactory results.



(a) Calculation of the deposition thickness r on the side walls. First, R_{top} is found using the top visibility, which is then used to find the deposition width r of points on the side wall.

(b) Slice of the geometry resulting from solving Eq. (5.2) in time using numerical solutions for the time-dependent view factor of a closing trench. The more complex shape of the opening due to the strongly changing view factor is modelled appropriately.

Figure 5.2: Resulting geometry of the geometric advection of a CVD process.

The model presented above requires some knowledge about the initial structure and it can therefore not be applied universally. In order to allow for a full physical description of the process, particle transport and surface chemistry must be modelled. This way, arbitrary initial geometries can be simulated without further considerations, as the model describes the inherent physical nature of the fabrication process without any assumptions about the simulated geometry. A polysilicon deposition process can be modelled straight-forwardly using the modelling framework presented in Section 4.5.1. A two-particle model, similar to the one used in [188], was developed to form an accurate description of the deposition process using TEOS as the feed gas.

From experiments and sophisticated chemical kinetics simulations [183], it has been established that two precursors in the gas phase are responsible for the deposition of polycrystalline silicon (poly-Si) on the substrate. Our physical model involves the simulation of two different types of particles traversing the feature scale region, one representing TEOS with a low sticking probability of around 10^{-4} and one representing the highly reactive precursor triethoxysilane with a sticking probability close to unity ≈ 1. This means that there is a strong isotropic growth component through TEOS with a low sticking probability, while anisotropy is generated by the deposition through triethoxysilane. All other modelling parameters are taken from [183], with the simplifying assumption that particles remain on the surface once captured, are not re-emitted, and do not diffuse along the surface. Therefore, the deposition rate is directly proportional to the coverage of the respective particle, similar to the model presented in Eq. (3.17).

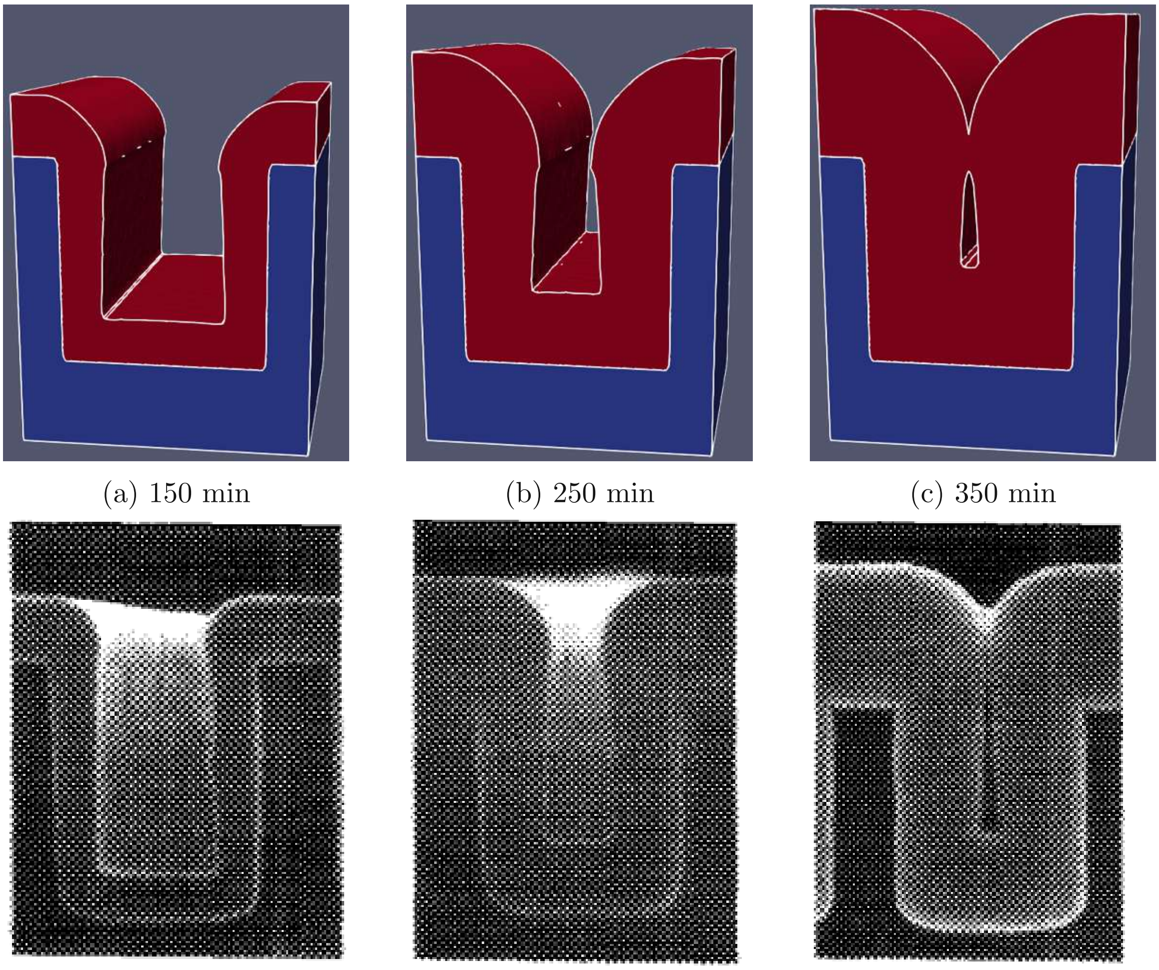
The resulting geometries are depicted in Figs. 5.3a to 5.3c and are compared to the experimental data from [188] in Figs. 5.3d to 5.3f. As can be seen clearly, the main features are replicated well for each time step using the sticking probabilities listed above. The overall deposition is strongly dominated by the isotropic growth from TEOS with a very small anisotropy introduced by the deposition of triethoxysilane. Using our model, solving for the deposition rates on top of the trench reveals that the deposition rate of the reactive species contributes roughly 37% of the total deposition rate. Due to the rapidly changing rate for triethoxysilane this results in a clear pinch-off of the trench even for the low aspect ratio used here.

Similarly to the actual process, the model must be tuned to the specific process parameters, such as temperature or pressure. Therefore, any discrepancy with a real process could be incorporated by understanding the physical origin of the behaviour and modelling it in the simulation. Hence, any process can be modelled using this physical approach, with the only disadvantage being long simulation times due to the computational expense of including additional physical effects and the computationally demanding Monte Carlo approach.

5.1.2 Epitaxial Growth

Epitaxy is a film growth process during which mono-crystalline layers are formed on top of a crystal substrate [194], which means that the newly grown layer only has one well-defined crystal orientation with respect to the substrate. Epitaxial silicon is usually grown using vapour-phase epitaxy, which is a type of CVD, as the deposited material is introduced to the substrate using precursor gases. In semiconductor fabrication, one of the main applications for epitaxial growth is the deposition of crystalline silicon on top of a wafer substrate of the same material, referred to as homoepitaxy [195].

If a different material to the substrate is grown epitaxially, this process is referred to as heteroepitaxy [196]. This results in a crystal mismatch at the interface between



(a) 150 min (b) 250 min (c) 350 min

(d) 150 min (e) 250 min (f) 350 min

Figure 5.3: Polysilicon deposition profiles after 150, 250 and 350 minutes from the simulation using the two precursor model (a)-(c), compared to experimental results (d)-(f) obtained from [188]. Reprinted with permission from [188]. Copyright 1993, American Vacuum Society.

the two different materials, leading to stress inside the material, which may also be temperature dependent [197].

On the atomic scale, epitaxial growth is driven by several physical processes. First, atoms from the gas phase are adsorbed onto the substrate and are loosely bound. These adatoms may now diffuse along the surface, remaining longer at energetically favourable surface sites. Therefore, clusters of adsorbed particles form on the substrate, serving as nucleation sites for other adsorbed particles, as the edges of these clusters form energetically favourable surface sites. As several clusters grow together, they form larger and larger islands, until they coalesce into a film covering the substrate [198]. Due to the fact that clusters of particles form to be energetically compatible with the substrate, they have the same ordering as the crystal they grow on.

5.1.2.1 Empirical Model

Since epitaxial growth is strongly dependent on the atomistic description of the surface, it cannot be described exhaustively using the continuum approach employed in this work. However, the macroscopic effect, namely the growth rates in different crystal directions can be measured experimentally [199] and used for the velocity fields describing the growth of the substrate. Usually, the growth rate is only measured along certain crystal directions, so the growth rates have to be interpolated for all other possible normal directions on the surface. This can be done robustly using the interpolation method presented in [200], which has already been successfully employed in [100, 201]. Assuming that the transport of molecules to the surface does not change drastically across the substrate, the velocity field only depends on the orientation of the surface. Hence, the growth rate can be found given only the crystal orientation of the substrate and the surface normals at each point on the surface.

The exact shape resulting from epitaxial growth depends strongly on the initial geometry and the temporal evolution of the surface. As discussed in [100], whether the fast or slow growth planes will dominate is strongly dependent on the curvature of the initial geometry. For complex shapes of the substrate, the growth might therefore change between fast and slow planes over time and thus a general prediction for which crystal plane will dominate growth is not possible. Due to this temporal dependence of the growth process, it is not possible to formulate a general geometric advection distribution and hence the emulation model has to be carried out using iterative advection. As the growth rates change drastically with the surface normal, the Stencil Local Lax-Friedrichs (SLLF) scheme discussed in Section 2.4.2.4 is used for the iterative advection of the growing substrate.

One of the most critical fabrication steps for modern vertical transistors is the epitaxy of the source and drain (S/D) contacts. Using a stacked nanowire field-effect transistor (FET) geometry [202], the model was applied to simulate the formation of the source and drain contacts, resulting in the characteristic shapes produced by fast growing crystal planes shown in Fig. 5.4.

5.1.2.2 Physical Model

Although continuum models describing epitaxial growth exist [203, 204], they are usually used to examine the fundamental physical processes driving growth rather than the simulation of large structures during fabrication processes. Therefore, the empirical model presented above is used for the description of the surface chemistry, while the necessary particle fluxes are generated using a Monte Carlo (MC) ray tracing approach.

Epitaxial growth of silicon is usually carried out in CVD reactors under specific conditions which allow for crystalline growth. A commonly used precursor is dichlorosilane ($Si_2H_2Cl_2$) which reacts with the surface to deposit Si forming hydrogen chloride (HCl) as a by-product [205]. The latter can then act as an etching species, slowing the silicon growth rate. However, HCl is required to achieve selectivity of the process, as silicon also deposits on other materials, forming poly-Si films. Due to

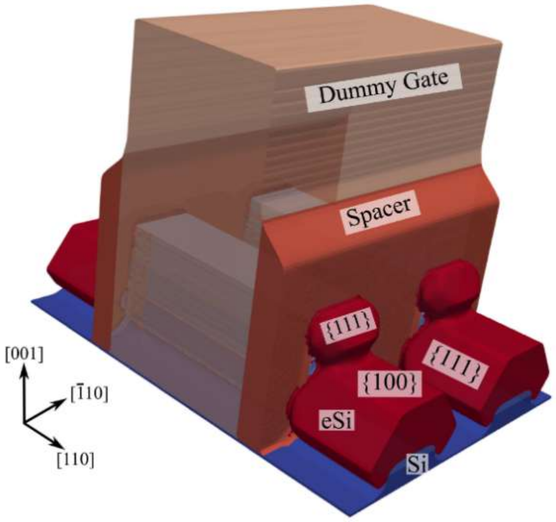


Figure 5.4: Geometry resulting from the growth of epitaxial silicon (red) for the S/D regions on top of a silicon substrate (blue). The crystal planes for the characteristic facets are indicated using Miller indices. CC BY 4.0 [100]

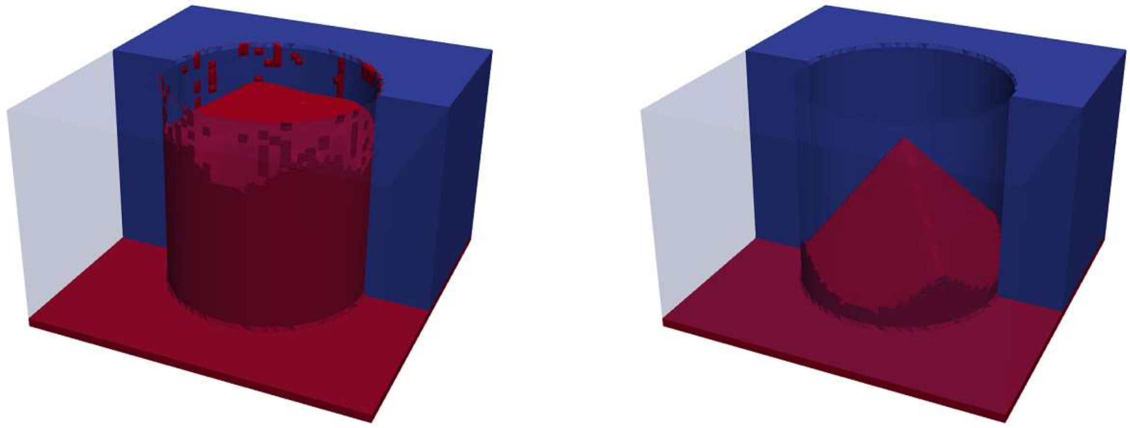
the lower binding energy of poly-Si to mask materials, compared to the binding energy of crystalline silicon, HCl can remove these unwanted films without etching the crystalline substrate. Therefore, finding the fine balance between the gases which are present in the reactor is critical to achieving selective growth only on the crystalline silicon substrate.

The chemical model was set up as described in [205] and selective epitaxial growth (SEG) on the bottom of a 60 nm wide via was simulated for 140 s. A lack of etchant leads to the unwanted deposition of poly-Si on top of other materials, as shown in Fig. 5.5a. Since no crystalline growth planes are exposed, the deposition of silicon proceeds in an unordered fashion. Only if the correct balance of depositing species and etchant is achieved, does a crystalline silicon film form on top of the substrate and no other materials are affected, as shown in Fig. 5.5b.

5.1.3 Anisotropic Wet Etching

Wet etching is one of the corner stones of semiconductor fabrication, especially in the field of microelectromechanical systems (MEMS). Etchants are introduced to the material surface using a liquid solution, which is why this process is called wet etching. Wet etchants usually remove material isotropically, due to the fast and isotropic transport of etchant species to the surface, meaning that every part of the surface is submerged and therefore etched by the solution. For non-crystalline or poly-crystalline substrates, the etching thus proceeds isotropically.

However, crystalline substrates may be etched at very different rates depending on the exposed crystal planes, as certain orientations lead to a more or less favourable bond of surface atoms with the bulk material, meaning that atoms of one crystal plane may be removed more easily than atoms on a different plane [206]. Therefore,



(a) Not enough etchant available to clean the mask surface.

(b) Etchant and silicon precursor in balance.

Figure 5.5: SEG of silicon at the bottom of a via using dichlorosilane as a precursor. a) If there is not enough HCl available to clean the other materials, silicon will be deposited everywhere. b) Careful tuning of the feed gases results in crystalline growth only on the silicon substrate.

crystalline silicon substrates can be etched anisotropically, leading to well defined crystal faces dominating the final geometry.

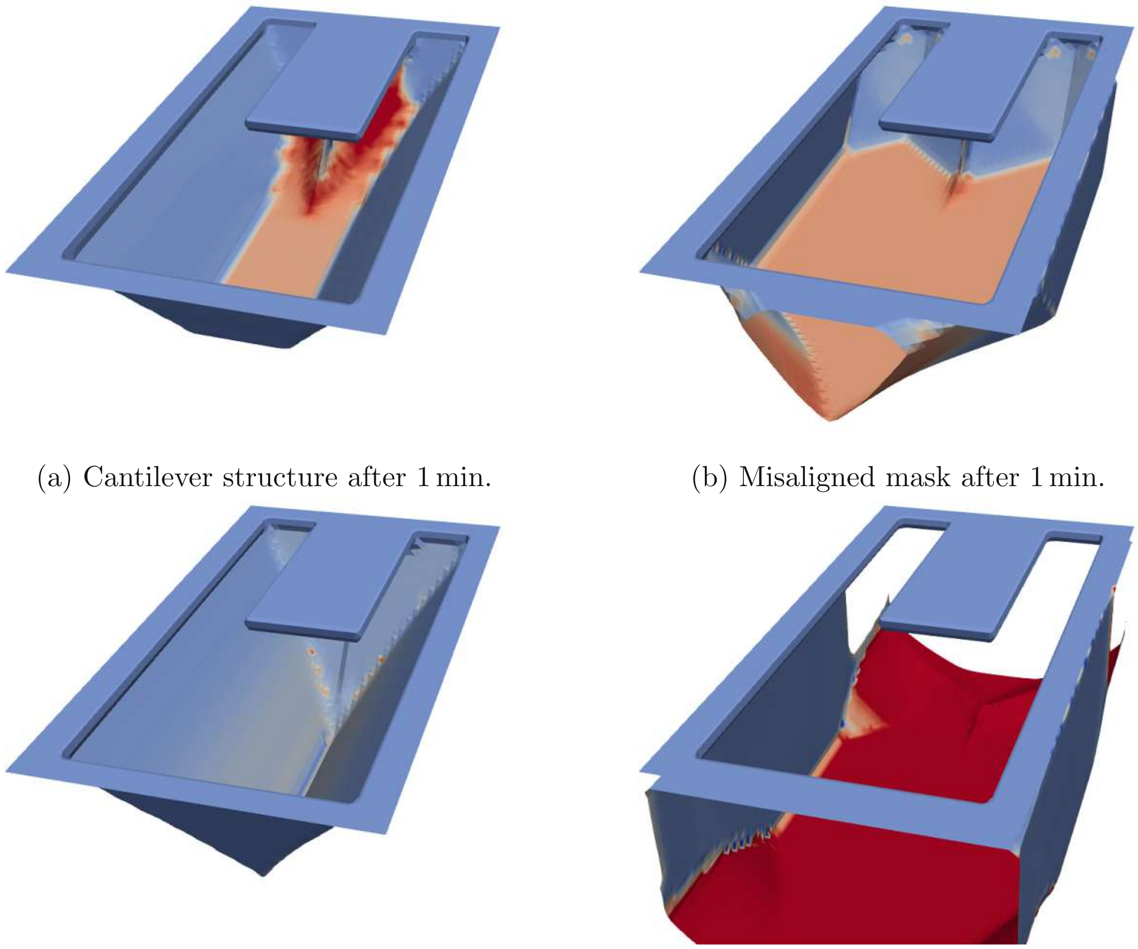
Similarly to SEG, wet etching can be modelled by considering the etch rates in certain crystal directions and interpolating to all other surface orientations. Additionally, since the process is taking place in a liquid, the etchant transport to the surface is much faster than the etch reaction, meaning the process is only reaction rate limited and there is no need to model particle transport as for SEG. Therefore, an empirical model using empirically determined rates results in a very accurate description and no additional modelling is necessary.

Fig. 5.6 shows the result of a wet etching simulation for a MEMS cantilever, using the same interpolation scheme [200] for the etch rates as used for the modelling of SEG described in Section 5.1.2.

The etch rates for different crystal orientations and temperatures are found in literature [207] and the rates passed to the interpolation scheme simply have to be adapted to the corresponding etch chemistry. Since the etch rates are strongly dependent on the crystal direction, the wafer has to be aligned carefully in order to achieve the desired cantilever structure depicted in Fig. 5.6c. If the wafer is rotated by 45 degrees, an entirely different geometry is produced, as shown in Fig. 5.6d.

5.1.4 Physical Plasma Etch Models

In modern semiconductor fabrication, wet etching processes have generally fallen out of favour due to additional cleaning steps which are required to remove residues left on the wafer. Additionally, wet etching processes do not provide enough control over the anisotropy of the etched features for high aspect ratio structures or 3D



(a) Cantilever structure after 1 min.

(b) Misaligned mask after 1 min.

(c) Correct cantilever after 2 min.

(d) Broken final structure after 2 min due to misaligned mask.

Figure 5.6: Wet etching of differently oriented silicon substrates with the same mask, resulting in entirely different geometries. The colours hint at the etch rate at every point on the surface, where blue corresponds to a low etch rate and red to a high one.

device geometries [208]. Therefore, most process flows in industrial fabrication of semiconductor devices rely heavily on dry plasma etch processes [209].

Usually, physical etching proceeds through ion-enhanced etching of volatile chemical reactants on the surface of the substrate, as described in Section 3.2.5. However, other mechanisms may also contribute to the etching properties of a process, such as ion energy, ion distribution and the exact surface chemistry. Therefore, in order to develop a reliable plasma etch model, the dominant physical processes have to be identified and their properties carefully tuned to achieve an appropriate description of the underlying physical behaviour.

Silicon is the most important material in semiconductor fabrication, so there is an abundance of etching chemistries developed for the patterning of silicon [210, 211]. Therefore, the chemistries presented in the following section, are mostly concerned with the etching of silicon substrates. Nonetheless, the properties of some of these processes are also discussed for different substrates of interest, such as titanium nitride (TiN) and hafnium dioxide (HfO_2) which are commonly encountered in modern FET geometries [212].

5.1.4.1 Chlorine Plasma Etching

Chlorine has been used to etch silicon [213] and several of its compounds, such as silicon dioxide (SiO_2) [214], silicon nitride (SiN) [215], as well as other commonly used materials such as TiN, tantalum nitride (TaN) [216] and HfO_2 [217].

Plasma etching in molecular or atomic chlorine chemistries proceeds through the adsorption of Cl on the silicon substrate and the formation of $SiCl_2$, which is then desorbed thermally [218] or via ion-enhanced etching of chlorine ions [219]. Several additional gases can be added to achieve different etch properties. The addition of argon [220] or nitrogen [221] as an ion source leads to higher etching yields, due to the increased number of energetic ions assisting in the removal of $SiCl_2$[222].

In the etching of TiN, chlorine may be combined with BCl_3 or CHF_3 [223], leading to non-volatile BN or TiF_4 being deposited on the side walls, respectively. These passivation layers protect the structure from lateral chemical etching through $TiCl_x$. Ion-enhanced etching and ion sputtering lead to high etch rates at the bottom of the substrate, leading to anisotropic and highly vertical profiles. These mechanisms for the etching of TiN in a BCl_3/Cl/Ar chemistry are depicted in Fig. 5.7.

When etching HfO_2, BCl_3 plasmas may even be tuned to achieve infinite selectivity to Si, since boron leads to the formation of a thick SiClB layer above the silicon substrate, protecting it from further etching [224]. Thus, BCl_3 plasmas are well suited for the removal of the thin HfO_2 dielectric layer in modern high-k metal gate FETs [225].

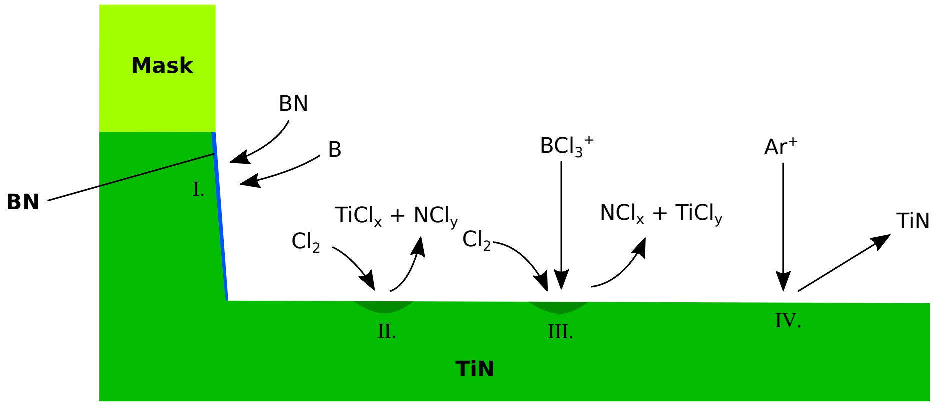


Figure 5.7: Active etching and deposition mechanics in Cl chemistries used to etch TiN: I. Chemical deposition of BN, II. Chemical etching through $TiCl_x$ and NCl_y radicals, III. Ion-enhanced etching and IV. Ion sputtering through high energy ions.

5.1.4.2 Fluorocarbon Plasma Etching

For decades, fluorocarbon (CF) chemistries have been used to etch Si and SiO_2 due to the possibility of tuning these chemistries to the etching of different materials with fine control over the selectivity to other materials [226]. In order to adjust the process for specific substrates, a variety of additive gases can be used to change the characteristics of the process [227]. Hence, good etch selectivity can be achieved for certain materials, without infringing on the anisotropic nature of the process [228].

The main etch mechanisms in CF plasmas are chemical etching, ion-enhanced etching and physical sputtering. In the case of silicon, the substrate can be removed chemically by reacting with fluoride to form silicon tetrafluoride (SiF_4) which can evaporate back to the gas phase [229], as shown in Fig. 5.8. The rate at which chemical etching proceeds is strongly dependent on the temperature and the amount of carbon on the surface, which may lead to a reduction in etch rates.

Another etch mechanism is the ion-enhanced etching of the substrate, where highly reactive CF^+ ions bombard the surface. The substrate is either sputtered by the ion, or the ion reacts with the substrate to form SiF_4 which may then evaporate [230]. The temperature should be low enough for SiF_4 not to evaporate thermally. Rather, the etch product would be removed by the additional kinetic energy provided by incoming ions. This results in highly anisotropic etching, since substrate can only be removed through energetic ions. However, the ion energy should be as small as possible since ions may otherwise penetrate deep into the substrate, creating impurities. Hence, there is an optimal combination of process temperature and ion energy which leads to highly anisotropic etching with minimal ion-induced damage in the substrate.

Physical sputtering is strongly related to the binding energy of the substrate and appears only above a certain ion energy. The deposition of the passivation layers on the side walls is achieved through the polymerisation of neutral species, such as CF_2, which form SiC bonds protecting the substrate. Additionally, deposition may also be driven by direct absorption of energetic ions into the substrate [231].

All of the above discussed etching mechanisms can be described using the general plasma etching model presented in Section 3.2.5. In order to represent this particular chemistry, supplying textbook values for particle and substrate properties is sufficient.

CF plasma chemistries may also be used to etch TiN, although high etch rates can only be achieved at high temperatures, which is why chlorine-based chemistries are commonly used instead [232]. However, the addition of small amounts of CF has been shown to increase the etch rate in chlorine chemistries due to the additional generation of volatile etch reactants in the gas phase [233].

HfO_2 which is commonly used as a dielectric, may also be etched in CF plasmas with good selectivity and reasonable etch rates [234]. However, CF chemistries form thick fluorocarbon layers which can be omitted by adding hydrogen as a feed gas [235]. The additional hydrogen will form loosely bound hydrocarbon etch products [236], leading to a significant reduction in carbon contamination. Nonetheless, as HfO_2 is often used as an atomically thin dielectric, even small amounts of fluorocarbon residue present a challenge [237], which is why CF chemistries have fallen out of favour when etching HfO_2.

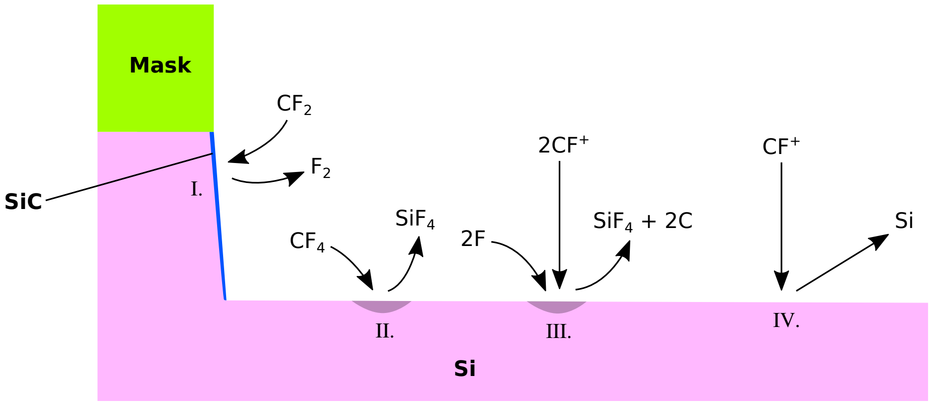


Figure 5.8: Active etching and deposition mechanics in CF type chemistries used to etch poly-Si: I. Chemical deposition of carbon forming an SiC passivation layer, II. Chemical etching, III. Ion-enhanced etching and IV. Ion sputtering through high energy ions.

5.1.4.3 Sulphur Hexafluoride Plasma Etching

A commonly used alternative to CF chemistries is sulphur hexafluoride (SF_6) since it allows for finer control over the etch properties through additional gases fed into the reactor while still achieving high etch rates [238]. Without any additional feed gases, pure SF_6 dry etching of silicon proceeds isotropically, which can be useful when creating undercuts as it saves the additional cleaning step required after isotropic wet etching [239]. Adding oxygen results in the formation of a thin SiO_2 layer being formed on the sidewalls inhibiting lateral etching [240]. The addition of oxygen also leads to higher vertical etch rates, since it binds sulphur, freeing more fluoride which is then available for etching [241]. The oxygen concentration must be tuned carefully, as high concentrations lead to competing surface adsorption with fluoride, forming thick SiO_2 protective layers and thus reducing the etch rate [242]. If the oxygen concentration is optimal, only thin SiO_2 layers are formed, which are removed on horizontal surfaces through directional energetic ions. Therefore, fluoride atoms can react with the silicon substrate, forming SiF_4 which is removed chemically or through ion-enhanced processes [243], similarly to the reactions shown in Fig. 5.8.

The properties of SF_6 etching can be tuned through the addition of hydrogen bromide (HBr) and oxygen as feed gases. This leads to the formation of a SiO_xBr_y passivation layer reflecting high energy ions and thus leading to higher vertical etch rates [244]. Therefore, through the introduction of additional feed gases, the behaviour of an etch process might change drastically and a new model describing this behaviour may have to be developed.

Other additional passivating species, such as difluoromethane (CH_2F_2), have also been successfully applied for the dry etching of silicon [245]. As shown in Fig. 5.9, the build up of the passivation layer on the sidewall is not created through the deposition from the gas phase, but rather through line of sight deposition of sputtered CF etch products obtained from vertical etching [246]. Therefore, the modelling of this process requires an additional ray tracing step, where sputtered etch products are emitted from the collision site of an energetic ion and traced to the sidewall where they might deposit. The thereby created shadowing effects can only be observed with this additional ray tracing step, meaning that this model may incur more computational effort than simpler models. Hence, depending on the modelled process, even the underlying methods used to simulate the model may have to be adjusted.

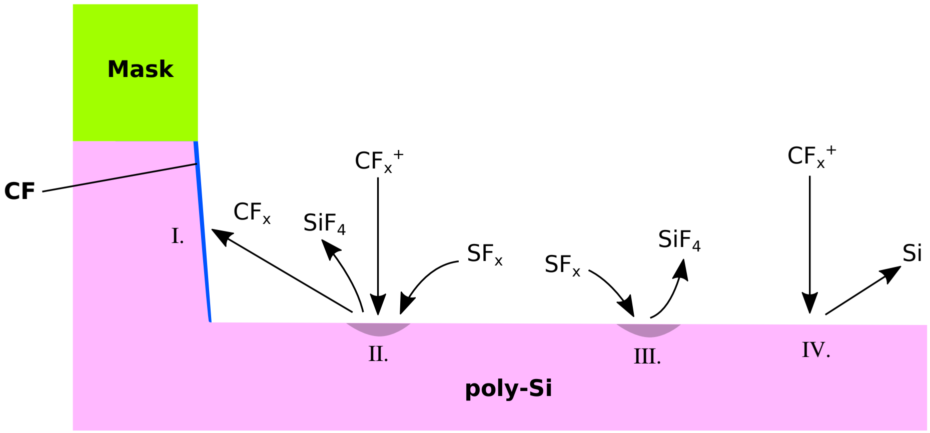


Figure 5.9: Sulphur with Fluoride (SF) type etching and deposition mechanics with additional CH_2F_2 feed gas. I. Line of sight deposition of a CF passivation layer; II. ion-enhanced etching; III. chemical etching; and IV. physical ion sputtering.

The described chemistry was implemented using a physical model employing MC ray tracing. The chemical parameters of the involved materials were extracted from [243] and [244]. The model was characterised by simulating the etching of a trench in a silicon substrate for different source fluxes. Fig. 5.10 shows the behaviour of the implemented plasma etch model for these different processing conditions, hinting towards the behaviour of the etch chemistry based on its feed gases. A higher etchant flux thus results in higher etch rates, but a fine balance with the passivating species is required to achieve satisfactory profiles.

Similarly to fluorocarbon plasmas, SF_6 chemistries require high temperatures to etch TiN, although its use as a supplementary feed gas for chlorine chemistries can increase etch rates significantly [233]. Although high etch rates of HfO_2 substrates have been achieved in SF_6 chemistries, they are limited to low pressures for the appropriate tuning of the selectivity to silicon [234].

5.1.4.4 Hydrogen Bromine Plasma Etching

HBr chemistries have been in use for a long time to etch silicon for its good selectivity towards SiO_2 [130, 247]. Despite the lower silicon etch rate compared to CF and SF_6 chemistries, HBr provides excellent selectivity towards other materials, which makes it suitable for the over-etch processing step, used to entirely remove silicon on top of other materials [248]. Ion-enhanced etching typically dominates the removal of the substrate, while the deposition of side wall passivation proceeds mainly chemically [244]. Therefore, simple models can be used to describe the process, as shown in

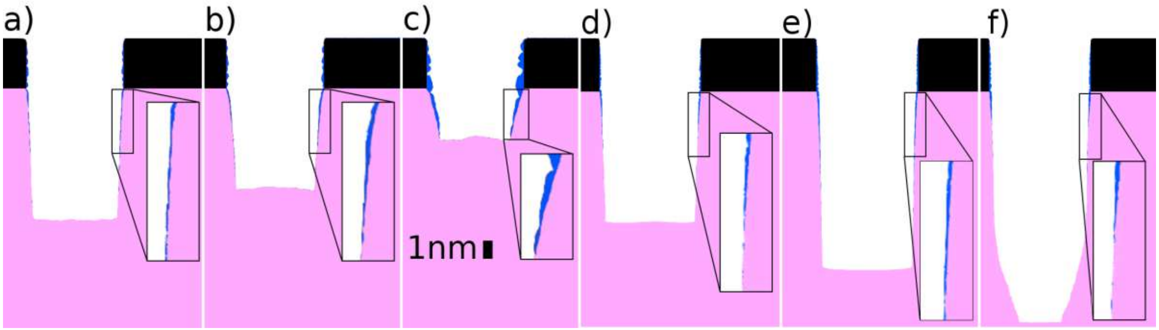


Figure 5.10: Two-dimensional trenches formed by etching silicon (pink) with a mask (black). The ion flux was kept constant at 10^{16}cm^{-2}s^{-1}. In a)-c), the trenches were etched for 25 s with a constant etchant flux of 1.3x10^{16}cm^{-2}s^{-1}, while the polymer (blue) concentration was varied: a)5x10^{15}cm^{-2}s^{-1}, b)10^{16}cm^{-2}s^{-1} and c) 5x10^{16} cm^{-2}s^{-1}. In d)-f), the polymer flux was constant at 5x10^{15}cm^{-2}s^{-1} and the etchant flux was changed: d)5x10^{15}cm^{-2}s^{-1}, e)2x10^{16}cm^{-2}s^{-1} and f) 5x10^{16} cm^{-2}s^{-1}.

Fig. 5.11. The side walls and substrate are protected by thick SiO_xBr_y layers stopping energetic ions from reaching the material below.

Despite the apparent simplicity of the process, the passivation layer undergoes a change during the process, as bromine is removed from the layer and replaced by oxygen. Therefore, a dense SiO_2 layer is generated at passivation layers formed earlier, while an amorphous bromine rich material dominates the passivation layers formed later in the process [249]. This densening of the passivation layers can only be modelled properly using volume information, meaning a pure level set description is not sufficient to capture the process appropriately. It may be approximated by storing the bromine concentration of the material on the surface and reducing it through the thinning of the material, although this may be insufficient for complex geometries. Oftentimes, etching in HBr plasmas is followed by a cleaning step which reduces this layer [250], meaning that it is changed to a dense SiO_2 layer everywhere.

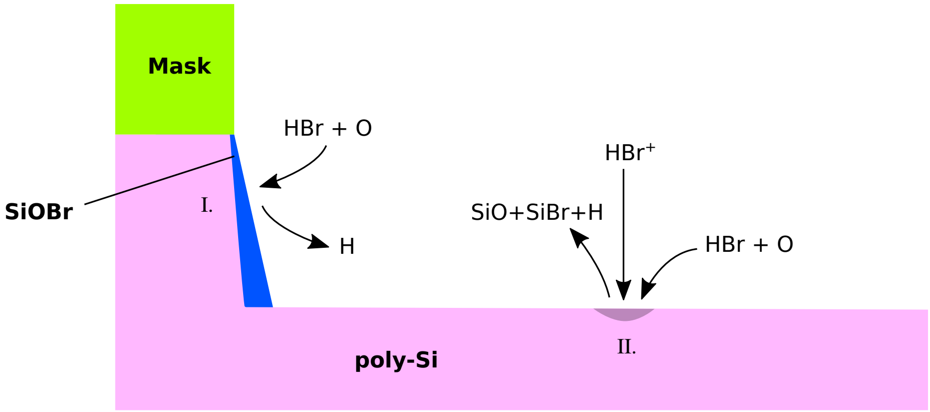


Figure 5.11: Dominant etch mechanics during silicon etching using hydrogen bromide. I. Sidewall passivation proceeds though chemical deposition from the gas phase and II. Vertical etching is dominated by ion-enhanced etching.

A model for this etch chemistry was implemented in ViennaPS using the physical parameters in [249]. Fig. 5.12 shows the profile of a trench etched in a silicon substrate, resulting from HBr etching for different source fluxes of all active species.

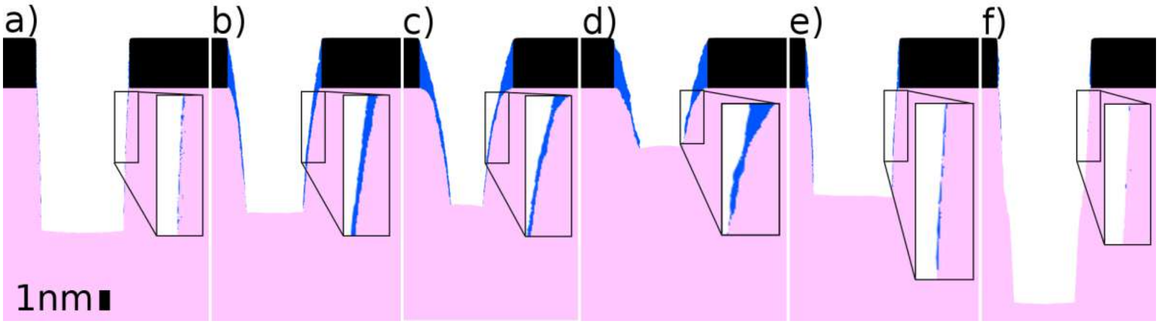


Figure 5.12: HBr/O_2 etching of Silicon (pink) with a mask (black) for 25 s. The ion flux was kept constant at 10^{16}cm^{-2}s^{-1}. In a)-c), the Si was etched with a constant etchant flux of 10^{16}cm^{-2}s^{-1}, while the polymer (blue) concentration was varied: a)10^{16}cm^{-2}s^{-1}, b)3x10^{16}cm^{-2}s^{-1} and c) 5x10^{16} cm^{-2}s^{-1}. In d)-f), the polymer flux was constant at 10^{16}cm^{-2}s^{-1} and the etchant flux was changed: d)10^{15}cm^{-2}s^{-1}, e)5x10^{15}cm^{-2}s^{-1} and f) 5x10^{16} cm^{-2}s^{-1}.

HBr is also well suited for the etching of TiN, achieving satisfactory etch rates [251] and clean final surfaces [237]. However, bromine etches TiN significantly slower than chlorine, although the former can be added in small concentrations to the latter in order to obtain better overall control over etch properties [225].

5.1.5 Gate Stack Etching Sequence

In order to highlight the simulation capabilities of the presented physical models, a full gate stack etching sequence of a 14 nm gate-first fabrication sequence [208] was simulated using the modelling framework developed during the course of this work. Due to the physical descriptions, not only each process individually, but also the interplay between sequential steps is taken into account. This is especially important when considering different passivation layers building up in each of the etch steps [110].

Poly-Si Main Etch

The first etch step is the poly-Si main etch in a SF_6 type plasma chemistry. The model presented in Section 5.1.4.3 was used to simulate this etch step for 60 s with the model parameters listed in Table 5.1. The geometry of the gate stack after this etch step is shown in Fig. 5.13a. The slanted side wall stems from the deposition of polymer (blue) layers on the poly-Si (pink), which prevents lateral etching. As expected, the polymer layers are thicker at the top of the structure than at the bottom, since the top side walls were exposed earlier, leaving more time for the polymer to grow.

Poly-Si Over Etch

After the main etch, a poly-Si over etch step is carried out in a more selective $HBrO_2$ plasma chemistry in order not to damage the TiN underneath. As depicted in Fig. 5.13b, due to the deposition of a several nm thick SiBr-type polymer, the side walls are strongly slanted. Thanks to the selectivity of the etch process, the gate metal underneath the poly-Si is left intact without damage.

TiN Main and Over Etch

The TiN is subsequently etched in a Cl_2CH_4 plasma chemistry, which has a lateral to vertical etch ratio of about 0.4, which leads to a visible under etch in the TiN layer. The high lateral etch rate is due to the small amount of passivating polymer deposited on the sidewalls, as can be seen in Fig. 5.13c. This wears down the previously deposited passivation layers, which now play an important role in protecting the poly-Si from the etchant.

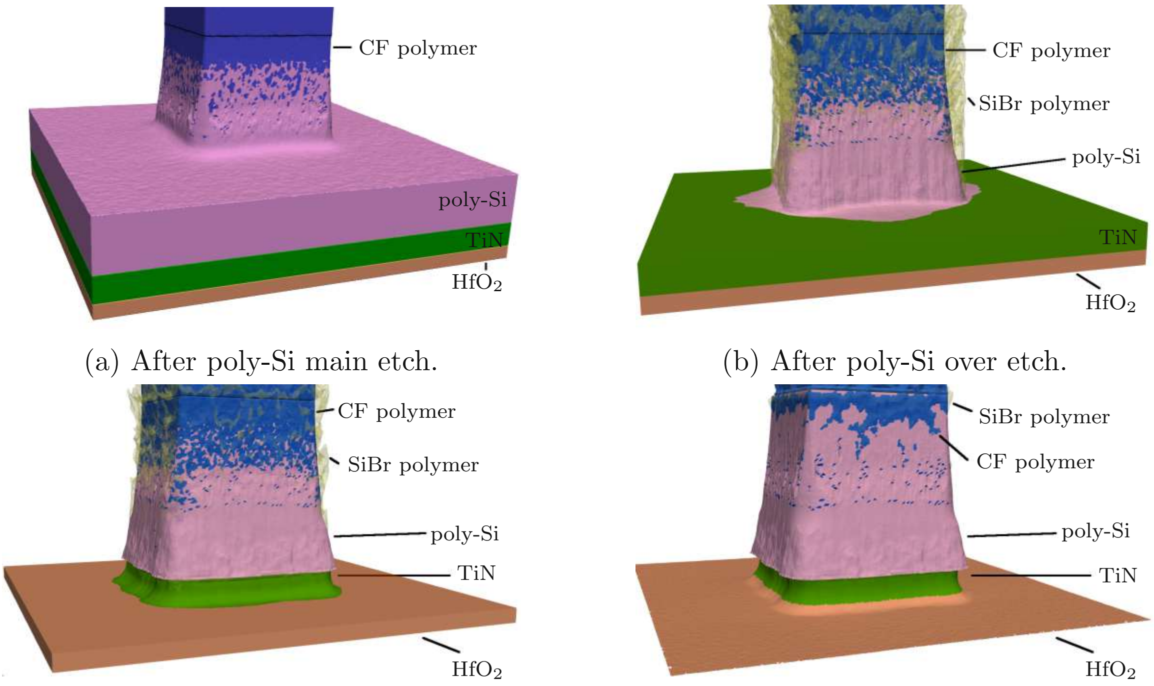
In order not to damage the thin high-k dielectric HfO_2 layer underneath the TiN, a more selective etch process is used for the over etch step. The etching in a Cl_2/N_2 chemistry proceeds highly isotropically and appropriately cleans the HfO_2 surface for the last etch step, as shown in Fig. 5.13c.

HfO_2 Etch

The HfO_2 layer which is only a few nanometres thick, is then etched in a BCl_3/Cl_2 plasma. This etch chemistry also proceeds highly isotropically and has a near-infinite selectivity against the SiO_2 substrate underneath the gate dielectric [224]. The final structure after this etch step is visualised in Fig. 5.13d, which highlights the importance of the previously deposited passivation layers to protect the masked gate stack from etching in the subsequent, more isotropic etch steps.

Process Step	Model	Ion Flux	Etchant Flux	Polymer Flux
Poly-Si ME	SF_6/CH_2F_2	$1.5 \cdot 10^{15}$	$1.3 \cdot 10^{16}$	$4.5 \cdot 10^{15}$
Poly-Si OE	HBr/O_2	$1.0 \cdot 10^{16}$	$1.0 \cdot 10^{16}$	$2.0 \cdot 10^{16}$
TiN ME	Cl_2CH_4	$1.0 \cdot 10^{15}$	$5.0 \cdot 10^{15}$	$2.8 \cdot 10^{16}$
TiN OE	Cl_2/N_2	-	$1.0 \cdot 10^{17}$	$1.0 \cdot 10^{16}$
HfO_2 Etch	BCl_3/Cl_2	-	$1.1 \cdot 10^{16}$	$6.2 \cdot 10^{15}$

Table 5.1: Model parameters of the physical models used to simulate the gate stack etching sequence with fluxes given in units of $cm^{-2}\,s^{-1}$. Main etch and over etch chemistries are indicated as ME and OE, respectively. All models were simulated using a pressure of 1.35 Pa with a bias voltage of 60 V for the plasma etch steps.



(a) After poly-Si main etch.

(b) After poly-Si over etch.

(c) After TiN main and over etch.

(d) After HfO$_2$ etch.

Figure 5.13: Gate stack geometry after important patterning steps, highlighting the importance of passivation layers throughout the process. The thick, protective layers formed during the poly-Si etch steps protect the material throughout the process.

5.1.6 Bosch Process

The cyclic Bosch Process [252] was developed in 1994 for generating high aspect-ratio structures in the field of MEMS [253, 254]. Since then, this fabrication technique has become important for numerous other applications, such as 3D integration [255, 256], high bandwidth memory [257, 258] and even on-chip wireless communication [259].

This process is a type of deep reactive ion etching (DRIE), where sidewall passivation and etching do not proceed simultaneously but are separated into individual steps carried out sequentially. By repeating these sequential process steps many times, aspect ratios far higher than those achievable using conventional plasma etching approaches have been achieved [260]. The original approach will be discussed in the following section, followed by further developments through the introduction of additional processing steps.

5.1.6.1 DEM Sequence

This sequence of process steps was used when the Bosch Process was first introduced. It consists of a deposition step followed by a plasma etching step, which is why this approach is also referred to as the deposit-etch-multiple-times (DEM) sequence. Since the etch step is not perfectly anisotropic, a scallop is formed during each cycle of the process, as depicted in Fig. 5.14.

However, robust sidewall passivation and very straight profiles can be achieved reliably, reducing sidewall tapering significantly compared to other plasma etching processes. However, when the etched structures reach aspect ratios higher than 50, reactive ion etching (RIE) lag is observed [261, 262]. Due to a smaller and smaller number of etchant particles and ions reaching the bottom of the structure, the etch rate decreases significantly until etching is perfectly balanced with the deposition step at the bottom of the generated structure. This effect can be reduced by increasing etch times for later cycles, but due to the prolonged etch step, the mask or top of the structures can be worn away quickly [263].

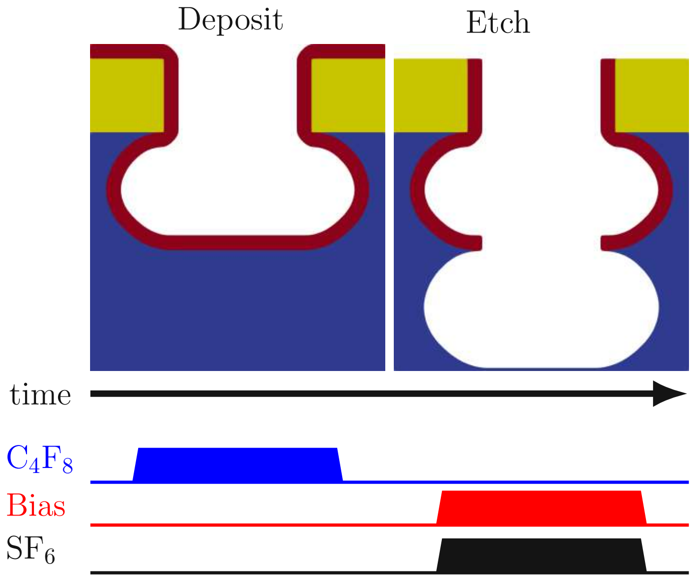


Figure 5.14: One cycle of the DEM sequence to generate high aspect-ratio structures with the chemistries used for each step. For clarity, the second cycle of the sequence is shown.

5.1.6.2 DREM Sequence

Due to the problems encountered during the etch step of the DEM sequence for high aspect ratios, this step can be split once again [264]. In order to remove the protective layer at the bottom of the structure, a low pressure etching process is conducted at high plasma bias, resulting in highly anisotropic etching [260]. After this step, the exposed substrate is etched isotropically at low bias, as shown in Fig. 5.15. Due to the additional passivation removal step, this procedure is referred to as the deposit-remove-etch-multiple-times (DREM) sequence.

Due to the separation of the etch step, a shorter time under high bias is required, leading to higher mask selectivity [265]. For high aspect ratios, RIE lag can thus be countered by only increasing the isotropic etch time, which leads to uniform scallop sizes down the feature without damaging the mask [266].

However, if the deposition of the protective layer and its removal are not perfectly balanced, more polymer may be deposited on the top opening than necessary. Therefore, each cycle increases the thickness of the protective layer at the top, effectively closing off the opening. This increases shadowing and thereby reduces the number of high energy ions reaching the bottom of the structure to remove the protective layer [265], leading to a smaller hole in the deposited material at the bottom of the feature. Due to the time ramping of the isotropic etch step, the scallop size does not change drastically, but the smaller hole sizes are copied down, resulting in tapered side walls.

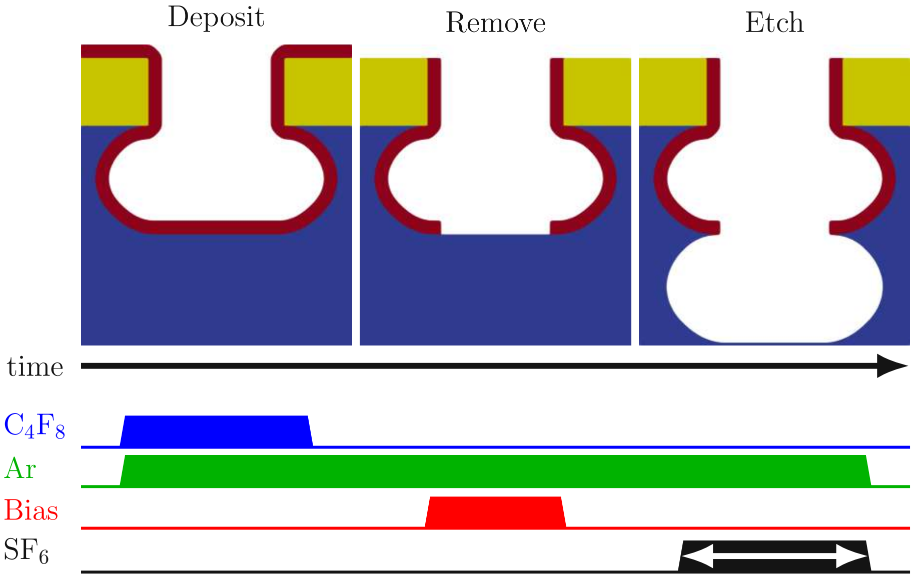


Figure 5.15: One cycles of the DREM sequence where the etch step of the DEM sequence is split into a directional remove and an isotropic etch step. The length of the etch step can be adjusted to counter RIE lag.

5.1.6.3 DREAM Sequence

As described in the previous section, the pile-up of polymer can only be avoided by carefully tuning the interplay of the deposition and removal of the protective layer. However, tuning these steps is time consuming and may require frequent calibration.

The buildup of polymer at the top opening may also be avoided by introducing an additional ashing step [267], which selectively removes the polymer, leading to a clean surface [268]. This is performed with the so-called deposit-remove-etch-ash-multiple-times (DREAM) sequence which is shown in Fig. 5.16. This method prevents the closing of the hole opening and the subsequent tapering without the need for frequent and precise tuning.

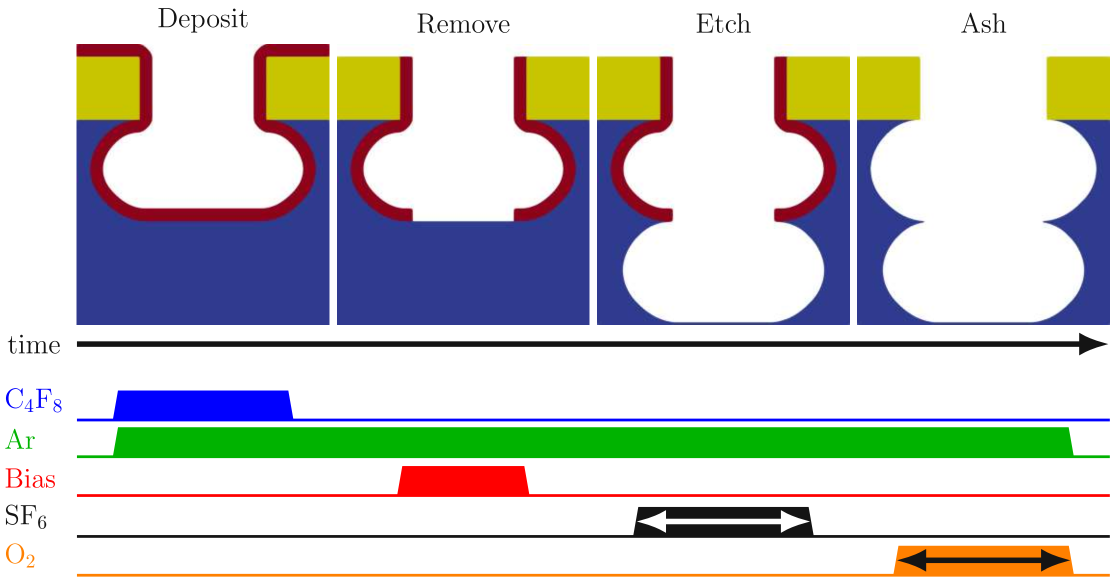


Figure 5.16: DREAM sequence with the additional ashing step compared to the DREM sequence. The length of the ash step needs to be adjusted according to the length of the deposition step in order to properly remove the piled up polymer at the top opening.

5.1.6.4 Empirical Model

Due to the complex final structure generated by the DRIE process, two separate geometric advection steps are necessary to create it:

1. Smooth Profile: The profile describing the outline of the high aspect-ratio feature is generated.

2. Scallops: The scallops produced by the cyclic nature of the process are formed, starting with the smooth profile.

Since the exact physical nature of how the structure was generated is not important in process emulations, these two steps suffice to generate the final structure. This is independent of the number of cycles that would be needed in the actual fabrication of the structure and is thus highly efficient.

Smooth Profile

Since the final profile will be highly directional, a rectangular geometric advection distribution is used to generate the smooth outline. The edge length of this distribution will be L, which is given by the number of cycles N_c times the etch depth per cycle d_c

$$L = N_c d_c \quad . \tag{5.3}$$

As discussed in the previous sections, decreasing etch rates may also lead to tapered side walls. Therefore, tapering is modelled by reducing the total etch depth L

according to the lateral distance to the mask. During DRIE, the side walls are usually tapered inwards, meaning the feature becomes smaller with increased etch depth. Therefore, the coordinate dependent etch depth $L(\vec{x})$ is smallest at the sidewall and increases with the lateral distance to masked regions.

Scallops

The smooth profile generated in the first step is the starting geometry for scallop creation. Since the scallops are produced by the isotropic etching of the substrate, they can be emulated using spherical geometric advection distributions. These distributions should only be placed at points, where the substrate was initially available for isotropic etching. Therefore, the radii of all geometric distributions are set to zero, except at the heights where scallops would be formed by the process. Therefore, ignoring tapering, the n^{th} scallop is formed at a height z_n given by

$$z_n = d_c \left(n + \frac{1}{2} \right) \quad \text{for } n \in \mathbb{Z} \quad . \tag{5.4}$$

The diameter of the spherical distributions centred at z_n is given by the isotropic etch distance per cycle d_{iso} which is closely related to the etch depth per cycle d_c. If the physical phenomena driving the etching are dominated by free molecular transport, d_c and d_{iso} are equal. However, if the etch process is mostly diffusion-limited, then d_{iso} is larger than d_c. Therefore, it is useful to express d_{iso} as a multiple of d_c:

$$d_{iso} = f_t d_c \quad , \tag{5.5}$$

where $1 \leq f_t \leq 2$. The difference between the two transport phenomena is shown in Fig. 5.17 for the extreme cases of perfect free molecular transport when $f = 1$ and perfect diffusion-limited transport when $f_t = 2$ in Section 5.1.6.4 and Section 5.1.6.4, respectively. Usually, a combination of both transport phenomena is observed and the common value of $f_t \approx 1.5$ for modern plasma etching processes [211, 267] is also shown in Section 5.1.6.4.

The spherical distributions discussed so far imply that etching is perfectly isotropic, meaning that the lateral and vertical etch rate are equal. However, this may not always be the case and, especially in modern SF_6 plasma chemistries, lateral etching is usually slower than vertical etching. In order to capture these differing etch distances, it is useful to express the lateral etch rate R_l as a fraction of the vertical etch rate R_v

$$R_l = f_l R_v \quad , \tag{5.6}$$

where it is assumed here that $0 \leq f_l \leq 1$ holds.

The geometric advection distribution, taking the anisotropic rates into account, resembles a spherical cap shape similar to a convex lens, as shown in Fig. 5.18, and will be referred to as lens distribution in the following. Graphically, this distribution is created by removing the centre section of a unit sphere in all lateral directions and moving the slices back to the origin. The centre section ranges from $-1(1 - f_l)$ to $(1 - f_l)$ in the lateral dimensions, as shown in Fig. 5.18a. The outer parts of the

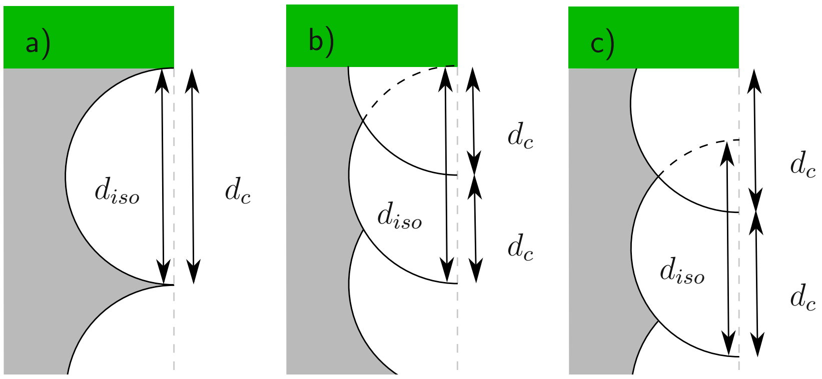


Figure 5.17: Scallops for different values of f_t: (a) $f_t = 1$ corresponding to perfect free molecular transport, (b) $f_t = 2$ corresponding to perfect diffusion-limited etching and (c) $f_t = 1.5$ corresponding to a combination of both transport phenomena. CC BY 4.0 [269]

sphere are then moved back to the origin and scaled by r_{lens} to achieve the original etch depth of d_{iso}. The correct scaling is achieved by defining r_{lens} as

$$r_{lens} = \frac{d_{iso}}{2} \begin{cases} 1 & \text{if} \quad f_l \geq 1 \\ (1 - f_l^2)^{-1/2} & \text{if} \quad 0 \leq f_l < 1 \\ \text{undefined} & \text{if} \quad f_l < 0 \end{cases} . \tag{5.7}$$

The lens distribution is then defined by adjusting the definition for the spherical distribution and is given by

$$\phi_{lens}(\vec{v}, f_l) = \frac{|\vec{v} + (1 - f_l) r_{lens} \overrightarrow{\text{sgn}}(\vec{v}_D)| - r_{lens}}{\Delta x} , \tag{5.8}$$

where $\vec{v}_D$ consists only of the lateral dimensions of the vector $\vec{v}$, meaning $\vec{v}_2 = (v_x, 0)$ and $\vec{v}_3 = (v_x, v_y, 0)$ for two and three dimensions, respectively. $\overrightarrow{\text{sgn}}(\vec{v})$ is the sign function of the vector $\vec{v}$. The additional term $(1 - f_l) r_{lens} \overrightarrow{\text{sgn}}(\vec{v}_D)$ as compared to Eq. (2.65) essentially shifts the distance vector $\vec{v}$ outside of the spherical distribution, creating the lens distribution shown in Fig. 5.18a.

This distribution can fully describe the scallops created by an anisotropic etch step, as those generated in modern Bosch process cycles. An example for such scallops using typical values is shown in Fig. 5.18b, showing a clear difference to the scallops generated by the spherical distributions shown in Fig. 5.17.

DEM Sequence

In high aspect ratio structures this sequence suffers from RIE lag, which means that the etch rate decreases with increasing aspect ratio [270], due to the depletion of etchant particles at the feature bottom [271].

Therefore, the feature will become smaller with increasing depth due to the positively tapered side walls. The decrease in etchant particle flux down the feature depends on the exact geometry and particle transport in the feature scale region.

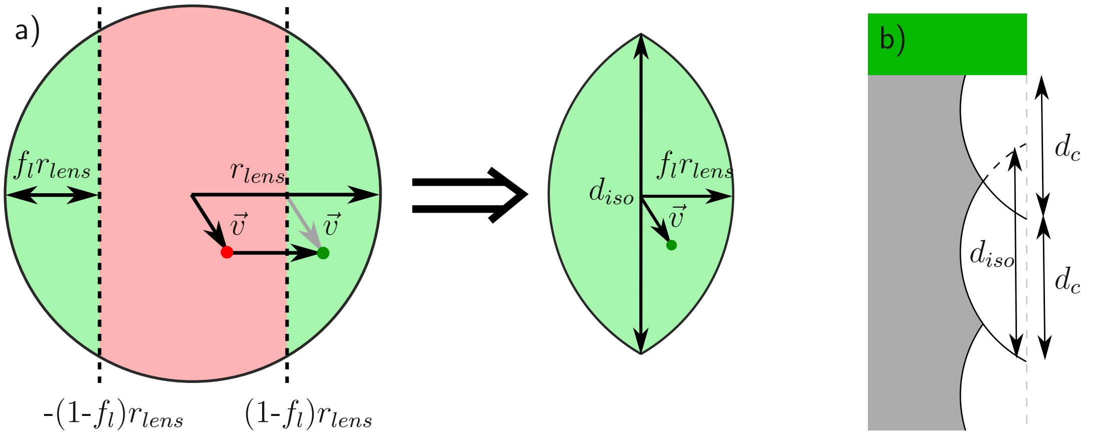


Figure 5.18: (a) Starting from a spherical distribution, a lens distribution can be created by removing the centre part. The vector $\vec{v}$ is translated to the remaining parts. (b) Using this lens distribution, a difference in vertical and lateral etch rate can be modeled for $f_t = 1.5$ and $f_l = 0.5$. CC BY 4.0 [269]

The simplest approximation is to assume that the etchant flux decreases linearly after a certain depth L_t. Each cycle performed below this depth will then be affected by the reduced etch rate. Given the total number of cycles N_c and the depth at which tapering starts, the number of tapered cycles is given by

$$N_t = N_c - \frac{L_t}{d_c} = N_c - N_s \quad , \tag{5.9}$$

where N_s is the number of cycles unaffected by the tapering.

The depth of the rectangular distributions $L(\vec{x})$ used to generate the smooth profile is then given by

$$L(\vec{x}) = \min\left(L_t + \frac{|\vec{x} - \vec{m}|}{w_t}(L_0 - L_t), L_b\right) \quad , \tag{5.10}$$

where w_t is the tapering width, which is the lateral distance between the tapered side wall and a straight side wall at the bottom of the structure. L_0 is the depth at which the etch rate would balance the deposition rate, so it represents the maximum depth achievable with the process, $\vec{m}$ is the laterally nearest point on the mask surface to $\vec{x}$ and L_b is the bottom of the structure after the process given by

$$L_b = L_t + \bar{z}_{N_t}(d_c, D) \quad , \tag{5.11}$$

where D is the depth along which the etch depth per cycle decreases from d_c to zero. This depth is given in Appendix B.1 and is used to find the total depth of all tapered cycles $\bar{z}_{N_t}(d_c, D)$. This depth depends only on the number of tapered cycles and the ratio of the etch depth of the final cycle and the initial etch depth per cycle d_c, denoted as r_e:

$$r_e = \frac{d_{N_t}}{d_1} = 1 - \frac{w_t}{w_{tot}} \quad , \tag{5.12}$$

where w_{tot} is the tapering width at depth L_0, which are both properties of the specific etch process and must therefore be tuned to the precise chemistry used [272]. If several features of different sizes should be modelled simultaneously, w_{tot} and L_t may also have to be adjusted depending on the size of each feature, in order to include effects such as aspect ratio dependent etching (ARDE) [273].

Once the straight profile has been obtained for the DEM sequence, the radius of all scallops must be found. They will be constant until tapering starts at depth L_t. They will then decrease during each cycle, with the n^{th} radius given by

$$d_n = d_{iso} \begin{cases} 1 & \text{if} \quad n \leq N_s \\ 1 - \left(\bar{z}_{N_t}(d_c, D) - L_t\right)/D & \text{if} \quad n > N_s \end{cases} , \tag{5.13}$$

with the lens distributions describing the n_{th} scallop centred at the height

$$z_n = \begin{cases} d_c n & \text{if} \quad n \leq N_s \\ d_c N_s + \bar{z}_n(d_c, D) & \text{if} \quad n > N_s \end{cases} . \tag{5.14}$$

The above model for the DEM sequence was used to emulate a structure from [265] and the resulting geometry was compared to experimental data. As shown in Fig. 5.19, the decreasing scallop size observed in the experiment is matched well by the simple approximation of a linear decrease in etchant flux down the feature. Despite the resolution of the level set (LS) being barely sufficient to properly represent the smallest scallops, the model can produce an accurate description of the final structure without introducing numerical artefacts, as is often the case when using iterative advection schemes [274]. The model parameters used to perform the emulation are listed in Table 5.2 and hence describe the modelled chemistry and can be applied to different geometries, providing reliable results for any geometry or number of cycles.

L_t[µm]	d_c[µm]	f_t	f_l	r_e
-	-0.98	1.20	0.75	0.30

Table 5.2: Fitting parameters for the model of the DEM sequence presented in [265].

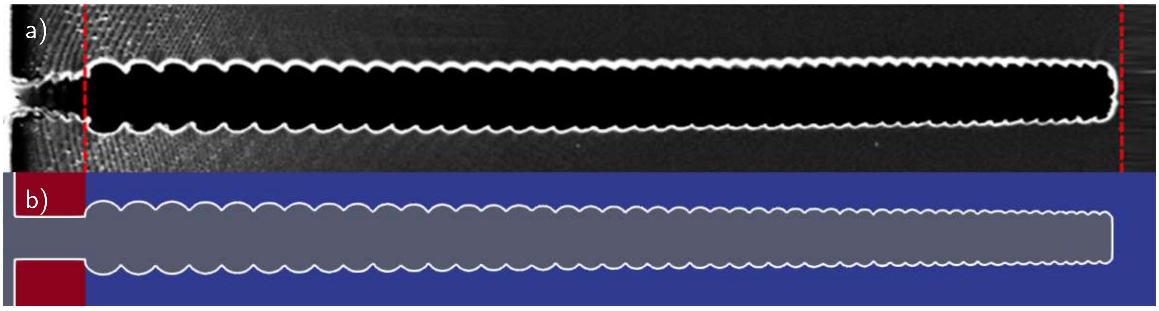


Figure 5.19: (a) DEM process without tapering, but a decrease in scallop sizes down the feature. (b) Emulation of the same process with $N_c = 50$. CC BY 4.0 [269]

DREM Sequence

The tapering of the smooth profile for the DREM sequence can be modelled similarly to the DEM sequence through the use of thin vertical box distributions changing in depth depending on their proximity to the mask. Therefore, the vertical distribution can also be described by Eq. (5.10). However, due to the differing physical mechanisms leading to the tapering, the definitions of L_0 and L_b differ from those of the DEM sequence. In the DREM sequence, the scallops are constant in size down the feature and therefore the bottom of the trench L_b is simply

$$L_b = L_t + d_c N_t = d_c N_c \quad . \tag{5.15}$$

The maximal depth achievable with the modelled process L_0 is then given by

$$L_0 = L_t + \frac{d_c N_t}{1 - r_e} = L_t + \frac{d_c N_t}{w_t / w_{tot}} \quad , \tag{5.16}$$

which means that w_t describes how the closing of the top of the feature affects the removal of the protective layer at the bottom of the trench. If there is no tapering, $w_t = 0$ and there is no effect on the trench. However, if $w_t \geq w_{tot}$, the top of the feature is pinched off entirely and therefore the tapered side walls meet at the bottom of the feature. Therefore, etching stops since no etching species can reach the bottom at this stage. Due to the segregation of the removal of passivation and the etching of the substrate, time ramping can be used to ensure scallops which are constant in size, as described in Section 5.1.6.2. Therefore, no additional considerations are necessary to describe the scallops down the feature as long as the time ramping is tuned correctly. If the time ramping, however, is not conducted adequately, then the DEM sequence model can be used to describe the resulting geometry.

A well tuned DREM process was simulated using the above model to emulate the sausage-chain-like pillars presented in [265]. Since pillars, rather than vias, are etched, particle transport to the bottom of the feature is not of great concern and it can be assumed that the sequence is tuned well. The structure is generated using 8 cycles of a DREM sequence followed by an isotropic etch step creating an under etch resulting in a large scallop at the end. In the original experiment, these 8 cycles plus an isotropic etch are repeated 8 times to generate tall pillars resembling sausage chains in appearance. Using the geometric DREM model, the 3D structure shown in Fig. 5.20 was created using the model parameters listed in Table 5.3. The final structure consists of 72 individual pillars represented using 3,827,322 LS values and shows good agreement with the structure generated in [265] shown on the right in Fig. 5.20.

Due to the efficient algorithm employed for the emulation of the geometry, even large geometries can be created quickly, including process-specific effects which may strongly influence the final structure and lead to significant changes in the electrical characteristics of the structure. Therefore, even large and complex structures can be generated efficiently, allowing for the process-aware generation of entire devices, such as MEMS actuator [275] or photonic crystals [276].

L_t[µm]	d_c[µm]	f_t	f_l	r_e
-	-0.25	1.20	0.50	1.00

Table 5.3: Model parameters for the DREM model used to generate the sausage-chain-like geometry in Fig. 5.20.

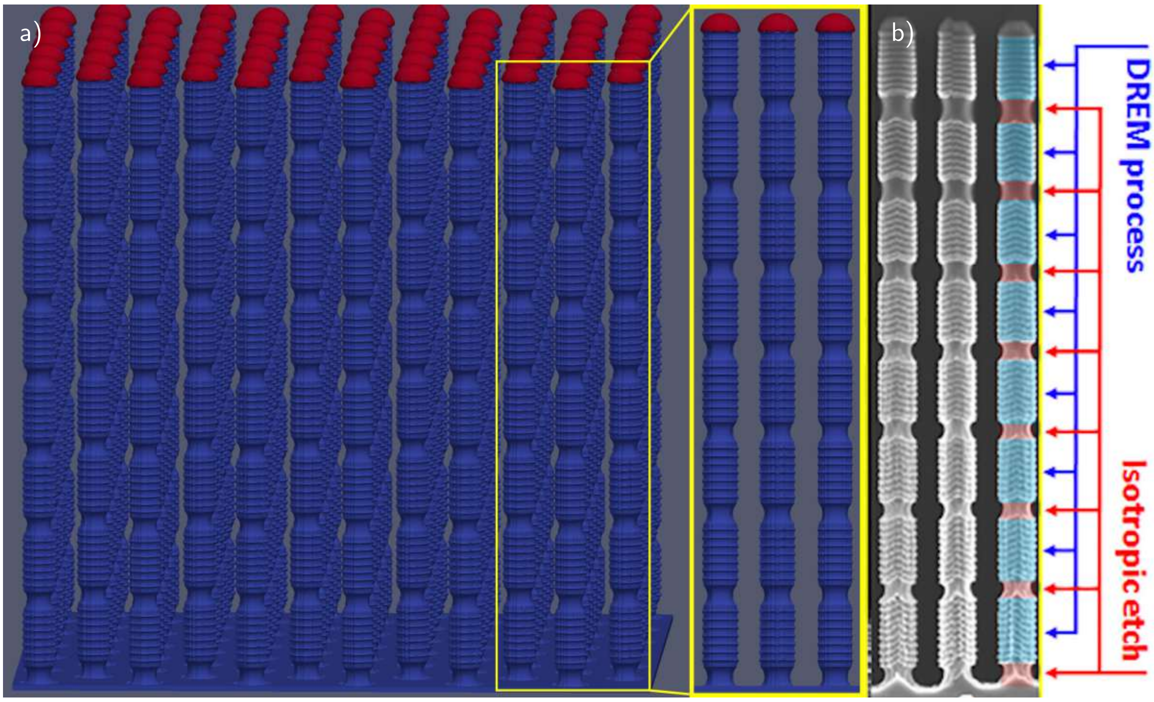


Figure 5.20: (a) Emulation of sausage-chain like structures using several DREM sequences, as well as longer intermittent isotropic etch steps. (b) SEM image of the sausage-chain pillars generated by the DREM process. The emulation was performed using 32 threads and took 14 minutes and 52 seconds to execute. 80 cycles of the DREM model were performed without tapering using the fitting parameters given in Table 5.3. In order to generate the isotropic etch sections, every 10th scallop's isotropic etch depth was increased to $d_{iso} = -1.0$. CC BY 4.0 [269]

DREAM Sequence

The distributions required to describe the DREAM sequence are the same as those used in the DREM sequence. Since the additional ash step prevents the closing of the top of the feature, only this process step influences the tapering width [266]. If it is not effective, the pile up of material on top leads to the time ramping failing due to the smaller top opening. Therefore, scallop sizes cannot be kept constant and RIE lag, similar to that of the DEM sequence, is observed. Therefore, in order to describe the DREAM sequence, the model parameters can be adjusted according to the applied ash time.

In the simplest case, only the ratio between the initial and the final scallop r_e is adjusted according to the ash time t_a. Using a simple model based on Knudsen diffusion down the feature to describe the changing properties of the process with changing ash time (see Appendix B.2), the scallop ratio is given by

$$r_e(t_a) = p_0 - \frac{p_1}{p_2 + t_a} \quad , \tag{5.17}$$

where $p0$, $p1$ and $p2$ are fitting parameters. Using experimental data from [267] and [266], r_e was calibrated to the ash time per cycle assuming constant N_c and L_t, resulting in fitting values of $p_0 = 1.17$, $p_1 = 0.59$, $p_2 = -0.44$ (see Appendix B.2 for details). Since $0 \leq r_e \leq 1$, ashing will only influence the result if it lasts longer than the minimal ash time $t_0 = p_1/p_0 - p_2 = 0.94\,\mathrm{s}$ and no additional effect is observed for ash times longer than $t_m = p_1/(p_0 - 1) - p_2 = 3.91\,\mathrm{s}$.

Therefore, if the ash time approaches zero, the DEM model is recovered, while for the maximum ash time, the result is equivalent to that of a perfectly tuned DREM model. The model described above was used to emulate a series of high aspect-ratio vias using the model parameters shown in Table 5.4. As can be seen in Fig. 5.21, the results match the experimental data quite well over the entire range of ashing times.

L_t[µm]	d_c[µm]	f_t	f_l	r_e
-24.96	-0.37	1.20	0.50	$r_e(t_a)$ (see Eq. (5.17))

Table 5.4: Model parameters for the DREAM sequence used for a series of via etch emulations with changing ashing time.

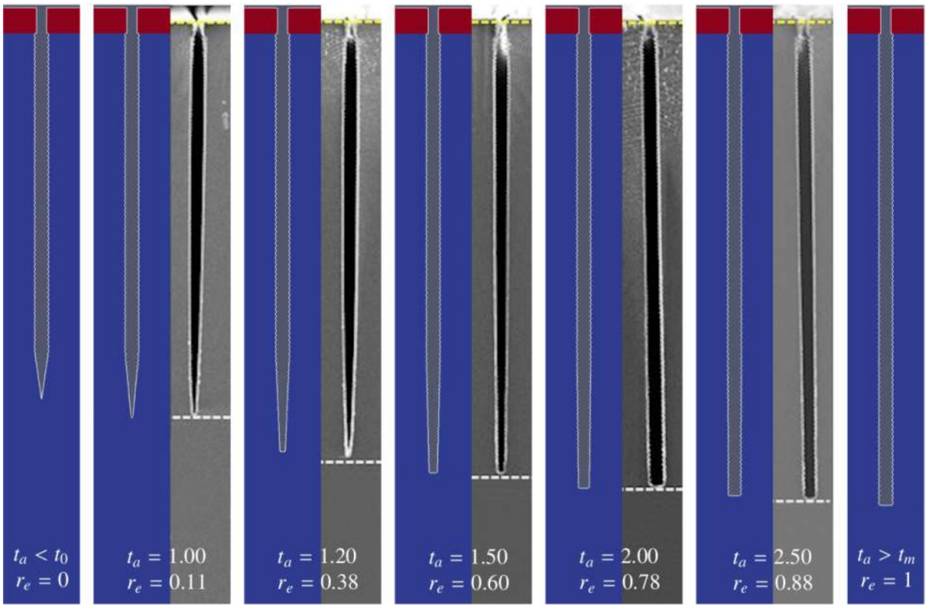


Figure 5.21: DREAM Sequence simulations for $N_c = 100$ with different ash step times compared to experimental data. Only the ash time was varied as input for the DREAM model, while all other parameters were fixed with values provided in Table 5.4. CC BY 4.0 [269]

5.1.6.5 Physical Model

In order to predict transport phenomena or to evaluate new chemistries whose effect on the structure is not known, a physical model is required. Emulation cannot provide any features due to wrong process parameters, such as too high etchant fluxes during a process. However, these properties are especially important when considering many cycles of a Bosch Process, where fluxes may change drastically down a trench geometry. In order to display these effects, a physical model for the clear-oxidise-remove-etch (CORE) sequence presented in [267] was implemented.

First, the surface is cleared from any contamination such as left over polymer in the Clear step. This is conducted similarly to the ash step of the DREAM sequence, in a dry O_2 chemistry without any plasma bias. In this physical model, this process step is simulated using MC ray tracing of a single particle species which isotropically and selectively removes materials other than silicon. Due to its high selectivity, it can be applied for longer times without damaging the underlying substrate.

Secondly, the surface is oxidised for a few seconds at higher O_2 pressures. This step was modelled similarly to the Clear step, but due to the higher chamber pressure, more oxygen radicals actually hit the surface, which is why this process step was simulated with a much higher source flux. When oxygen radicals contact the surface, they form thin SiO_2 layers, protecting the substrate in the subsequent etch step. Since these layers are very thin and only small amounts of the substrate are lost to the oxidation, this step was modelled as isotropic deposition with input parameters as shown in Table 5.5. This is more efficient than physically modelling oxidation which requires both the substrate and the oxide surface to be advanced depending on their width.

Step	Model	Ion Flux	Ion Energy	Etchant Flux	Polymer Flux
Clear	O_2	0	-	0	$1 \cdot 10^{14}$
Oxidise	O_2	0	-	0	$1 \cdot 10^{16}$
Remove	SF_6	$1 \cdot 10^{16}$	100 eV	$2 \cdot 10^{16}$	0
Etch	SF_6	0	-	$7 \cdot 10^{16}$	0

Table 5.5: Input parameters to physical models used for each step of the CORE sequence with fluxes given in units of $\mathrm{cm}^{-2}\,\mathrm{s}^{-1}$.

Next, the remove step is carried out in a SF_6 plasma at plasma bias and low pressure to ensure that ions gain enough energy without colliding with other species in the reactor scale. This step is modelled using the SF_6 plasma model described in Section 5.1.4.3. Since there is no polymer flux, no sidewall deposition is observed and the protective SiO_2 layers are removed highly directionally. This step is modelled using different ion and neutral etching species, leading to ion-enhanced etching of the polymer and the substrate once it is exposed.

In the final etch step, the SF_6 flow rate is increased and there is no plasma bias anymore, which means that etching proceeds isotropically. The etch time must be set carefully because it sets the scallop size and may lead to defects in the final structure if it is too long. Such damage can be seen in Fig. 5.22, where the passivation was not thick enough to sustain prolonged etching times. This results in additional under etches which create unwanted holes in the side walls. Furthermore, the time ramping discussed in Section 5.1.6.2 must be applied in order to ensure uniform scallop sizes for later etch steps.

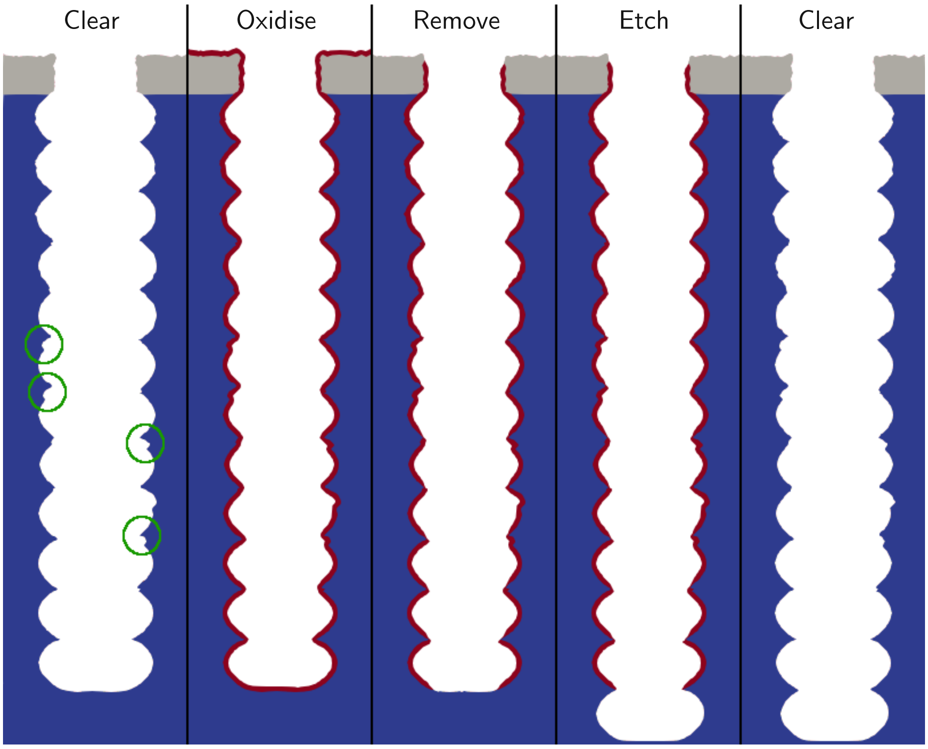


Figure 5.22: Cycle 13 of a CORE sequence, highlighting etching damage (green circles) of previous cycles due to a short oxidise step, which results in too thin passivation layers. These are then etched and additional features are etched into the scallops during subsequent etch steps.

Much like the real process, physical modelling is driven by underlying descriptions of the interacting molecules and the surface. Therefore, careful tuning and fitting of process parameters is also necessary in every physical model. Due to the modelling of the underlying physics of the process steps, the behaviour of etch chemistries or more complex etch cycles, such as the CORE sequence, can be evaluated. Hence, physical models can be used to test the applicability of new chemistries or processing techniques.

However, due to the more intense fitting effort compared to empirical emulation models, physical models are not well suited for quick and efficient structure generation. In this case, it is more appropriate to include required properties in geometric models and apply them to structures of interest.

5.2 Device Process Flows

Using the models developed during the course of this work, as presented in the previous section, a full process flow for the manufacture of a semiconductor device can be simulated or emulated. When the models are applied correctly, specific process-induced effects on the final device can be observed and can aid in the understanding of the effect of processes on specific properties of a device. Especially effects which stem predominantly from the combination of fabrication steps can be modelled this way, giving insights into the underlying physical effects which might lead to unexpected features of the final device. For example, a cleaning step may be necessary between two sequential fabrication steps due to the first process contaminating the substrate, which negatively impacts the successful execution of subsequent processing steps.

In this section, several full process flows used to generate entire devices will be presented. The exact combination of processing steps used to create the devices are taken from literature, using commonly observed values for unpublished process characteristics. Therefore, the models presented in this section should not be seen as exact replications of certain device processing flows. They rather present proofs of concept for the simulation of large structures using a combination of interdependent physical and empirical process models for the process-aware generation of full device geometries. In this way, dominant physical mechanisms can be identified and investigated in detail to find optimal processing conditions for the generation of complex structures.

5.2.1 22 nm FinFET

The process flow for the production of the 22 nm transistor employs self-aligned double patterning (SADP) for the definition of the fins on a bulk silicon substrate [277]. The high-k metal gate (HKMG) [278] is manufactured using the replacement metal gate (RMG) process [279], where the high-k material is deposited before the poly-Si dummy gate. During dummy gate removal through selective etching, this thin dielectric layer is left intact and the S/D regions are protected by the interlayer dielectric (ILD). In the final front-end of line (FEOL) processing step, the metal and gate contact materials are then deposited on top of the existing layer.

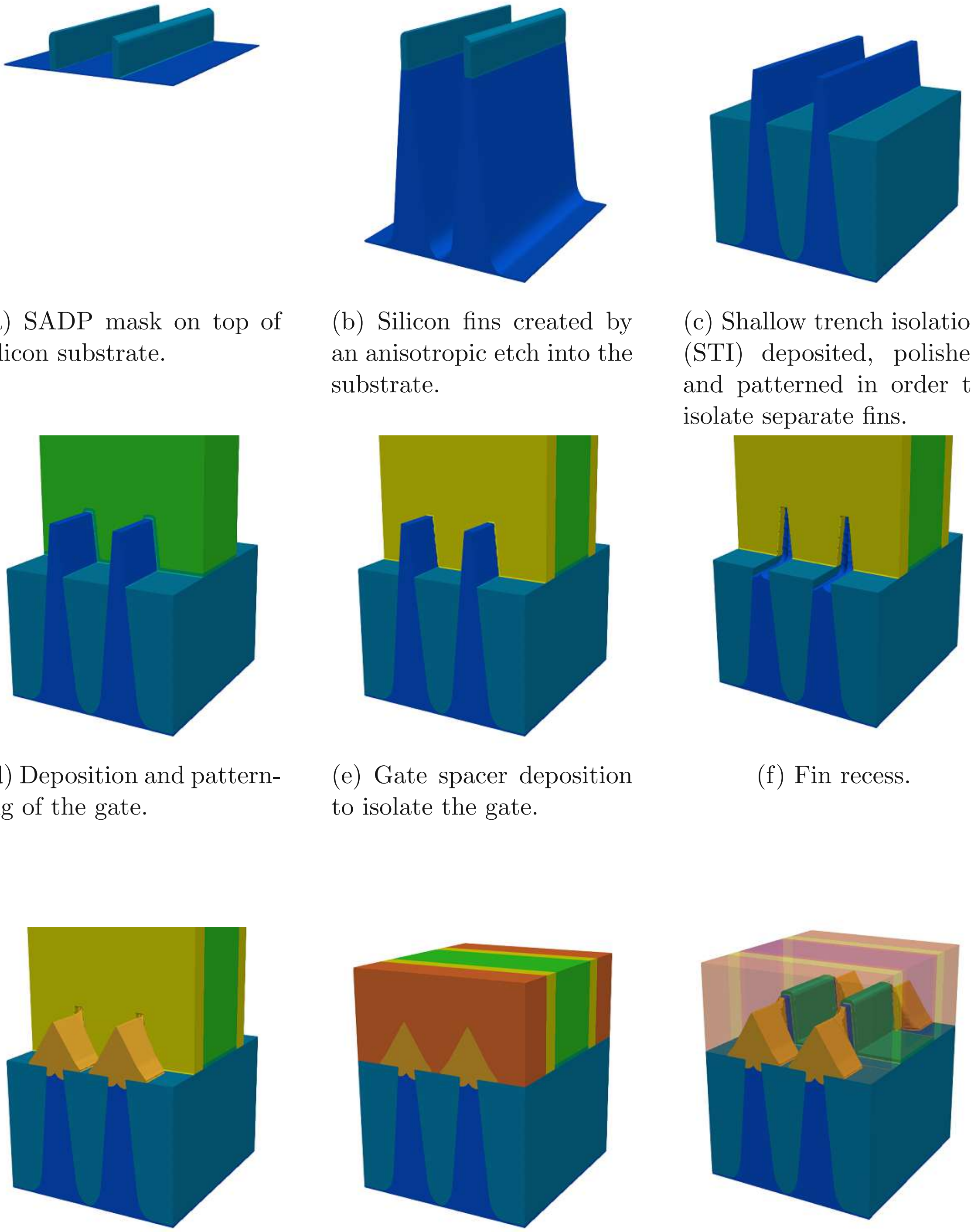
Emulation models were applied consecutively to generate a 22 nm FinFET consisting of two fins rather than a single one, which leads to better electrical characteristics of the device [280]. The exact models carried out are listed in Table 5.6, referring to the structures shown in Fig. 5.23, which resulted after critical process steps were conducted. Models written in bold text are executed using iterative advection rather than geometric advection.

The final structure clearly shows all the relevant features, including the crystal facet dependent shape of the epitaxially grown S/D contacts. The epitaxial growth is the only process step emulated using iterative advection, as the temporal evolution of the crystal facets dominates the final geometry, which cannot be represented appropriately using geometric advection. However, since no expensive MC ray tracing simulations are required, only the computational cost of the SLLF advection scheme

Fabrication step	Applied model
Fin Mask	Mask
SADP	Geometric Deposition, Geometric Etch
Mask Removal	Delete Material
Fin Patterning	Geometric Etch
SADP Mask Removal	Delete Material
STI Deposition, CMP	Geometric Deposition, CMP
STI Etching	Geometric Etch
High-k Deposition	Geometric
Dummy Gate Deposition, CMP	Geometric, CMP
Dummy Gate Mask	Mask
Gate Patterning	Geometric Etch
Gate Mask Removal	Delete Material
Spacer Deposition	Geometric Deposition
Spacer Patterning	Geometric Etch
Fin Recess	Geometric Etch
S/D Epitaxy	**SLLF Epitaxial Growth**
ILD Deposition, CMP	Geometric Deposition, CMP
Dummy Gate Removal	Delete Material
Gate Metal Deposition	Geometric Deposition
Gate Contact Deposition	Geometric Deposition
Chemical mechanical planarisation (CMP)	CMP

Table 5.6: Sequence of process steps used to generate the 22 nm FinFET structure and the models used. Geometric deposition and etching refers to a model executed by the algorithm presented in Section 2.4.3, while SLLF Epitaxial Growth refers to the physical epitaxial model presented in Section 5.1.2.1. CMP is modelled by simply clipping all level sets by a plane using Boolean operations.

is incurred. Therefore, the entire structure could be generated in a matter of a few minutes, which allows for changes in the process flow to be made quickly and effectively, resulting in a structure that could be used for subsequent device simulation within a design technology co-optimisation (DTCO) flow straight-forwardly.



(a) SADP mask on top of silicon substrate.

(b) Silicon fins created by an anisotropic etch into the substrate.

(c) Shallow trench isolation (STI) deposited, polished and patterned in order to isolate separate fins.

(d) Deposition and patterning of the gate.

(e) Gate spacer deposition to isolate the gate.

(f) Fin recess.

(g) S/D epitaxy.

(h) ILD deposition and dummy gate removal.

(i) Final structure after gate metal and contact deposition.

Figure 5.23: FEOL processing steps for the fabrication of the 22 nm FinFET employing the RMG process flow.

5.2.2 5 nm SRAM Cell

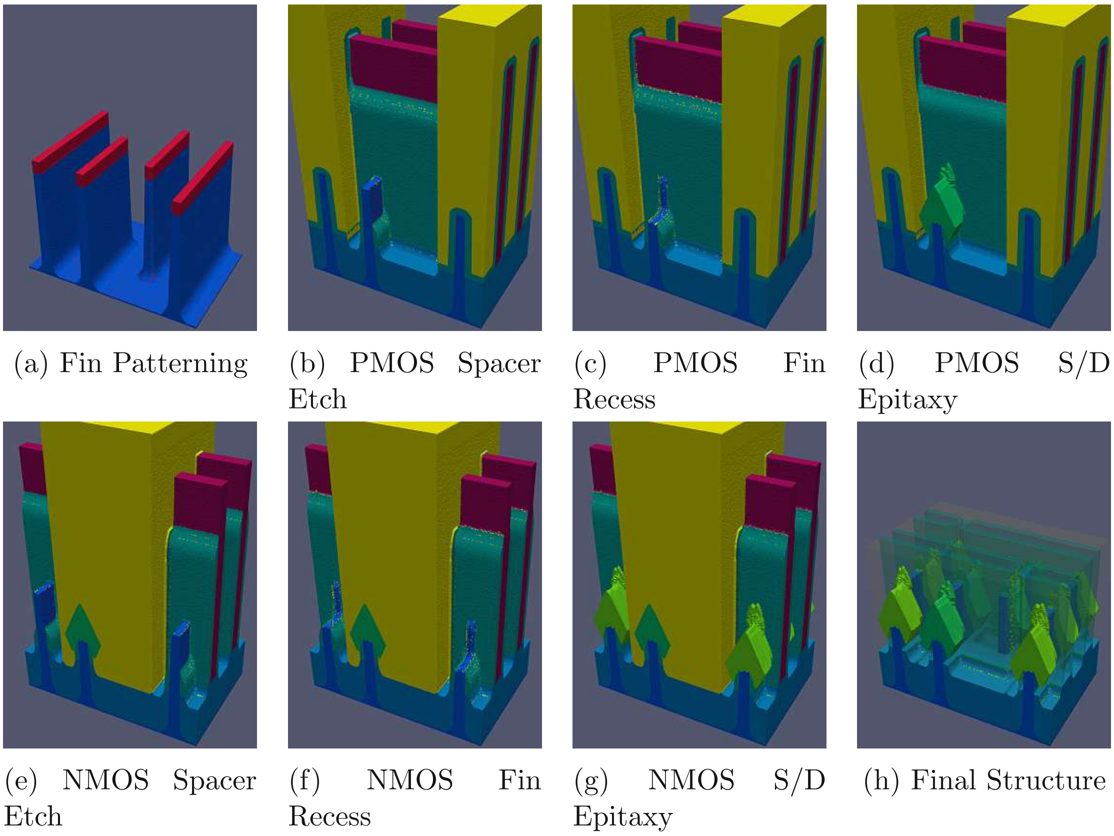
FinFETs have been the standard design for modern 3D transistors up to the most recent technology nodes [281]. Due to the complex 3D arrangement, parasitic resistances and capacitances can become a problem for stable device operation [282]. As these properties are strongly dependent on the exact geometry of the transistor, the geometries resulting from a simulation should be as accurate as possible. For a full description of the electric characteristics of devices, the entire circuit must be considered, so a full static random access memory (SRAM) cell of the 5 nm technology node was simulated.

Since parasitic resistances and capacitances mostly originate from the S/D regions, simulation time and effort can be saved by only physically simulating these sections of the device, while emulating all other fabrication steps [283]. The process flow for the SRAM cell is listed in Table 5.7, with critical process steps for the formation of the S/D regions shown in bold text.

Fabrication step	Applied model
Fin Mask	Mask
Fin Patterning	**SF_6/CH_2F_2 Plasma Etching**
STI Deposition, CMP	Geometric Deposition, CMP
STI Etching	Geometric Etch
Dummy Gate Depo, CMP	Geometric Deposition, CMP
Dummy Gate Mask	Mask
Gate Patterning	Geometric Etch
Spacer Deposition	Geometric Deposition
ILD Deposition, CMP	Geometric Deposition, CMP
Once for NMOS and PMOS each	
Mask NMOS/PMOS	Mask
PMOS/NMOS ILD Etch	Geometric Etch
Spacer Etch	**CH_3F Plasma Etching**
Fin Recess	**Selective Dry Etch**
S/D Epitaxy	**SLLF Epitaxial Growth**
ILD Deposition, CMP	Geometric Deposition, CMP
Dummy Gate Removal	Geometric Etch
HKMG Deposition, CMP	Geometric Deposition, CMP

Table 5.7: Process steps used to generate the final SRAM structure and the corresponding modelling approaches. The bold text shows which steps were applied using physical models.

The first process step of interest is the fin patterning, which is simulated using MC ray tracing with the chemical model for CH_3F presented in Section 5.1.4.1. All other critical process steps are applied during S/D formation, which means that they must be repeated once for the NMOS and once for the PMOS regions of the cell. The second critical step is CH_3F plasma etching of the gate spacer whereby the top of the fin is exposed. Next, the fin is etched in a SF_6/CH_2F_2 plasma to create a clean surface for the subsequent epitaxial growth of the S/D contacts. The SEG of silicon on top of the exposed fin is simulated using the model presented in Section 5.1.2 using the empirical rates presented in [100]. The geometries resulting from these critical process steps are shown in Fig. 5.24 and clearly show the physical behaviour of the specific process steps, which can only be appropriately described using these sophisticated models.


(a) Fin Patterning (b) PMOS Spacer Etch (c) PMOS Fin Recess (d) PMOS S/D Epitaxy

(e) NMOS Spacer Etch (f) NMOS Fin Recess (g) NMOS S/D Epitaxy (h) Final Structure

Figure 5.24: (a)-(g) SRAM structure after the fabrication steps with physical models, highlighted in bold in Table 5.7. (h) Final SRAM structure, with high-k dielectric and metal gate (HKMG), spacer and ILD transparent to show the structure of the fins and S/D regions. ©2021 IEEE [283]

Since only a small number of processes actually affect the exact geometry of the S/D regions, only these processes need to be simulated using computationally expensive physical models. All other process steps can be emulated using the highly efficient geometric advection algorithm presented in Section 2.4.3. Therefore, the entire SRAM structure can be generated in less than 16 minutes, where more than 85% of the total simulation time is consumed by the physical models, as shown in Table 5.8. Again, models in bold text indicate physical models, which clearly require the vast majority of computational effort. Therefore, intricate process specific properties influencing the electric properties of the devices in a circuit can be simulated for the entire cell in a reasonable time frame on a consumer desktop computer.

Fabrication step	Simulation Runtime
Fin Mask	0.1 s
Fin Patterning	**186.8 s**
STI Deposition, CMP	0.3 s
STI Etching	13.5 s
Dummy Gate Deposition, CMP	0.3 s
Dummy Gate Mask	0.1 s
Gate Patterning	24.4 s
Spacer Deposition	2.8 s
ILD Deposition, CMP	0.3 s
Once for NMOS and PMOS each	
Mask NMOS/PMOS regions	0.2 s / 0.1 s
PMOS/NMOS ILD Etching	60.7 s / 76.6 s
Spacer Etching	**204.7 s / 225.9 s**
Fin Recess	**60.6 s / 63.6**
S/D Epitaxy	**28.8 s / 28.2 s**
ILD Deposition, CMP	0.3 s / 0.4 s
Dummy Gate Removal	0.1 s
HKMG Deposition, CMP	6.7 s
Emulation Models	2 min 22.8 s
Physical Models	**13 min 18.6 s**
Total Runtime	15 min 41.4 s

Table 5.8: Runtime for the simulation of each modelled process step. Physical models are shown in bold text. The simulation was carried out on an AMD Ryzen3950X processor and took less than 16 minutes to complete, where more than 85% of the simulation time was consumed for the evaluation of the physical models.

5.2.3 Beyond 5 nm Stacked Nanosheet FET

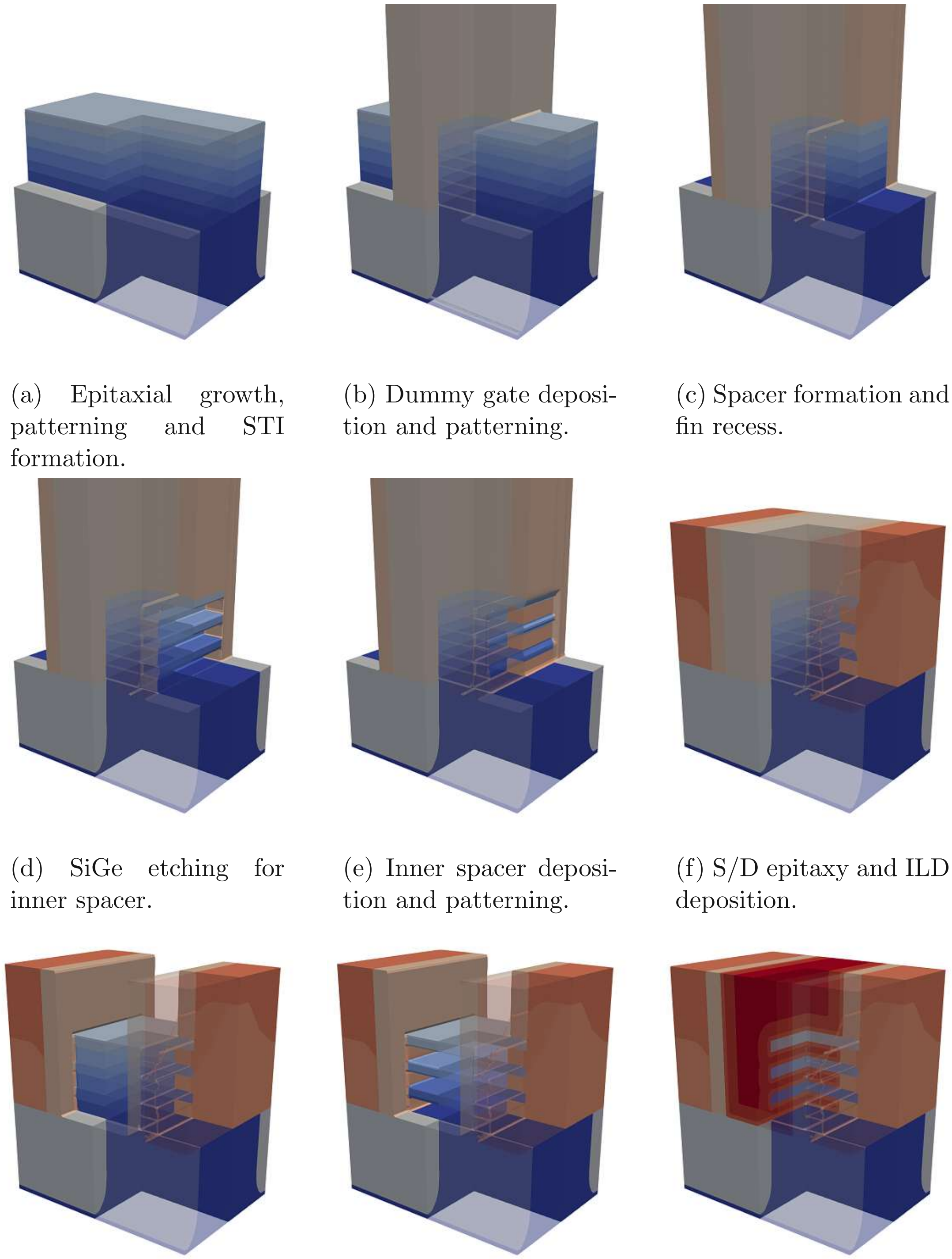
Since its introduction in the 22 nm technology node, the FinFET has been steadily scaled down to reach ever smaller dimensions. However, electrostatic control over the channel of FinFETs is not ideal and the scaling of FinFETs is reaching its limit. Hence, for technology nodes beyond the 5 nm node, the use of gate all-around (GAA) transistors has been proposed [284, 285].

A GAA transistor was emulated using the process flow described in [285]. The models used to generate the structure are listed in Table 5.9, with key processes shown in Fig. 5.25. Due to the wide nano sheets, existing masking technology can be used without reaching critical process limits [286]. Additionally, the stacking of several sheets, which increases power, does not affect the footprint of the structure. Therefore, the electrical characteristics can be improved without requiring larger transistors, as would be the case when additional fins are required in a FinFET.

The individual process steps required to build this type of transistor are very similar to the FinFET process flow, with a few additional steps, as listed in Table 5.9. First, a multilayered stack of SiGe/Si/SiGe/Si is grown epitaxially, as shown in Fig. 5.25a. The silicon layers will form the nanosheet channels of the final device, so the layer width is equivalent to the height of the nanosheets [287]. Subsequently, the same process steps as in FinFET production are used for fin generation: STI and dummy gate formation. However, after the gate spacer is formed, a selective etch process is used to create an under etch in the SiGe/Si stack, as shown in Fig. 5.25d. This under etch will be used to deposit the inner spacer shown in Fig. 5.25e, which isolates the S/D region from the gate and therefore should be as wide as the gate spacer [288]. During the RMG process, the SiGe layers in the channel region are selectively removed to expose the stacked silicon nanosheets, as shown in Fig. 5.25h. Finally, the conventional HKMG is deposited on the suspended nanosheets to form gate contacts all around the channels, leading to optimal electrostatic control [202]. The final structure is shown in Fig. 5.25i, clearly showing the HKMG wrapped around the nanosheet channels, as well as the characteristic crystal facets in the epitaxially grown S/D regions.

Fabrication step	Applied model
SiGe/Si/SiGe/Si/SiGe/Si Epitaxy	Geometric Deposition
Fin Mask	Mask
SADP	Geometric Deposition, Geometric Etch
Mask Removal	Delete Material
Fin Patterning	Geometric Etch
SADP Mask Removal	Delete Material
STI Deposition, CMP	Geometric Deposition, CMP
STI Etching	Geometric Etch
Dummy Gate Deposition, CMP	Geometric, CMP
Dummy Gate Mask	Mask
Gate Patterning	Geometric Etch
Gate Mask Removal	Delete Material
Spacer Deposition	Geometric Deposition
Spacer Patterning	Geometric Etch
Fin Recess	Geometric Etch
SiGe Etch for Inner Spacer	Geometric Etch
Inner Spacer Formation	Geometric Deposition, Geometric Etch
S/D Epitaxy	SLLF Epitaxial Growth
ILD Deposition, CMP	Geometric Deposition, CMP
Dummy Gate Removal	Delete Material
High-k Gate Dielectric Deposition	Geometric Deposition
Gate Metal Deposition	Geometric Deposition
Gate Contact Deposition	Geometric Deposition
CMP	CMP

Table 5.9: Emulation models used to simulate the fabrication of a GAA FET anticipated for applications beyond the 5 nm technology node.



(a) Epitaxial growth, patterning and STI formation.

(b) Dummy gate deposition and patterning.

(c) Spacer formation and fin recess.

(d) SiGe etching for inner spacer.

(e) Inner spacer deposition and patterning.

(f) S/D epitaxy and ILD deposition.

(g) Dummy gate removal.

(h) Nanosheet channel release.

(i) Final geometry after HKMG deposition.

Figure 5.25: Critical process steps in the manufacture of the GAA transistor described in [285].

5.3 Summary

In this chapter, several microelectronic fabrication steps were introduced and their modelling approaches discussed. Long-standing and well understood processes, such as CVD were emulated using empirical or analytical methods and simulated using physical modelling for particle transport and surface chemistry.

Epitaxial growth is highly important for semiconductor fabrication and has been emulated using geometric models for the S/D regions of a modern FinFET structure, as well as simulated with physical models for a sensitive $Si_2H_2Cl_2$ chemistry. Crystal direction dependent wet etching was shown for a MEMS structure using the SLLF numerical advection scheme developed during the course of this work.

Subsequently, the main physical mechanisms leading to the observed etch behaviour were discussed for commonly used plasma etch processes. Physical models for these chemistries are described and applied in the physical simulation of the etching of a gate stack of the 14 nm technology node.

Furthermore, intricate emulation models for the efficient modelling of different types of modern Bosch Processes were presented. These models have all been developed during the course of this work and are discussed in detail, including comparisons to experimental results.

Process flows for the fabrication of entire devices are presented and the previously introduced models are used to generate the final structures. These models include a combination of geometric and physical models, leading to the efficient generation of process-aware structures which include physical effects specific to the applied fabrication processes.

Chapter 6

Summary and Outlook

State of the art simulation and emulation techniques for process technology computer aided design (TCAD) were presented and their implications on the modelled processes were discussed. In particular, modelling strategies for materials and material interfaces were presented and described, focussing on the underlying numerical methods and their relative advantages. A modelling framework for the highly physical simulation and for the efficient empirical emulation of semiconductor fabrication processes was designed and implemented in a C++ high performance toolchain. This framework allows for high flexibility in the modelling capabilities due to a careful consideration of the numerical methods employed. Hence, a number of existing modelling techniques were reviewed, their relative advantages discussed, and several additional algorithms developed to create a powerful and computationally efficient modelling framework for process TCAD.

Continuum modelling was identified as the most appropriate technique for the description of materials involved in semiconductor fabrication. Specifically the sparse-field level set method was shown to provide all the necessary functionality to describe, analyse and manipulate complex material interfaces with minimal computational requirements. Furthermore, the volume representation of materials using a cell-based approach, developed during the course of this work in the C++ ViennaCS library, was shown as a viable data structure complementing the topographic level set description. Using a sparse data approach on the same grid as the level set, allows for efficient access of volume and topography data with minimal memory requirements.

The robust description of material evolution through surface advection is key for the modelling of fabrication processes. Several numerical techniques for the manipulation of materials were reviewed. Due to the robust handling of complex topographic changes, the level set provides better computational efficiency compared to explicit methods.

The iterative advection of level sets and numerical schemes for the solution of the level set equation were presented and their limitations for process simulation were discussed. For highly anisotropic velocity fields, the Stencil Local Lax-Friedrichs integration scheme was developed which reduces numerical errors by setting appropriate dissipation factors and limiting the integration time to robust lengths. The evolution of materials through volume properties can best be achieved using the proposed volume data structure, extracting a velocity field from this structure, and modelling the topographical change using the level set engine.

Geometric advection provides an entirely different way to evolve interfaces in a highly efficient manner. The material is not moved in time, but rather geometric considerations about the evolved surface at the end of a fabrication step are described using geometric distributions which are applied to the initial surface in order to generate the final structure directly. Therefore, material interfaces can be advanced in a single step without any limitations on advection distance or resolution, as is the case when using iterative advection.

The exact properties of material evolution are a complex result of the interaction of atoms and molecules with material interfaces. Therefore, physical models rely heavily on transport mechanisms and surface reactions to be modelled appropriately. Top down Monte Carlo ray tracing was shown to be the most appropriate for this application, as it provides great flexibility in the physical description of manufacturing processes. Using this computational technique, a general modelling approach for ion-enhanced plasma etching was presented.

For the implementation of the sparse-field level set, hierarchical run-length encoding was implemented in a specialised data structure ViennaHRLE which provides fast sequential data access. All presented algorithms operating on the level set were implemented in the ViennaLS library, which provides a comprehensive tool set for the creation, manipulation and analysis of level set surfaces in multiple dimensions. The ViennaRay library provides the Monte Carlo ray tracing functionality using the open-source library embree which is used to perform efficient surface intersection tests. Finally, all computational capabilities are combined in the ViennaPS library, which provides a highly flexible modelling framework encompassing numerous types of process models. Due to the combination of these different computational techniques in one single process simulation library, highly efficient emulation models can be carried out on the same data structure as sophisticated physical simulations, which allows for maximal flexibility when modelling process flows for semiconductor fabrication.

Finally, numerous fabrication steps and models developed using the implemented process TCAD framework are presented. Empirical and physical process models are provided and their relative accuracy and run times are compared. These process models were then applied in full device process flows to generate process-aware structures highly efficiently. A full SRAM circuit at the 5 nm technology node was created using a combination of emulation and simulation techniques to provide a realistic description of crucial sections of the circuit while drastically reducing simulation time by emulating all other sections.

Due to the high flexibility of the implemented simulation framework, numerous additional process models can be developed and implemented straight-forwardly. Especially fabrication processes which strongly depend on volumetric information, such as ion implantation, diffusion, or oxidation could provide great insights into physical mechanisms and allow for an in-depth evaluation of modern process steps and how they influence each other. This simulation of such volumetric properties with sophisticated physical models is crucial in achieving a fully encompassing description of the fabricated structures and has been made possible by the flexibility of the process TCAD toolchain developed during the course of this work.

In order to make the simulator more comparable to experiments, reactor simulations can be used to extract the input parameters to the physical models and thus allow for better integration with processing equipment. However, coupling the simulation framework with fabrication equipment requires deep knowledge of the used reactors, their construction and their operation. Nonetheless, the flexible software design of the implemented framework allows for the integration with equipment-specific software straight-forwardly, making the full simulation of reactors and their effect on the fabricated structures possible.

The structures generated by the process simulation framework ViennaPS can already provide insights into the effects of certain processing techniques. However, the extraction of electrical properties using device simulations still relies heavily on manual conversions to data formats appropriate for device simulators. In order to make full use of the presented process modelling capabilities, a direct integration with device and circuit simulations is indispensable. Therefore, algorithms which allow for the generation of meshes compatible with commonly used device simulators, such as those developed at the Institute for Microelectronics, are crucial and present the logical next step in the development of a full TCAD toolchain.

Appendix A

Calculating Filling Fractions from Level Set Values

Consider a single cell in a grid. In its centre, there is a point of a level set grid, $\vec{O} = (0.5, 0.5, 0.5)$. In order to approximate a surface corresponding to this level set point, a plane is constructed using $\hat{n}$, the normalised normal vector of the level set at $\vec{O}$. A point on the plane, $\vec{P}$ is then found by shifting $\vec{O}$ in the direction of $\hat{n}$ by the level set value at $\Phi(\vec{O})$:

$$\vec{P} = \vec{O} - \hat{n}\frac{\Phi(\vec{O})}{\mid \nabla\Phi(\vec{O}) \mid} \tag{A.1}$$

The plane approximating the surface inside the cell, $p(\vec{x})$ is therefore defined as

$$\hat{n} \cdot (\vec{x} - \vec{P}) = 0 \tag{A.2}$$

Due to the symmetry of the cubic cell, the normal vector can always be mapped into one half/quarter of the first quadrant/octant, bounding the polar angles by $0 < \theta \leq \frac{\pi}{4}$ and $0 < \phi \leq \frac{\pi}{4}$.

A.1 2D Problem

In two dimensions the filling fraction is the area below the line describing the surface within the cell shown in Fig. A.1. This line is given by

$$p = \frac{\hat{n} \cdot \vec{P}}{n_y} - \frac{n_x}{n_y}x = q - \frac{n_x}{n_y}x \tag{A.3}$$

Therefore, the integral is bound to $(0,0) \leq \vec{x} \leq (1,1)$. However, the line might intersect the x-aligned edge of the cell at other points, for $y = 0$ and $y = 1$. These intersections, a and b, are defined by:

$$a = p(y = 0) = \frac{n_y q}{n_x} = \frac{\hat{n} \cdot \vec{P}}{n_x} \quad , \tag{A.4}$$

$$b = p(y=1) = \frac{n_y}{n_x}q - \frac{n_y}{n_x} = \frac{\hat{n}\cdot\vec{P} - n_y}{n_x} \quad , \tag{A.5}$$

where both a and b are bound to the interval [0, 1]. The area A is then given by

$$A = \int_0^b dx + \int_b^a q - \frac{n_x}{n_y}x dx \tag{A.6}$$

Solving this integral gives an expression for the area under curve within the cell, i.e. the filling fraction:

$$A = b + \frac{\hat{n}\cdot\vec{P}}{n_y}(a-b) - \frac{n_x}{2n_y}(a^2 - b^2) \tag{A.7}$$

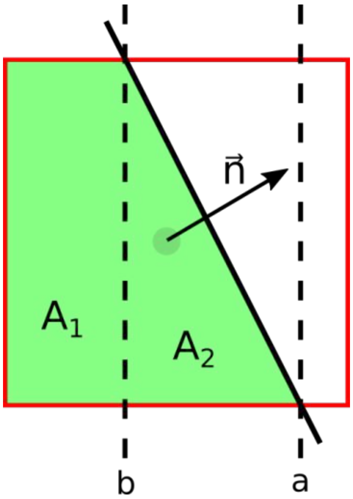


Figure A.1: 2D area calculation for a square cut by a straight line.

A.2 3D Problem

The three dimensional problem can be approached in exactly the same way. The plane which intersects a cube is visualised in Fig. A.2 and is given by:

$$p = \frac{\hat{n}\cdot\vec{P}}{n_z} - \frac{n_x x + n_y y}{n_z} = q - \frac{n_x x + n_y y}{n_z} \quad . \tag{A.8}$$

In the x-direction, the integral is bound by a and b, which are defined as

$$a = \frac{n_z}{n_x}q = \frac{\hat{n}\cdot\vec{P}}{n_x} \quad , \tag{A.9}$$

$$b = \frac{n_z}{n_x}q - \frac{n_y}{n_x} = \frac{\hat{n}\cdot\vec{P} - n_y}{n_x} \tag{A.10}$$

On the top plane of the cube, the integral is bound by c and d:

$$c = \frac{\hat{n}\cdot\vec{P}}{n_x} - \frac{n_z}{n_x} \tag{A.11}$$

$$d = \frac{\hat{n}\cdot\vec{P}}{n_x} - \frac{n_z}{n_x} - \frac{n_y}{n_x} \tag{A.12}$$

In addition to the bounds on the x-axis, the bounds for the y-axis also need to be defined, so p does not contribute to the integral if it is outside of the cube. These are the lines describing the intersection of p with the sides of the cube parallel to the x-y plane. $f(x)$ is the intersect with $p = 0$, given by

$$f(x) = \frac{\hat{n}\cdot\vec{P}}{n_y} - \frac{n_x}{n_y}x \quad . \tag{A.13}$$

The intersect with the side of the cube at $p = 1$ is given by

$$g(x) = \frac{\hat{n}\cdot\vec{P} - n_z}{n_y} - \frac{n_x}{n_y}x \quad . \tag{A.14}$$

The volume V under the plane is then given by the sum of volumes:

$$V = V_1(b) + V_2(a,b) - V_3(d) - V_4(c,d) \tag{A.15}$$

Explicitly, these volumes are:

$$\begin{aligned} V = &\int_{x=0}^{b}\int_{y=0}^{1} p\ dydx + \int_{x=b}^{a}\int_{y=0}^{f(x)} p\ dydx \\ -&\int_{x=0}^{d}\int_{y=0}^{1} p\ dydx - \int_{x=d}^{c}\int_{y=0}^{g(x)} p\ dydx \end{aligned} \tag{A.16}$$

The solutions to the integrals are:

$$V_1(b) = -\frac{n_x}{2n_z}b^2 + \left(\frac{\hat{n}\cdot\vec{P}}{n_z} - \frac{n_y}{2n_z}\right)b \tag{A.17}$$

$$V_2(a,b) = \frac{1}{2n_zn_y}\left[\frac{n_x^2}{3}\left(a^3 - b^3\right) - n_x\left(\hat{n}\cdot\vec{P}\right)\left(a^2 - b^2\right) + \left(\hat{n}\cdot\vec{P}\right)^2(a-b)\right] \tag{A.18}$$

$$V_3(d) = V_1(d) \tag{A.19}$$

$$V_4(c,d) = V_2(c,d) - \frac{n_z}{2n_y}(c-d) \tag{A.20}$$

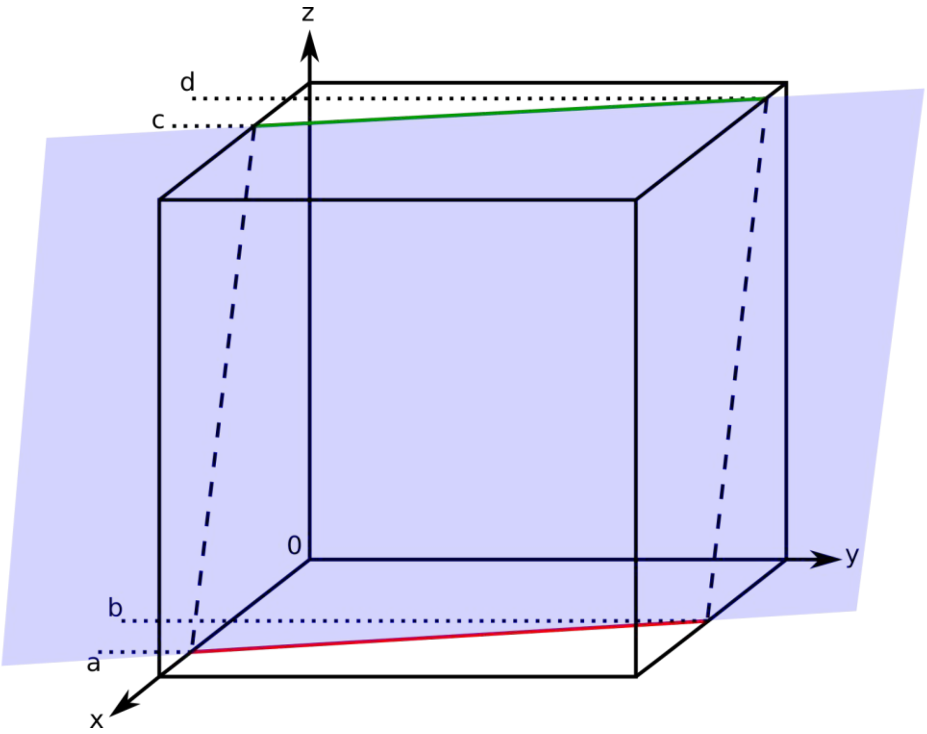


Figure A.2: Volume calculation for a cube cut by a plane.

Appendix B

Geometric Modelling of Deep Reactive Ion Etching

The geometric advection models introduced in Section 5.1.6.4 require some additional considerations due to their complexity. In the following, the necessary equations for the correct evaluation of the smooth profile, as well as the positioning of the lens distributions along the profile are derived and their effects on the advection are discussed.

B.1 Profile and Scallop Generation for the DEM Sequence

In this sequence, the depletion of etchant flux down the feature [261, 262] leads to a decrease in etch depth per cycle d_c. For simplicity, it is assumed that d_c is constant until a certain depth L_t, where tapering starts and d_c decreases linearly down the feature. Each distribution must be spaced by a distance d_c from the distribution above, which means that the scallops are more closely spaced down the feature. If the etch process is continued to infinity, the etch rate would, at some point, balance the deposition and d_c would go to zero. The depth at which d_c reaches zero is defined as L_0, which is used to find the distance D along which d_c decreases from its initial value to zero:

$$D = L_0 - L_t \tag{B.1}$$

Using L_t as the origin in the z-direction, the distance between the n^{th} and $(n+1)^{\text{th}}$ tapered scallop is

$$z_{n+1} - z_n = \left(1 - \frac{z_{n+1}}{D}\right)\frac{d_c}{2} + \left(1 - \frac{z_n}{D}\right)\frac{d_c}{2} \quad . \tag{B.2}$$

Rearranging Eq. (B.2) gives a recursive relation for the z coordinate of the centre of the n^{th} scallop:

$$z_{n+1} = \frac{d_c}{1 + \frac{d_c}{2D}} + \frac{1 - \frac{d_c}{2D}}{1 + \frac{d_c}{2D}} z_n = a + b z_n \quad . \tag{B.3}$$

Since tapering only starts a distance L_t down the trench, where the origin in z is placed, the first scallop is centred at $z_0 = 0$.

Eq. (B.3) is a geometric progression and therefore the origin of the n^{th} scallop may also be expressed as

$$z_n = a\frac{1-b^n}{1-b} \quad . \tag{B.4}$$

Therefore, if there are enough etch cycles to reach an etch rate of zero, the expression for D in Eq. (B.1) can be used to find the centres of all lens distributions.

However, experiments are not always conducted until the etch rate per cycle approaches zero, but are rather stopped after a certain number of cycles N_c. In this case, D can be found from the number of cycles and the ratio r_e of the etch depths d_c and d_f of the first and last cycle, respectively. This ratio is also closely related to the tapering width w_t and is given by

$$r_e = \frac{d_f}{d_c} = 1 - \frac{w_t}{w_{tot}} \quad , \tag{B.5}$$

where w_{tot} is the tapering width when the etch rate per cycle goes to zero. This final tapering width may also be geometry dependent in geometries with high aspect-ratios since it cannot exceed the radius of a via or half the width of a trench. Given either d_f and d_c or w_t and w_{tot}, r_e can be calculated and used to find D with the relation:

$$1 - r_e = \left(1 - \left(\frac{1-\frac{d_c}{2D}}{1+\frac{d_c}{2D}}\right)^{N_t}\right)\left(1 + \frac{d_c}{2D}\right) \quad . \tag{B.6}$$

D can therefore be found straight-forwardly using a root-finding algorithm, as it cannot be solved analytically. Since D is constant throughout the process, it only has to be calculated once, so the computational effort to find a numerical solution can be neglected.

This value of D is then used to find a and b defined in Eq. (B.3), which is used to find the values of all z_n. Then, given the number of etch cycles N_c to be performed, the final feature depth is given by

$$L_b = L_t + z_{N_t} = L_t + a\frac{1-b^{N_t}}{1-b} \quad , \tag{B.7}$$

which is used to generate the final smooth profile of the process.

B.2 DREAM Sequence Model

The closing of the top of the etched feature during the deposition cycles of the DREAM sequence can be avoided by properly tuning the ashing step. Otherwise, the closing leads to a decrease in etchant concentration reaching the bottom of the feature, leading to slowed etch rates. The step coverage n_r of a deep via is given by [289]

$$n_r = \frac{2e^{-h_T}}{1+e^{-2h_T}} \quad , \tag{B.8}$$

where h_T is the Thiele modulus describing the conformality of a process. For $h_T \ll 1$, the process is conformal, while for $h_T \gg 1$ the concentration of active particles down the feature varies strongly. For simple geometries, analytical expressions for h_T can be found, such as for a deep cylindrical via with diameter d:

$$h_T = \sqrt{3\beta}\,\frac{z}{d} \quad , \tag{B.9}$$

where β is the etchant sticking probability and z the vertical coordinate down the via.

When $h_T \approx 1$, the process is transitioning between being conformal and non-conformal, meaning that tapering will start at this point down the via. The etchant concentration in Eq. (B.8) at this point down the via can be approximated as

$$n_r \approx \mathrm{sech}(1)\,(1 - \tanh(1)(h_T - 1)) \quad , \tag{B.10}$$

where the etch rate in the via is directly proportional to this concentration. Therefore, for a given depth down the via z, the step coverage n_r depends only on the top opening of the via d. Since n_r and the etch rate ratio r_e defined in the previous section are equivalent, they can simply be written as

$$r_e = n_r \propto -\frac{1}{d} \quad . \tag{B.11}$$

If there is no ashing, as is the case for the DREM sequence, the top diameter d will decrease by a constant value for each cycle, leading to a linear decrease in d with time. Assuming the ashing removes passivating material at a constant rate, the closing rate of the top opening is also slowed linearly. Therefore, the opening diameter d at the top of the feature during the last cycle is directly proportional to the ash time t_a. If the ash time is increased, then also the final top opening diameter d is increased if all other process parameters stay the same so that $t_a \propto d$. Therefore, this linear model for the effect of ash time on the etch rate ratio is written as

$$r_e(t_a) = p_0 - \frac{p_1}{p_2 + t_a} \quad , \tag{B.12}$$

where $p0$, $p1$ and $p2$ are fitting parameters encompassing all physical parameters of the system. These also allow for a minimal ash time required to start removing passivating material, given as

$$t_0 = \frac{p_1}{p_0} - p_2 \quad , \tag{B.13}$$

as well as a maximum time, above which additional ashing does not have an effect, written as

$$t_m = \frac{p_1}{p_0 - 1} - p_2 \quad . \tag{B.14}$$

Model Fitting

The above model was fit to experimental data in [268] for 100 cycles of the DREAM process. In order to give the best approximation of Eq. (B.8) using Eq. (B.10), the depth L_t at which tapering is assumed to start in the model should be taken at the depth where $r_e = 0.96$, i.e. the depth at which the via diameter is 0.96 times the initial top opening diameter. From the experimental data, this depth was found to be $L_t = 24.96\,\mu\mathrm{m}$ from the top opening of the via. Since the etch depth per cycle was found to be $d_c = 0.37\,\mu\mathrm{m}$, the first 67 cycles were not influenced by the tapering. Measurements of the depths of vias for different ash times were taken to find values for D, which could be used to find experimental values of r_e for each ash time t_a. These measured values and the model fit are shown in Fig. B.1 with error bars indicating the measured values and the least squares fit shown in orange. From this fit, the parameters for the model could be extracted and were found as $p_0 = 1.17$, $p_1 = 0.59\,\mathrm{s}$ and $p_2 = -0.44\,\mathrm{s}$.

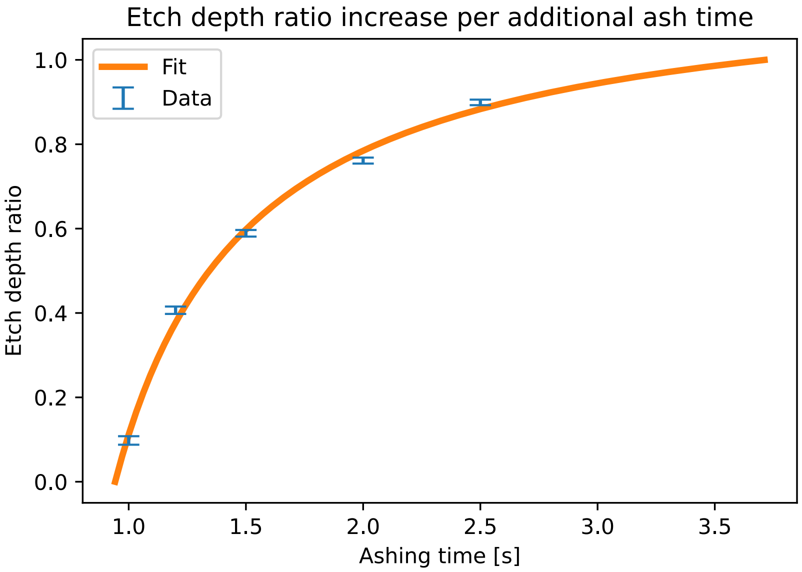


Figure B.1: Fit of the DREAM Sequence model to experimental values of the etch depth ratio generated from trench depths for different ash times of the DREAM sequence presented in [268]. Experimental values are shown as blue error bars and the fitted model is shown as an orange line going from $r_e = 0$ to $r_e = 1$.

Bibliography

[1] G. E. Moore, "Progress in digital integrated electronics," in *Proc. International Electron Devices Meeting (IEDM)*. IEEE, 1975, pp. 11–13, doi: 10.1109/N-SSC.2006.4804410.

[2] W. Shockley, "Circuit element utilizing semiconductive material," U.S. Patent 2 569 347, 1948, [Accessed: 2021, July]. [Online]. Available: https://patents.google.com/patent/US2569347A/en

[3] R. N. Noyce, "Semiconductor device-and-lead structure," U.S. Patent 2 981 877, 1959, doi: 10.1109/N-SSC.2007.4785577.

[4] C. D. Brock, *How Moore's Law came to be.* Computer History Museum, 2015, pp. 31–33, [Accessed: 2021, July]. [Online]. Available: http://s3data.computerhistory.org/core/core-2015.pdf

[5] T. N. Theis and H.-S. P. Wong, "The end of Moore's Law: A new beginning for information technology," *Computing in Science & Engineering*, vol. 19, no. 2, pp. 41–50, 2016, doi: 10.1109/MCSE.2017.29.

[6] B. J. Lin, "The future of subhalf-micrometer optical lithography," *Microelectronic Engineering*, vol. 6, no. 1-4, pp. 31–51, 1987, doi: 10.1016/0167-9317(87)90015-3.

[7] P. Chidambaram, C. Bowen, S. Chakravarthi, C. Machala, and R. Wise, "Fundamentals of silicon material properties for successful exploitation of strain engineering in modern CMOS manufacturing," *IEEE Transactions on Electron Devices*, vol. 53, no. 5, pp. 944–964, 2006, doi: 10.1109/TED.2006.872912.

[8] T. Skotnicki, J. Hutchby, T. King, H.-S. Wong, and F. Boeuf, "The end of CMOS scaling: Toward the introduction of new materials and structural changes to improve MOSFET performance," *IEEE Circuits and Devices Magazine*, vol. 21, no. 1, pp. 16–26, 2005, doi: 10.1109/MCD.2005.1388765.

[9] P. Batude *et al.*, "Advances in 3D CMOS sequential integration," in *Proc. International Electron Devices Meeting (IEDM)*. IEEE, 2009, pp. 1–4, doi: 10.1109/IEDM.2009.5424352.

[10] C.-t. Sah, R. Noyce, and W. Shockley, "Carrier generation and recombination in P-N junctions and P-N junction characteristics," *Proc. of the IRE*, vol. 45, no. 9, pp. 1228–1243, 1957, doi: 10.1109/JRPROC.1957.278528.

[11] H. Gummel, "A self-consistent iterative scheme for one-dimensional steady state transistor calculations," *IEEE Transactions on Electron Devices*, vol. 11, no. 10, pp. 455–465, 1964, doi: 10.1109/T-ED.1964.15364.

[12] Y. Y. Illarionov, M. I. Vexler, M. Karner, S. E. Tyaginov, J. Cervenka, and T. Grasser, "TCAD simulation of tunneling leakage current in $CaF_2/Si(111)$ MIS structures," *Current Applied Physics*, vol. 15, no. 2, pp. 78–83, 2015, doi: 10.1016/j.cap.2014.10.015.

[13] International Technology Roadmap for Semiconductors, 2015, [Accessed: 2021, July]. [Online]. Available: http://www.itrs2.net/

[14] A. R. Brown *et al.*, "From devices to circuits: Modelling the performance of 5 nm nanosheets," in *Proc. Simulation of Semiconductor Processes and Devices (SISPAD).* IEEE, 2019, pp. 1–4, doi: 10.1109/SISPAD.2019.8870357.

[15] R. J. Baker, *CMOS: Circuit Design, Layout, and Simulation.* John Wiley & Sons, 2019.

[16] C. Auth, "22-nm fully-depleted tri-gate CMOS transistors," in *Proc. Custom Integrated Circuits Conference (CICC).* IEEE, 2012, pp. 1–6, doi: 10.1109/CICC.2012.6330657.

[17] B. W. Smith, Y. Fan, M. Slocum, and L. Zavyalova, "25 nm immersion lithography at 193 nm wavelength," in *Proc. Optical Microlithography.* SPIE, 2005, p. 12, doi: 10.1117/12.602414.

[18] C. Bencher, Y. Chen, H. Dai, W. Montgomery, and L. Huli, "22 nm half-pitch patterning by CVD spacer self alignment double patterning (SADP)," in *Proc. Optical Microlithography*, vol. 6924. SPIE, 2008, p. 69244E, doi: 10.1117/12.772953.

[19] G. Yeap *et al.*, "5 nm CMOS production technology platform featuring full-fledged EUV, and high mobility channel FinFETs with densest 0.021 μm^2 SRAM cells for mobile SoC and high performance computing applications," in *Proc. International Electron Devices Meeting (IEDM).* IEEE, 2019, pp. 36.7.1–36.7.4, doi: 10.1109/IEDM19573.2019.8993577.

[20] ViennaTS - The Vienna Topography Simulator. [Online]. Available: https://github.com/viennats/viennats-dev

[21] S. Narasimha *et al.*, "22 nm high-performance SOI technology featuring dual-embedded stressors, epi-plate high-k deep-trench embedded DRAM and self-aligned via 15LM BEOL," in *Proc. International Electron Devices Meeting (IEDM).* IEEE, 2012, pp. 3.3.1–3.3.4, doi: 10.1109/IEDM.2012.6478971.

[22] Y. Nishi and R. Doering, *Handbook of Semiconductor Manufacturing Technology.* CRC Press, 2017, doi: 10.1201/9781420017663.

[23] R. Chau *et al.*, "Advanced metal gate/high-k dielectric stacks for high-performance CMOS transistors," in *Proc. International Conference of Microelectronics and Interfaces (ICMI)*, vol. 3. AVS, 2004, [Accessed: 2021, July]. [Online]. Available: https://www.intel.com/content/dam/doc/white-paper/high-k-gate-dielectrics-for-cmos-transistors-paper.pdf

[24] T.-Y. J. Chang *et al.*, "A 5-nm 135-Mb SRAM in EUV and high-mobility channel FinFET technology with metal coupling and charge-sharing write-assist circuitry schemes for high-density and low-V_{MIN} applications," *IEEE Journal of Solid-State Circuits*, vol. 56, no. 1, pp. 179–187, 2021, doi: 10.1109/JSSC.2020.3034241.

[25] A. Kostovic. (2020) GLOBALFOUNDRIES and GlobalWafers sign MOU to increase capacity, supply of 300 mm SOI wafers. [Accessed: 2021, July]. [Online]. Available: https://www.techpowerup.com/264207/globalfoundries-and-globalwafers-sign-mou-to-increase-capacity-supply-of-300mm-soi-wafers

[26] D. Pan, D. Guan, T.-C. Jen, and C. Yuan, “Atomic layer deposition process modeling and experimental investigation for sustainable manufacturing of nano thin films,” *Journal of Manufacturing Science and Engineering*, vol. 138, no. 10, pp. 101 010–1 – 101 010–9, 2016, doi: 10.1115/1.4034475.

[27] E. C. Nwanna, R. A. M. Coetzee, and T.-C. Jen, “Investigating the purge flow rate in a reactor scale simulation of an atomic layer deposition process,” in *Proc. International Mechanical Engineering Congress and Exposition (IMECE)*, vol. 2B. American Society of Mechanical Engineers, 2019, doi: 10.1115/IMECE2019-10692.

[28] Y. Zhang, Y. Ding, and P. D. Christofides, “Multiscale computational fluid dynamics modeling of thermal atomic layer deposition with application to chamber design,” *Chemical Engineering Research and Design*, vol. 147, pp. 529–544, 2019, doi: 10.1016/j.cherd.2019.05.049.

[29] P. Zheng, D. Connelly, F. Ding, and T.-J. K. Liu, “FinFET evolution toward stacked-nanowire FET for CMOS technology scaling,” *IEEE Transactions on Electron Devices*, vol. 62, no. 12, pp. 3945–3950, 2015, doi: 10.1109/TED.2015.2487367.

[30] D. J. Bell, T. J. Lu, N. A. Fleck, and S. M. Spearing, “MEMS actuators and sensors: Observations on their performance and selection for purpose,” *Journal of Micromechanics and Microengineering*, vol. 15, no. 7, pp. S153–S164, 2005, doi: 10.1088/0960-1317/15/7/022.

[31] L. Pelaz, L. Marques, M. Aboy, P. Lopez, I. Santos, and R. Duffy, “Atomistic process modeling based on kinetic monte carlo and molecular dynamics for optimization of advanced devices,” in *Proc. International Electron Devices Meeting (IEDM)*. IEEE, 2009, pp. 1–4, doi: 10.1109/IEDM.2009.5424309.

[32] P. Castrillo, R. Pinacho, M. Jaraiz, and J. E. Rubio, “Physical modeling and implementation scheme of native defect diffusion and interdiffusion in SiGe heterostructures for atomistic process simulation,” *Journal of Applied Physics*, vol. 109, no. 10, pp. 103 502–1 – 103 502–13, 2011, doi: 10.1063/1.3581113.

[33] L. Pelaz, L. A. Marqués, M. Aboy, P. López, and I. Santos, “Front-end process modeling in silicon,” *The European Physical Journal B*, vol. 72, no. 3, pp. 323–359, 2009, doi: 10.1140/epjb/e2009-00378-9.

[34] M. Shayan, “Study on atomistic model for simulation of anisotropic wet etching,” *Journal of Micro/Nanolithography, MEMS, and MOEMS*, vol. 10, no. 2, pp. 029 701–1 – 029 701–6, 2011, doi: 10.1117/1.3586798.

[35] M. Fermeglia, A. Mio, S. Aulic, D. Marson, E. Laurini, and S. Pricl, “Multiscale molecular modelling for the design of nanostructured polymer systems: Industrial applications,” *Molecular Systems Design & Engineering*, vol. 5, no. 9, pp. 1447–1476, 2020, doi: 10.1039/D0ME00109K.

[36] R. Bergamaschini, M. Salvalaglio, R. Backofen, A. Voigt, and F. Montalenti, “Continuum modelling of semiconductor heteroepitaxy: An applied perspective,” *Advances in Physics: X*, vol. 1, no. 3, pp. 331–367, 2016, doi: 10.1080/23746149.2016.1181986.

[37] N. Zographos, C. Zechner, I. Martin-Bragado, K. Lee, and Y.-S. Oh, “Multiscale modeling of doping processes in advanced semiconductor devices,” *Materials Science in Semiconductor Processing*, vol. 62, pp. 49–61, 2017, doi: 10.1016/j.mssp.2016.10.037.

[38] T. Thurgate, “Segment-based etch algorithm and modeling,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 10, no. 9, pp. 1101–1109, 1991, doi: 10.1109/43.85756.

[39] J. Bigler, A. Stephens, and S. Parker, “Design for parallel interactive ray tracing systems,” in *Proc. Interactive Ray Tracing (RT)*. IEEE, 2006, pp. 187–196, doi: 10.1109/RT.2006.280230.

[40] A. Agathos, I. Pratikakis, S. Perantonis, N. Sapidis, and P. Azariadis, “3D mesh segmentation methodologies for CAD applications,” *Computer-Aided Design and Applications*, vol. 4, no. 6, pp. 827–841, 2007, doi: 10.1080/16864360.2007.10738515.

[41] Cha Zhang and Tsuhan Chen, “Efficient feature extraction for 2D/3D objects in mesh representation,” in *Proc. International Conference on Image Processing (ICIP)*, vol. 2. IEEE, 2001, pp. 935–938, doi: 10.1109/ICIP.2001.958278.

[42] S. Osher and J. A. Sethian, “Fronts propagating with curvature-dependent speed: Algorithms based on hamilton-jacobi formulations,” *Journal of Computational Physics*, vol. 79, no. 1, pp. 12–49, 1988, doi: 10.1016/0021-9991(88)90002-2.

[43] J. Bloomenthal *et al.*, *Introduction to Implicit Surfaces*. Morgan Kaufmann Publishers, 1997.

[44] J. Sethian, *Level Set Methods and Fast Marching Methods: Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision, and Materials Science*. Cambridge University Press, 1999.

[45] M. Pauly, M. Gross, and L. P. Kobbelt, “Efficient simplification of point-sampled surfaces,” in *Proc. Visualization (VIS)*. IEEE, 2002, pp. 163–170, doi: 10.1109/VISUAL.2002.1183771.

[46] D. L. Chopp, “Computing minimal surfaces via level set curvature flow,” *Journal of Computational Physics*, vol. 106, no. 1, pp. 77–91, 1993, doi: 10.1006/jcph.1993.1092.

[47] D. Adalsteinsson and J. A. Sethian, “A fast level set method for propagating interfaces,” *Journal of Computational Physics*, vol. 118, no. 2, pp. 269–277, 1995, doi: 10.1006/jcph.1995.1098.

[48] S. Osher and R. Fedkiw, *Level Set Methods and Dynamic Implicit Surfaces*. Springer, 2003, vol. 153.

[49] J. A. Sethian, “A fast marching level set method for monotonically advancing fronts,” *Proceedings of the National Academy of Sciences*, vol. 93, no. 4, pp. 1591–1595, 1996, doi: 10.1073/pnas.93.4.1591.

[50] J. Tsitsiklis, “Efficient algorithms for globally optimal trajectories,” *IEEE Transactions on Automatic Control*, vol. 40, no. 9, pp. 1528–1538, 1995, doi: 10.1109/9.412624.

[51] Y.-h. R. Tsai, “Rapid and accurate computation of the distance function using grids,” *Journal of Computational Physics*, vol. 178, no. 1, pp. 175–195, 2002, doi: 10.1006/jcph.2002.7028.

[52] S. Kim, “An $\mathcal{O}(n)$ level set method for eikonal equations,” *SIAM Journal on Scientific Computing*, vol. 22, no. 6, pp. 2178–2193, 2001, doi: 10.1137/S1064827500367130.

[53] L. Yatziv, A. Bartesaghi, and G. Sapiro, “$\mathcal{O}(n)$ implementation of the fast marching algorithm,” *Journal of Computational Physics*, vol. 212, no. 2, pp. 393–399, 2006, doi: 10.1016/j.jcp.2005.08.005.

[54] H. K. Zhao, “Fast sweeping method for eikonal equations,” *Mathematics of Computation*, vol. 74, no. 250, pp. 603–627, 2005, doi: 10.1090/S0025-5718-04-01678-3.

[55] C. Y. Kao, S. Osher, and J. Qian, “Lax–Friedrichs sweeping scheme for static Hamilton–Jacobi equations,” *Journal of Computational Physics*, vol. 196, no. 1, pp. 367–391, 2004, doi: 10.1016/j.jcp.2003.11.007.

[56] H. Zhao, “Parallel implementations of the fast sweeping method,” *Journal of Computational Mathematics*, pp. 421–429, 2007. [Online]. Available: http://www.jstor.org/stable/43693378

[57] W.-K. Jeong and R. T. Whitaker, “A fast iterative method for eikonal equations,” *SIAM Journal on Scientific Computing*, vol. 30, no. 5, pp. 2512–2534, 2008, doi: 10.1137/060670298.

[58] R. T. Whitaker, “A level-set approach to 3D reconstruction from range data,” *International Journal of Computer Vision*, vol. 29, no. 3, pp. 203–231, 1998, doi: 10.1023/A:1008036829907.

[59] R. Malladi and J. A. Sethian, “Image processing via level set curvature flow,” *Proceedings of the National Academy of Sciences*, vol. 92, no. 15, pp. 7046–7050, 1995, doi: 10.1073/pnas.92.15.7046.

[60] O. Ertl and S. Selberherr, “Three-dimensional topography simulation using advanced level set and ray tracing methods,” in *Proc. Simulation of Semiconductor Processes and Devices (SISPAD)*. IEEE, 2008, pp. 325–328, doi: 10.1109/SISPAD.2008.4648303.

[61] R. Goldman, “Curvature formulas for implicit curves and surfaces,” *Computer Aided Geometric Design*, vol. 22, no. 7, pp. 632–658, 2005, doi: 10.1016/j.cagd.2005.06.005.

[62] Y. Qin *et al.*, “Research and application of boolean operation for triangular mesh model of underground space engineering—boolean operation for triangular mesh model,” *Energy Science & Engineering*, vol. 7, no. 4, pp. 1154–1165, 2019, doi: 10.1002/ese3.335.

[63] A. Pasko, V. Adzhiev, A. Sourin, and V. Savchenko, “Function representation in geometric modeling: Concepts, implementation and applications,” *The Visual Computer*, vol. 11, no. 8, pp. 429–446, 1995, doi: 10.1007/BF02464333.

[64] B. Wyvill, A. Guy, and E. Galin, “Extending the CSG tree. Warping, blending and boolean operations in an implicit surface modeling system,” *Computer Graphics Forum*, vol. 18, no. 2, pp. 149–158, 1999, doi: 10.1111/1467-8659.00365.

[65] A. Requicha and H. Voelcker, “Boolean operations in solid modeling: Boundary evaluation and merging algorithms,” *Proceedings of the IEEE*, vol. 73, no. 1, pp. 30–44, 1985, doi: 10.1109/PROC.1985.13108.

[66] A. V. Kumar, S. Padmanabhan, and R. Burla, “Implicit boundary method for finite element analysis using non-conforming mesh or grid,” *International Journal for Numerical Methods in Engineering*, vol. 74, no. 9, pp. 1421–1447, 2008, doi: 10.1002/nme.2216.

[67] G. S. Oehrlein, “Study of sidewall passivation and microscopic silicon roughness phenomena in chlorine-based reactive ion etching of silicon trenches,” *Journal of Vacuum Science & Technology B: Microelectronics and Nanometer Structures*, vol. 8, no. 6, pp. 1199–1211, 1990, doi: 10.1116/1.584896.

[68] S. M. George, “Atomic layer deposition: An overview,” *Chemical Reviews*, vol. 110, no. 1, pp. 111–131, 2010, doi: 10.1021/cr900056b.

[69] N. Boukortt, B. Hadri, S. Patanè, A. Caddemi, and G. Crupi, “Investigation on TG n-FinFET parameters by varying channel doping concentration and gate length,” *Silicon*, vol. 9, no. 6, pp. 885–893, 2017, doi: 10.1007/s12633-016-9528-3.

[70] S. R. Suddapalli and B. R. Nistala, “Analog/RF performance of graded channel gate stack triple material double gate strained-Si MOSFET with fixed charges,” *Silicon*, 2021, doi: 10.1007/s12633-021-01028-0.

[71] Q. Du and D. Wang, “Tetrahedral mesh generation and optimization based on centroidal Voronoi tessellations,” *International Journal for Numerical Methods in Engineering*, vol. 56, no. 9, pp. 1355–1373, 2003, doi: 10.1002/nme.616.

[72] Z. Wang *et al.*, “3-D local mesh refinement XFEM with variable-node hexahedron elements for extraction of stress intensity factors of straight and curved planar cracks,” *Computer Methods in Applied Mechanics and Engineering*, vol. 313, pp. 375–405, 2017, doi: 10.1016/j.cma.2016.10.011.

[73] K. Wang, K. Adimulam, and T. Kesavadas, “Tetrahedral mesh visualization in a game engine,” in *Proc. Virtual Reality and 3D User Interfaces (VR)*. IEEE, 2018, pp. 719–720, doi: 10.1109/VR.2018.8446544.

[74] K. Yagi, S. Tanaka, T. Kawahara, K. Nihei, H. Okada, and N. Osawa, “Evaluation of crack propagation behaviors in a T-shaped tubular joint employing tetrahedral FE modeling,” *International Journal of Fatigue*, vol. 96, pp. 270–282, 2017, doi: 10.1016/j.ijfatigue.2016.11.028.

[75] A. Makarov *et al.*, “Hot-carrier degradation in FinFETs: Modeling, peculiarities, and impact of device topology,” in *Proc. International Electron Devices Meeting (IEDM)*. IEEE, 2017, pp. 13.1.1–13.1.4, doi: 10.1109/IEDM.2017.8268381.

[76] H. Si, “TetGen, a delaunay-based quality tetrahedral mesh generator,” *ACM Transactions on Mathematical Software*, vol. 41, no. 2, pp. 1–36, 2015, doi: 10.1145/2629697.

[77] M. Lee, L. De Floriani, and H. Samet, “Constant-time neighbor finding in hierarchical tetrahedral meshes,” in *Proc. International Conference on Shape Modeling and Applications (SMA)*. IEEE, 2001, pp. 286–295, doi: 10.1109/SMA.2001.923400.

[78] E. Strasser and S. Selberherr, “Algorithms and models for cellular based topography simulation,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 14, no. 9, pp. 1104–1114, 1995, doi: 10.1109/43.406712.

[79] Y. Zhang, C. Huard, S. Sriraman, J. Belen, A. Paterson, and M. J. Kushner, “Investigation of feature orientation and consequences of ion tilting during plasma etching with a three-dimensional feature profile simulator,” *Journal of Vacuum Science & Technology A: Vacuum, Surfaces, and Films*, vol. 35, no. 2, pp. 021 303–1–021 303–16, 2017, doi: 10.1116/1.4968392.

[80] C. M. Huard, “Nano-Scale Feature Profile Modeling of Plasma Material Processing,” Ph.D. dissertation, University of Michigan, 2018, [Accessed: 2021, July]. [Online]. Available: https://ui.adsabs.harvard.edu/abs/2018PhDT.......110H

[81] K. Toh, A. Neureuther, and E. Scheckler, “Algorithms for simulation of three-dimensional etching,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 13, no. 5, pp. 616–624, 1994, doi: 10.1109/43.277635.

[82] M. Fujinaga and N. Kotani, “3-D topography simulator (3-D MULSS) based on a physical description of material topography,” *IEEE Transactions on Electron Devices*, vol. 44, no. 2, pp. 226–238, 1997, doi: 10.1109/16.557710.

[83] Z.-F. Zhou, Q.-A. Huang, W.-H. Li, and W. Lu, “A novel 3-D dynamic cellular automata model for photoresist-etching process simulation,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 26, no. 1, pp. 100–114, 2007, doi: 10.1109/TCAD.2006.882510.

[84] M. E. Law and S. M. Cea, “Continuum based modeling of silicon integrated circuit processing: An object oriented approach,” *Computational Materials Science*, vol. 12, no. 4, pp. 289–308, 1998, doi: 10.1016/S0927-0256(98)00020-2.

[85] P. Alliez, E. de Verdire, O. Devillers, and M. Isenburg, “Isotropic surface remeshing,” in *Proc. Shape Modelling International (SMI)*, vol. 2003. IEEE, 2003, pp. 49–58, doi: 10.1109/SMI.2003.1199601.

[86] J. C. Butcher and N. Goodwin, *Numerical Methods for Ordinary Differential Equations*. Wiley Online Library, 2008, vol. 2.

[87] A. Harten, P. D. Lax, and B. van Leer, “On upstream differencing and godunov-type schemes for hyperbolic conservation laws,” *SIAM Review*, vol. 25, no. 1, pp. 35–61, 1983, doi: 10.1137/1025002.

[88] P. L. Roe, “Approximate riemann solvers, parameter vectors, and difference schemes,” *Journal of Computational Physics*, vol. 43, pp. 357–372, 1981, doi: doi: 10.1016/0021-9991(81)90128-5.

[89] C.-W. Shu, “Total-variation-diminishing time discretizations,” *SIAM Journal on Scientific and Statistical Computing*, vol. 9, no. 6, pp. 1073–1084, 1988, doi: 10.1137/0909073.

[90] R. Malladi, J. A. Sethian, and B. C. Vemuri, “Shape modeling with front propagation: A level set approach,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, no. 2, pp. 158–175, 1995, doi: 10.1109/34.368173.

[91] D. Adalsteinsson and J. A. Sethian, "The fast construction of extension velocities in level set methods," *Journal of Computational Physics*, vol. 148, no. 1, pp. 2–22, 1999, doi: 10.1006/jcph.1998.6090.

[92] R. Courant, K. Friedrichs, and H. Lewy, "Über die partiellen Differenzengleichungen der mathematischen Physik," *Mathematische Annalen*, vol. 100, no. 1, pp. 32–74, 1928, doi: 10.1007/BF01448839.

[93] R. Courant, E. Isaacson, and M. Rees, "On the solution of nonlinear hyperbolic differential equations by finite differences," *Communications on Pure and Applied Mathematics*, vol. 5, no. 3, pp. 243–255, 1952, doi: 10.1002/cpa.3160050303.

[94] B. Engquist and S. Osher, "Stable and entropy satisfying approximations for transonic flow calculations," *Mathematics of Computation*, vol. 34, no. 149, p. 45, 1980, doi: 10.2307/2006220.

[95] B. Radjenović, J. K. Lee, and M. Radmilović-Radjenović, "Sparse field level set method for non-convex hamiltonians in 3D plasma etching profile simulations," *Computer Physics Communications*, vol. 174, no. 2, pp. 127–132, 2006, doi: 10.1016/j.cpc.2005.09.010.

[96] S. Osher and C.-W. Shu, "High-order essentially nonoscillatory schemes for Hamilton-Jacobi equations," *SIAM Journal on Numerical Analysis*, vol. 28, no. 4, pp. 907–922, 1991, doi: 10.1137/0728049.

[97] R. J. LeVeque *et al.*, *Finite Volume Methods for Hyperbolic Problems.* Cambridge University Press, 2002, vol. 31, doi: 10.1017/CBO9780511791253.

[98] M. G. Crandall and P. L. Lions, "Two approximations of solutions of hamilton-jacobi equations," *Mathematics of Computation*, vol. 43, no. 167, p. 1, 1984, doi: 10.2307/2007396.

[99] C. Montoliu, N. Ferrando, M. Gosálvez, J. Cerdá, and R. Colom, "Implementation and evaluation of the level set method: Towards efficient and accurate simulation of wet etching for microengineering applications," *Computer Physics Communications*, vol. 184, no. 10, pp. 2299–2309, 2013, doi: 10.1016/j.cpc.2013.05.016.

[100] A. Toifl *et al.*, "The level-set method for multi-material wet etching and non-planar selective epitaxy," *IEEE Access*, vol. 8, pp. 115 406–115 422, 2020, doi: 10.1109/ACCESS.2020.3004136.

[101] W. H. Press, H. William, S. A. Teukolsky, W. T. Vetterling, A. Saul, and B. P. Flannery, *Numerical Recipes 3rd Edition: The Art of Scientific Computing.* Cambridge University Press, 2007.

[102] A. Harten and S. Osher, "Uniformly high-order accurate nonoscillatory schemes. I," *SIAM Journal on Numerical Analysis*, vol. 24, no. 2, pp. 279–309, 1987, doi: 10.1137/0724022.

[103] C.-W. Shu and S. Osher, "Efficient implementation of essentially non-oscillatory shock-capturing schemes," *Journal of Computational Physics*, vol. 77, no. 2, pp. 439–471, 1988, doi: 10.1016/0021-9991(88)90177-5.

[104] C.-W. Shu and S. Osher, "Efficient implementation of essentially non-oscillatory shock-capturing schemes, II," in *Upwind and High-Resolution Schemes.* Springer, 1989, pp. 328–374, doi: 10.1016/0021-9991(89)90222-2.

[105] M. F. Trujillo, L. Anumolu, and D. Ryddner, “The distortion of the level set gradient under advection,” *Journal of Computational Physics*, vol. 334, pp. 81–101, 2017, doi: 10.1016/j.jcp.2016.11.050.

[106] X. Klemenschits, S. Selberherr, and L. Filipovic, “Geometric advection algorithm for process emulation,” in *Proc. Simulation of Semiconductor Processes and Devices (SISPAD)*. IEEE, 2020, pp. 59–62, doi: 10.23919/SISPAD49475.2020.9241678.

[107] P. Pfäffli *et al.*, “TCAD modeling for reliability,” *Microelectronics Reliability*, vol. 88-90, pp. 1083–1089, 2018, doi: 10.1016/j.microrel.2018.06.109.

[108] Z. Tan, M. Furmanczyk, M. Turowski, and A. J. Przekwas, “CFD-micromesh: A fast geometric modeling and mesh generation tool for 3D microsystem simulations,” in *Proc. Design, Test, Integration, and Packaging of MEMS/MOEMSS (DTIP)*. SPIE, 2000, pp. 193–199, doi: 10.1117/12.382289.

[109] C. M. Huard, Y. Zhang, S. Sriraman, A. Paterson, and M. J. Kushner, “Role of neutral transport in aspect ratio dependent plasma etching of three-dimensional features,” *Journal of Vacuum Science & Technology A: Vacuum, Surfaces, and Films*, vol. 35, no. 5, pp. 05C301–1 – 05C301–18, 2017, doi: 10.1116/1.4973953.

[110] X. Klemenschits, S. Selberherr, and L. Filipovic, “Unified feature scale model for etching in SF_6 and Cl plasma chemistries,” in *Proc. EUROSOI Workshop and International Conference on Ultimate Integration on Silicon (EUROSOI-ULIS)*. IEEE, 2018, pp. 1–4, doi: 10.1109/ULIS.2018.8354763.

[111] L. Wang, A. R. Brown, B. Cheng, and A. Asenov, “Simulation of 3D FinFET doping profiles by ion implantation,” in *Proc. Ion Implantation Technology*. AIP, 2012, pp. 217–220, doi: 10.1063/1.4766527.

[112] J.-Y. Park, G.-B. Lee, and Y.-K. Choi, “A comparative study of the curing effects of local and global thermal annealing on a FinFET,” *IEEE Journal of the Electron Devices Society*, vol. 7, pp. 954–958, 2019, doi: 10.1109/JEDS.2019.2937802.

[113] D. Andriukaitis, R. Anilionis, and T. Kersys, “LOCOS CMOS process simulation,” in *Proc. Information Technology Interfaces (ITI)*. IEEE, 2006, pp. 489–494, doi: 10.1109/ITI.2006.1708530.

[114] Y. H. Lee, “Silicon doping effects in reactive plasma etching,” *Journal of Vacuum Science & Technology B: Microelectronics and Nanometer Structures*, vol. 4, no. 2, pp. 468–475, 1986, doi: 10.1116/1.583405. . [Online]. Available: http://scitation.aip.org/content/avs/journal/jvstb/4/2/10.1116/1.583405

[115] M. Uematsu, H. Kageshima, K. Shiraishi, M. Nagase, S. Horiguchi, and Y. Takahashi, “Two-dimensional simulation of pattern-dependent oxidation of silicon nanostructures on silicon-on-insulator substrates,” *Solid-State Electronics*, vol. 48, no. 6, pp. 1073–1078, 2004, doi: 10.1016/j.sse.2003.12.019.

[116] G. Schröpfer, D. King, C. Kennedy, and M. McNie, “Advanced process emulation and circuit simulation for co-design of MEMS and CMOS devices,” in *Proc. Design, Test, Integration, and Packaging of MEMS/MOEMSS (DTIP)*. SPIE, 2005, [Accessed: 2021, July]. [Online]. Available: https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.118.4615&rep=rep1&type=pdf

[117] B. Vianne *et al.*, “Investigations on contact punch-through in 28 nm fdsoi through virtual fabrication,” in *Proc. SOI-3D-Subthreshold Microelectronics Technology Unified (S3S)*, vol. 2018-March. IEEE, 2017, pp. 1–2, doi: 10.1109/S3S.2017.8309236.

[118] J.-H. Franke, M. Gallagher, G. Murdoch, S. Halder, A. Juncker, and W. Clark, “EPE analysis of sub-N10 BEoL flow with and without fully self-aligned via using Coventor SEMulator3D,” in *Proc. Metrology, Inspection, and Process Control for Microlithography*, vol. 10145. SPIE, 2017, pp. 1 014 529–1–1 014 529–10, doi: 10.1117/12.2258195.

[119] G. Murdoch *et al.*, “Feasibility study of fully self aligned vias for 5 nm node BEOL,” in *Proc. International Interconnect Technology Conference (IITC)*. IEEE, 2017, pp. 1–4, doi: 10.1109/IITC-AMC.2017.7968958.

[120] M. J. Chopra, X. Zhu, Z. Z. Zhang, S. Helpert, R. Verma, and R. Bonnecaze, “A model-based, Bayesian approach to the CF_4/Ar etch of SiO_2,” in *Proc. Design-Process-Technology Co-optimization for Manufacturability*. SPIE, 2018, p. 15, doi: 10.1117/12.2297482.

[121] J. Yeom, Y. Wu, J. C. Selby, and M. A. Shannon, “Maximum achievable aspect ratio in deep reactive ion etching of silicon due to aspect ratio dependent transport and the microloading effect,” *Journal of Vacuum Science & Technology B: Microelectronics and Nanometer Structures*, vol. 23, no. 6, pp. 2319 – 2329, 2005, doi: 10.1116/1.2101678.

[122] W. Vincenti and C. Kruger, *Introduction to Physical Gas Dynamics*. Krieger Publishing Company, 1967.

[123] J. D. Plummer, M. D. Deal, and P. B. Griffin, *Silicon VLSI Technology - Fundamentals, Practice and Modeling*. Prentice Hall, 2000.

[124] B. Cossou *et al.*, “Synthesis and optimization of low-pressure chemical vapor deposition-silicon nitride coatings deposited from $SiHCl_3$ and NH_3,” *Thin Solid Films*, vol. 681, pp. 47–57, 2019, doi: 10.1016/j.tsf.2019.04.045.

[125] H. Setyawan, M. Shimada, K. Ohtsuka, and K. Okuyama, “Visualization and numerical simulation of fine particle transport in a low-pressure parallel plate chemical vapor deposition reactor,” *Chemical Engineering Science*, vol. 57, no. 3, pp. 497–506, 2002, doi: 10.1016/S0009-2509(01)00373-6.

[126] T. P. Merchant, M. K. Gobbert, T. S. Cale, and L. J. Borucki, “Multiple scale integrated modeling of deposition processes,” *Thin Solid Films*, vol. 365, no. 2, pp. 368–375, 2000, doi: 10.1016/S0040-6090(99)01055-X.

[127] J.-H. Kim, S.-W. Cho, C. J. Park, H. Chae, and C.-K. Kim, “Angular dependences of SiO_2 etch rates at different bias voltages in CF_4, C_2F_6, and C_4F_8 plasmas,” *Thin Solid Films*, vol. 637, pp. 43–48, 2017, doi: 10.1016/j.tsf.2017.03.047.

[128] R. A. Gottscho, “Microscopic uniformity in plasma etching,” *Journal of Vacuum Science & Technology B: Microelectronics and Nanometer Structures*, vol. 10, no. 5, pp. 2133–2147, 1992, doi: 10.1116/1.586180.

[129] R. A. Gottscho, “Ion transport anisotropy in low pressure, high density plasmas,” *Journal of Vacuum Science & Technology B: Microelectronics and Nanometer Structures*, vol. 11, no. 5, pp. 1884 – 1889, 1993, doi: 10.1116/1.586516.

[130] H.-B. Kim, G. Hobler, A. Steiger, A. Lugstein, and E. Bertagnolli, “Full three-dimensional simulation of focused ion beam micro/nanofabrication,” *Nanotechnology*, vol. 18, no. 24, pp. 245 303–1 – 245 303–9, 2007, doi: 10.1088/0957-4484/18/24/245303.

[131] E. Wagner *et al.*, “Geometry of chemical beam vapor deposition system for efficient combinatorial investigations of thin oxide films: Deposited film properties versus precursor flow simulations,” *ACS Combinatorial Science*, vol. 18, no. 3, pp. 154–161, 2016, doi: 10.1021/acscombsci.5b00146.

[132] G. Teeter, “Conceptual design of a deposition system for uniform and combinatorial synthesis of multinary thin-film materials via open-boat physical-vapor deposition,” *Journal of Vacuum Science & Technology A: Vacuum, Surfaces, and Films*, vol. 24, no. 4, pp. 1119–1127, 2006, doi: 10.1116/1.2208991.

[133] J.-L. Vassent, A. Marty, B. Gilles, and C. Chatillon, “Angular distribution of molecular beams and homogeneous layer growth: Optimization of geometrical parameters in molecular beam epitaxy,” *Vacuum*, vol. 64, no. 1, pp. 65–85, 2001, doi: 10.1016/S0042-207X(01)00376-1.

[134] J. Ou and Y. Huang, “Ion energy distribution in a radio frequency sheath of plasma with kappa electron velocity distribution,” *Contributions to Plasma Physics*, vol. 61, no. 2, pp. e202 000 108–1 – e202 000 108–8, 2021, doi: 10.1002/ctpp.202000108.

[135] S. Chatterjee, “Prediction of step coverage during blanket CVD tungsten deposition in cylindrical pores,” *Journal of The Electrochemical Society*, vol. 137, no. 1, pp. 328–335, 1990, doi: 10.1149/1.2086413.

[136] G. B. Raupp and T. S. Cale, “Step coverage prediction in low-pressure chemical vapor deposition,” *Chemistry of Materials*, vol. 1, no. 2, pp. 207–214, 1989, doi: 10.1021/cm00002a009.

[137] M. Ylilammi, O. M. E. Ylivaara, and R. L. Puurunen, “Modeling growth kinetics of thin films made by atomic layer deposition in lateral high-aspect-ratio structures,” *Journal of Applied Physics*, vol. 123, no. 20, pp. 205 301–1 – 205 301–8, 2018, doi: 10.1063/1.5028178.

[138] R. L. Cook, “Shade trees,” in *Proc. Computer Graphics and Interactive Techniques (SIGGRAPH)*. ACM Press, 1984, pp. 223–231, doi: 10.1145/800031.808602.

[139] I. Wald *et al.*, “State of the art in ray tracing animated scenes,” *Computer Graphics Forum*, vol. 28, no. 6, pp. 1691–1722, 2009, doi: 10.1111/j.1467-8659.2008.01313.x.

[140] S. Parker, M. Parker, Y. Livnat, P. P. Sloan, C. Hansen, and P. Shirley, “Interactive ray tracing for volume visualization,” in *Proc. Computer Graphics and Interactive Techniques (SIGGRAPH)*. ACM, 2005, pp. 15–1–15–13, doi: 10.1145/1198555.1198754.

[141] T. J. Purcell, I. Buck, W. R. Mark, and P. Hanrahan, “Ray tracing on programmable graphics hardware,” in *Proc. Computer Graphics and Interactive Techniques (SIGGRAPH)*. ACM, 2002, pp. 703–712, doi: 10.1145/566570.566640.

[142] P. L. O’Sullivan, F. H. Baumann, and G. H. Gilmer, “Simulation of physical vapor deposition into trenches and vias: Validation and comparison with experiment,” *Journal of Applied Physics*, vol. 88, no. 7, pp. 4061 – 4068, 2000, doi: 10.1063/1.1310182.

[143] J. Singh and P. Narayanan, “Real-time ray tracing of implicit surfaces on the GPU,” *IEEE Transactions on Visualization and Computer Graphics*, vol. 16, no. 2, pp. 261–272, 2010, doi: 10.1109/TVCG.2009.41.

[144] J. C. Hart, *SIGGRAPH Course Notes: Ray tracing implicit surfaces.* Association for Computing Machinery, 1993, vol. 1, [Accessed: 2021, July]. [Online]. Available: https://infoguides.rit.edu/computer-graphics/coursenotes

[145] J. C. Hart, “Sphere tracing: A geometric method for the antialiased ray tracing of implicit surfaces,” *The Visual Computer*, vol. 12, no. 10, pp. 527–545, 1996, doi: 10.1007/s003710050084.

[146] D. DeMarle, S. Parker, M. Hartner, C. Gribble, and C. Hansen, “Distributed interactive ray tracing for large volume visualization,” *IEEE Sensors Journal*, pp. 87–94, 2003, doi: 10.1109/PVGS.2003.1249046.

[147] P. Manstetten, J. Weinbub, A. Hössinger, and S. Selberherr, “Using temporary explicit meshes for direct flux calculation on implicit surfaces,” *Procedia Computer Science*, vol. 108, pp. 245–254, 2017, doi: 10.1016/j.procs.2017.05.067.

[148] J.-C. Yu *et al.*, “Three-dimensional simulation of DRIE process based on the narrow band level set and monte carlo method,” *Micromachines*, vol. 9, no. 2, pp. 74–1 – 74–12, 2018, doi: 10.3390/mi9020074.

[149] J. C. Arnold and H. H. Sawin, “Charging of pattern features during plasma etching,” *Journal of Applied Physics*, vol. 70, no. 10, pp. 5314–5317, 1991, doi: 10.1063/1.350241.

[150] Seong-Hyok Kim, Sang-Hun Lee, Hyung-Taek Lim, Yong-Kweon Kim, and Seung-Ki Lee, “[110] silicon etching for high aspect ratio comb structures,” in *Proc. Conference on Emerging Technologies and Factory Automation (EFTA).* IEEE, 1997, pp. 248–252, doi: 10.1109/ETFA.1997.616277.

[151] H. Liao and T. S. Cale, “Three-dimensional simulation of an isolation trench refill process,” *Thin Solid Films*, vol. 236, no. 1-2, pp. 352–358, 1993, doi: 10.1016/0040-6090(93)90695-L.

[152] D. Adalsteinsson and J. A. Sethian, “A level set approach to a unified model for etching, deposition, and lithography,” *Journal of Computational Physics*, vol. 138, no. 1, pp. 193–223, 1997, doi: 10.1006/jcph.1997.5817.

[153] D. Strnad, “Parallel terrain visibility calculation on the graphics processing unit,” *Concurrency and Computation: Practice and Experience*, vol. 23, no. 18, pp. 2452–2462, 2011, doi: 10.1002/cpe.1808.

[154] P. Manstetten, “Efficient flux calculations for topography simulation,” Ph.D. dissertation, 2018, doi: 10.34726/hss.2018.57263.

[155] N. Cheimarios, G. Kokkoris, and A. G. Boudouvis, “Multiscale modeling in chemical vapor deposition processes: Coupling reactor scale with feature scale computations,” *Chemical Engineering Science*, vol. 65, no. 17, pp. 5018–5028, 2010, doi: 10.1016/j.ces.2010.06.004.

[156] B. Abraham-Shrauner, “Analytic models for plasma-assisted etching of semiconductor trenches,” *Journal of Vacuum Science and Technology B: Microelectronics and Nanometer Structures*, vol. 12, no. 4, pp. 2347 – 2351, 1994, doi: 10.1116/1.587762.

[157] M. Tuda, K. Nishikawa, and K. Ono, “Numerical study of the etch anisotropy in low-pressure, high-density plasma etching,” *Journal of Applied Physics*, vol. 81, no. 2, pp. 960–967, 1997, doi: 10.1063/1.364189.

[158] D. C. Gray, “Phenomenological modeling of ion-enhanced surface kinetics in fluorine-based plasma etching,” *Journal of Vacuum Science & Technology B: Microelectronics and Nanometer Structures*, vol. 11, no. 4, pp. 1243 – 1257, 1993, doi: 10.1116/1.586925.

[159] R. A. Barker, “Surface studies of and a mass balance model for Ar+ ion-assisted Cl_2 etching of Si,” *Journal of Vacuum Science & Technology B: Microelectronics and Nanometer Structures*, vol. 1, no. 1, pp. 37 – 42, 1983, doi: 10.1116/1.582539.

[160] A. L. Magna and G. Garozzo, “Factors affecting profile evolution in plasma etching of SiO_2,” *Journal of The Electrochemical Society*, vol. 150, no. 10, pp. F178 – F185, 2003, doi: 10.1149/1.1602084.

[161] D. J. Cooperberg, V. Vahedi, and R. A. Gottscho, “Semiempirical profile simulation of aluminum etching in a Cl_2/BCl_3 plasma,” *Journal of Vacuum Science & Technology A: Vacuum, Surfaces, and Films*, vol. 20, no. 5, pp. 1536–1556, 2002, doi: 10.1116/1.1494818.

[162] O. Ertl and S. Selberherr, “Three-dimensional plasma etching simulation using advanced ray tracing and level set techniques,” in *Proc. Microelectronics Technology and Devices (SBMicro)*. ECS, 2009, pp. 61–68, doi: 10.1149/1.3183702.

[163] X. Klemenschits, O. Ertl, P. Manstetten, J. Weinbub, and L. Filipovic. ViennaHRLE - Hierarchical Run-Length Encoded Data Structure. [Online]. Available: https://github.com/ViennaTools/ViennaHRLE

[164] X. Klemenschits, O. Ertl, P. Manstetten, J. Weinbub, and L. Filipovic. ViennaLS - A High Performance Sparse Level Set Library. [Online]. Available: https://github.com/ViennaTools/ViennaLS

[165] T. Reiter, A. Scharinger, and X. Klemenschits. ViennaRay - Top Down MC Ray Tracing library. [Online]. Available: https://github.com/ViennaTools/ViennaRay

[166] X. Klemenschits, P. Manstetten, J. Weinbub, and L. Filipovic. ViennaPS - Vienna Process Simulation Library. [Online]. Available: https://github.com/ViennaTools/ViennaPS

[167] B. Houston, M. B. Nielson, C. Batty, O. Nilsson, and K. Museth, “Hierarchical RLE level set: A compact and versatile deformable surface representation,” *ACM Transactions on Graphics*, vol. 25, no. 1, pp. 151–175, 2006, doi: http://doi.acm.org/10.1145/1122501.1122508.

[168] S. P. Awate and R. T. Whitaker, “An interactive parallel multiprocessor level-set solver with dynamic load balancing,” 2004. [Online]. Available: https://www.cs.utah.edu/docs/techreports/2005/pdf/UUCS-05-002.pdf

[169] O. Ertl, “Numerical Methods for Topography Simulation,” Ph.D. dissertation, Institut für Mikroelektronik, 2010, doi: 10.34726/hss.2010.001.

[170] W. E. Lorensen and H. E. Cline, “Marching cubes: A high resolution 3D surface construction algorithm,” in *Proc. Computer Graphics and Interactive Techniques (SIGGRAPH)*. ACM, 1987, pp. 163–169, doi: 10.1145/37401.37422.

[171] D. R. Chand and S. S. Kapur, “An algorithm for convex polytopes,” *Journal of the American Chemical Society*, vol. 17, no. 1, pp. 78 – 86, 1970, doi: 10.1145/321556.321564.

[172] J. Shen, D. Zhang, Y. Wang, and Y. Gan, “AFM and SEM study on crystallographic and topographical evolution of wet-etched patterned sapphire substrates (PSS),” *ECS Journal of Solid State Science and Technology*, vol. 6, no. 1, pp. R24–R34, 2017, doi: 10.1149/2.0221701jss.

[173] J. Shen, D. Zhang, Y. Wang, and Y. Gan, “AFM and SEM study on crystallographic and topographical evolutions of wet-etched patterned sapphire substrate (PSS): Part ii. cone-shaped PSS etched in H_2SO_4 and H_3PO_4 mixture with varying volume ratio at 230°c,” *ECS Journal of Solid State Science and Technology*, vol. 6, no. 9, pp. R122–R130, 2017, doi: 10.1149/2.0091709jss.

[174] Y. K. Ooi and J. Zhang, “Light extraction efficiency analysis of flip-chip ultraviolet light-emitting diodes with patterned sapphire substrate,” *IEEE Photonics Journal*, vol. 10, no. 4, pp. 1–13, 2018, doi: 10.1109/JPHOT.2018.2847226.

[175] L. Zhou and A. Pang, “Metrics and visualization tools for surface mesh comparison,” in *Proc. Visual Data Exploration and Analysis*. SPIE, 2001, pp. 99–110, doi: 10.1117/12.424920.

[176] M. Kampl and H. Kosina, “The backward Monte Carlo method for semiconductor device simulation,” *Journal of Computational Electronics*, vol. 17, no. 4, pp. 1492–1504, 2018, doi: 10.1007/s10825-018-1225-6.

[177] L. Gnam, P. Manstetten, A. Hössinger, S. Selberherr, and J. Weinbub, “Accelerating flux calculations using sparse sampling,” *Micromachines*, vol. 9, no. 11, pp. 550–1 – 550–18, 2018, doi: 10.3390/mi9110550.

[178] A. Kensler and P. Shirley, “Optimizing ray-triangle intersection via automated search,” in *Proc. Interactive Ray Tracing (RT)*. IEEE, 2006, pp. 33–38, doi: 10.1109/RT.2006.280212.

[179] X. Lu, Z. Deng, and W. Chen, “A robust scheme for feature-preserving mesh denoising,” *IEEE Transactions on Visualization and Computer Graphics*, vol. 22, no. 3, pp. 1181–1194, 2016, doi: 10.1109/TVCG.2015.2500222.

[180] A. Khadidos, V. Sanchez, and C.-T. Li, “Weighted level set evolution based on local edge features for medical image segmentation,” *IEEE Transactions on Image Processing*, vol. 26, no. 4, pp. 1979–1991, 2017, doi: 10.1109/TIP.2017.2666042.

[181] M. E. Coltrin, R. J. Kee, and J. A. Miller, “A mathematical model of silicon chemical vapor deposition: Further refinements and the effects of thermal diffusion,” *Journal of The Electrochemical Society*, vol. 133, no. 6, pp. 1206–1213, 1986, doi: 10.1149/1.2108820.

[182] H. Matsumura, “Formation of polysilicon films by catalytic chemical vapor deposition (cat-CVD) method,” *Japanese Journal of Applied Physics*, vol. 30, no. Part 2, No. 8B, pp. L1522–L1524, 1991, doi: 10.1143/JJAP.30.L1522.

[183] M. E. Coltrin, P. Ho, H. K. Moffat, and R. J. Buss, “Chemical kinetics in chemical vapor deposition: Growth of silicon dioxide from tetraethoxysilane (TEOS),” *Thin Solid Films*, vol. 365, no. 2, pp. 251–263, 2000, doi: 10.1016/S0040-6090(99)01059-7.

[184] A. Bagatur'yants, K. Novoselov, A. Safonov, L. Savchenko, J. Cole, and A. Korkin, "Atomistic modeling of chemical vapor deposition: Silicon nitride CVD from dichlorosilane and ammonia," *Materials Science in Semiconductor Processing*, vol. 3, no. 1-2, pp. 23–29, 2000, doi: 10.1016/S1369-8001(00)00006-8.

[185] Y. Wang and R. Pollard, "An approach for modeling surface reaction kinetics in chemical vapor deposition processes," *Journal of The Electrochemical Society*, vol. 142, no. 5, pp. 1712–1725, 1995, doi: 10.1149/1.2048645.

[186] R. J. Buss, P. Ho, W. G. Breiland, and M. E. Coltrin, "Reactive sticking coefficients for silane and disilane on polycrystalline silicon," *Journal of Applied Physics*, vol. 63, no. 8, pp. 2808–2819, 1988, doi: 10.1063/1.340982.

[187] E. Bar and J. Lorenz, "3-D simulation of LPCVD using segment-based topography discretization," *IEEE Transactions on Semiconductor Manufacturing*, vol. 9, no. 1, pp. 67–73, 1996, doi: 10.1109/66.484284.

[188] M. M. IslamRaja, "Two precursor model for low-pressure chemical vapor deposition of silicon dioxide from tetraethylorthosilicate," *Journal of Vacuum Science & Technology B: Microelectronics and Nanometer Structures*, vol. 11, no. 3, pp. 720 – 726, 1993, doi: 10.1116/1.586778.

[189] K. F. Roenigk and K. F. Jensen, "Low pressure CVD of silicon nitride," *Journal of The Electrochemical Society*, vol. 134, no. 7, pp. 1777–1785, 1987, doi: 10.1149/1.2100756.

[190] R. Arora and R. Pollard, "A mathematical model for chemical vapor deposition processes influenced by surface reaction kinetics: Application to low-pressure deposition of tungsten," *Journal of The Electrochemical Society*, vol. 138, no. 5, pp. 1523–1537, 1991, doi: 10.1149/1.2085820.

[191] P. Manstetten, L. Filipovic, A. Hössinger, J. Weinbub, and S. Selberherr, "Framework to model neutral particle flux in convex high aspect ratio structures using one-dimensional radiosity," *Solid-State Electronics*, vol. 128, pp. 141–147, 2017, doi: 10.1016/j.sse.2016.10.029.

[192] M. F. Modest, *Radiative Heat Transfer.* Elsevier, 2013, doi: 10.1016/C2010-0-65874-3.

[193] L. Filipovic and X. Klemenschits, "Fast model for deposition in trenches using geometric advection," in *Proc. Simulation of Semiconductor Processes and Devices (SISPAD)*, in press.

[194] D. Morgan and K. Board, *An Introduction to Semiconductor Microtechnology.* Wiley, 1990.

[195] B. Voigtländer, M. Kästner, and P. Šmilauer, "Magic islands in Si/Si(111) homoepitaxy," *Physical Review Letters*, vol. 81, pp. 858–861, 1998, doi: 10.1103/PhysRevLett.81.858.

[196] U. W. Pohl, *Epitaxy of Semiconductors: Physics and Fabrication of Heterostructures.* Springer International Publishing, 2020, doi: 10.1007/978-3-030-43869-2.

[197] A. Krost, A. Dadgar, G. Strassburger, and R. Clos, "GaN-based epitaxy on silicon: Stress measurements," *physica status solidi (a)*, vol. 200, no. 1, pp. 26–35, 2003, doi: 10.1002/pssa.200303428.

[198] M. L. Hammond, “Silicon epitaxy by chemical vapor deposition,” in *Handbook of Thin Film Deposition Processes and Techniques.* Elsevier, 2001, pp. 45–110, doi: 10.1016/B978-081551442-8.50007-9.

[199] D. Dutartre, A. Talbot, and N. Loubet, “Facet propagation in si and sige epitaxy or etching,” *ECS Transactions*, vol. 3, no. 7, pp. 473–487, 2019, doi: 10.1149/1.2355845.

[200] T. J. Hubbard, “MEMS design: The geometry of silicon micromachining,” Ph.D. dissertation, California Institute of Technology, 1994, [Accessed: 2021, July]. [Online]. Available: https://resolver.caltech.edu/CaltechETD:etd-09162005-134646

[201] B. Radjenović, M. Radmilović-Radjenović, and M. Mitrić, “Level set approach to anisotropic wet etching of silicon,” *Sensors*, vol. 10, no. 5, pp. 4950–4967, 2010, doi: 10.3390/s100504950.

[202] S. Barraud *et al.*, “Vertically stacked-nanowires MOSFETs in a replacement metal gate process with inner spacer and SiGe source/drain,” in *Proc. International Electron Devices Meeting (IEDM).* IEEE, 2016, pp. 17.6.1–17.6.4, doi: 10.1109/IEDM.2016.7838441.

[203] W. K. Burton, N. Cabrera, and F. C. Frank, “The growth of crystals and the equilibrium structure of their surfaces,” *Philosophical Transactions of the Royal Society of London. Series A, Mathematical and Physical Sciences*, vol. 243, no. 866, pp. 299–358, 1951, doi: 10.1098/rsta.1951.0006.

[204] C. Ratsch *et al.*, “Level-set method for island dynamics in epitaxial growth,” *Physical Review B*, vol. 65, no. 19, pp. 195 403–1 – 195 403–13, 2002, doi: 10.1103/PhysRevB.65.195403.

[205] P. Kongetira, “Expression for the growth rate of selective epitaxial growth of silicon using dichlorosilane, hydrogen chloride, and hydrogen in a low pressure chemical vapor deposition pancake reactor,” *Journal of Vacuum Science & Technology B: Microelectronics and Nanometer Structures*, vol. 15, no. 6, pp. 1902 – 1907, 1997, doi: 10.1116/1.589576.

[206] D. House and D. Li, “Anisotropic etching,” in *Encyclopedia of Microfluidics and Nanofluidics.* Springer US, 2008, pp. 47–49, doi: 10.1007/978-0-387-48998-8_35.

[207] P. Pal, V. Swarnalatha, A. V. N. Rao, A. K. Pandey, H. Tanaka, and K. Sato, “High speed silicon wet anisotropic etching for applications in bulk micromachining: A review,” *Micro and Nano Systems Letters*, vol. 9, no. 1, pp. 4–1 – 4–59, 2021, doi: 10.1186/s40486-021-00129-0.

[208] O. R. Bengoetxea, “Development and characterization of plasma etching processes for the dimensional control and LWR issues during high-k metal gate stack patterning for 14FDSOI technologies,” Ph.D. dissertation, 2016, [Accessed: 2021, July]. [Online]. Available: http://www.theses.fr/2016GREAT009.pdf

[209] O. Ros, E. Pargon, M. Fouchier, P. Gouraud, and S. Barnola, “Gate patterning strategies to reduce the gate shifting phenomenon for 14 nm fully depleted silicon-on-insulator technology,” *Journal of Vacuum Science & Technology A: Vacuum, Surfaces, and Films*, vol. 35, no. 2, pp. 021 306–1 – 021 306–12, 2017, doi: 10.1116/1.4972228.

[210] V. M. Donnelly and A. Kornblit, “Plasma etching: Yesterday, today, and tomorrow,” *Journal of Vacuum Science & Technology A: Vacuum, Surfaces, and Films*, vol. 31, no. 5, pp. 050 825–1 – 050 825–48, 2013, doi: 10.1116/1.4819316.

[211] B. Wu, A. Kumar, and S. Pamarthy, “High aspect ratio silicon etch: A review,” *Journal of Applied Physics*, vol. 108, no. 5, pp. 051 101–1 – 051 101–20, 2010, doi: 10.1063/1.3474652.

[212] M. M. Frank, “High-k/metal gate innovations enabling continued CMOS scaling,” in *Proc. European Solid-State Device Research Conference (ESSDERC)*. IEEE, 2011, pp. 25–33, doi: 10.1109/ESSDERC.2011.6044239.

[213] H. Shin, W. Zhu, V. M. Donnelly, and D. J. Economou, “Surprising importance of photo-assisted etching of silicon in chlorine-containing plasmas,” *Journal of Vacuum Science & Technology A: Vacuum, Surfaces, and Films*, vol. 30, no. 2, pp. 021 306–1 – 021 306–10, 2012, doi: 10.1116/1.3681285.

[214] C. Petit-Etienne *et al.*, “Etching mechanisms of thin SiO_2 exposed to Cl_2 plasma,” *Journal of Vacuum Science & Technology B, Nanotechnology and Microelectronics: Materials, Processing, Measurement, and Phenomena*, vol. 29, no. 5, pp. 051 202–1 – 051 202–8, 2011, doi: 10.1116/1.3622311.

[215] L. Luo *et al.*, “An effective process to remove etch damage prior to selective epitaxial growth in 3D NAND flash memory,” *Semiconductor Science and Technology*, vol. 34, no. 9, pp. 095 004–1 – 095 004–5, 2019, doi: 10.1088/1361-6641/ab3130.

[216] H. Shin *et al.*, “Selective etching of TiN over TaN and vice versa in chlorine-containing plasmas,” *Journal of Vacuum Science & Technology A: Vacuum, Surfaces, and Films*, vol. 31, no. 3, pp. 031 305–1 – 031 305–6, 2013, doi: 10.1116/1.4801883.

[217] M. Hélot *et al.*, “Plasma etching of HfO_2 at elevated temperatures in chlorine-based chemistry,” *Journal of Vacuum Science & Technology A: Vacuum, Surfaces, and Films*, vol. 24, no. 1, pp. 30–40, 2006, doi: 10.1116/1.2134707.

[218] J. A. Levinson, E. S. G. Shaqfeh, M. Balooch, and A. V. Hamza, “Ion-assisted etching and profile development of silicon in molecular and atomic chlorine,” *Journal of Vacuum Science & Technology B: Microelectronics and Nanometer Structures*, vol. 18, no. 1, pp. 172–190, 2000, doi: 10.1116/1.591170.

[219] J. A. Levinson, E. S. G. Shaqfeh, M. Balooch, and A. V. Hamza, “Ion-assisted etching and profile development of silicon in molecular chlorine,” *Journal of Vacuum Science & Technology A: Vacuum, Surfaces, and Films*, vol. 15, no. 4, pp. 1902–1912, 1997, doi: 10.1116/1.580658.

[220] M. Balooch, M. Moalem, W. Wang, and A. V. Hamza, “Low-energy Ar ion-induced and chlorine ion etching of silicon,” *Journal of Vacuum Science & Technology A: Vacuum, Surfaces, and Films*, vol. 14, no. 1, pp. 229–233, 1996, doi: 10.1116/1.579924.

[221] N. M. Muthukrishnan, K. Amberiadis, and A. Elshabini-Riad, “Characterization of titanium etching in Cl_2/N_2 plasmas,” *Journal of The Electrochemical Society*, vol. 144, no. 5, pp. 1780–1784, 1997, doi: 10.1149/1.1837679.

[222] H. K. Chiu *et al.*, "Characterization of titanium nitride etch rate and selectivity to silicon dioxide in a Cl_2 helicon-wave plasma," *Journal of Vacuum Science & Technology A: Vacuum, Surfaces, and Films*, vol. 19, no. 2, pp. 455–459, 2001, doi: 10.1116/1.1342866.

[223] J. Tonotani, T. Iwamoto, F. Sato, K. Hattori, S. Ohmi, and H. Iwai, "Dry etching characteristics of TiN film using Ar/CHF_3, Ar/Cl_2, and Ar/BCl_3 gas chemistries in an inductively coupled plasma," *Journal of Vacuum Science & Technology B: Microelectronics and Nanometer Structures*, vol. 21, no. 5, pp. 2163 – 2168, 2003, doi: 10.1116/1.1612517.

[224] E. Sungauer *et al.*, "Etching mechanisms of HfO_2, SiO_2, and poly-Si substrates in BCl_3 plasmas," *Journal of Vacuum Science & Technology B: Microelectronics and Nanometer Structures*, vol. 25, no. 5, pp. 1640 – 1646, 2007, doi: 10.1116/1.2781550.

[225] W. S. Hwang, J. Chen, W. J. Yoo, and V. Bliznetsov, "Investigation of etching properties of metal nitride/high-k gate stacks using inductively coupled plasma," *Journal of Vacuum Science & Technology A: Vacuum, Surfaces, and Films*, vol. 23, no. 4, pp. 964–970, 2005, doi: 10.1116/1.1927536.

[226] R. A. H. Heinecke, "Control of relative etch rates of SiO_2 and Si in plasma etching," *Solid State Electronics*, vol. 18, no. 1, pp. 1146–1147, 1975, doi: 10.1016/0038-1101(77)90147-2.

[227] J. W. Coburn and E. Kay, "Some chemical aspects of the fluorocarbon plasma etching of silicon and its compounds," *IBM Journal of Research and Development*, vol. 23, no. 1, pp. 33–41, 1979, doi: 10.1147/rd.231.0033.

[228] L. M. Ephrath, "Selective etching of silicon dioxide using reactive ion etching with CF_4-H_2," *Journal of The Electrochemical Society*, vol. 126, no. 8, pp. 1419 – 1421, 1979, doi: 10.1149/1.2129291.

[229] J. L. Mauer, J. S. Logan, L. B. Zielinski, and G. C. Schwartz, "Mechanism of silicon etching by a CF_4 plasma," *Journal of Vacuum Science and Technology*, vol. 15, no. 5, pp. 1734–1738, 1978, doi: 10.1116/1.569836.

[230] Y.-Y. Tu, T. J. Chuang, and H. F. Winters, "Chemical sputtering of fluorinated silicon," *Physical Review B*, vol. 23, no. 2, pp. 823–835, 1981, doi: 10.1103/PhysRevB.23.823.

[231] E. Gogolides, P. Vauvert, G. Kokkoris, G. Turban, and A. G. Boudouvis, "Etching of SiO_2 and Si in fluorocarbon plasmas: A detailed surface model accounting for etching and deposition," *Journal of Applied Physics*, vol. 88, no. 10, pp. 5570–5584, 2000, doi: 10.1063/1.1311808.

[232] E. R. Parker, B. J. Thibeault, M. F. Aimi, M. P. Rao, and N. C. MacDonald, "Inductively coupled plasma etching of bulk titanium for MEMS applications," *Journal of The Electrochemical Society*, vol. 152, no. 10, pp. C675 – C683, 2005, doi: 10.1149/1.2006647.

[233] K. Blumenstock, "Anisotropic reactive ion etching of titanium," *Journal of Vacuum Science & Technology B: Microelectronics and Nanometer Structures*, vol. 7, no. 4, pp. 627 – 632, 1989, doi: 10.1116/1.584806.

[234] S. Norasetthekul *et al.*, “Etch characteristics of HfO_2 films on si substrates,” *Applied Surface Science*, vol. 187, pp. 75–81, 2002, doi: 10.1016/S0169-4332(01)00792-9.

[235] K. S. Min *et al.*, “Selective etching of HfO_2 by using inductively-coupled Ar/C_4F_8 plasmas and the removal of etch residue on si by using an O_2 plasma treatment,” *Journal of the Korean Physical Society*, vol. 53, no. 3, pp. 1675–1679, 2008, doi: 10.3938/jkps.53.1675.

[236] K. Takahashi and K. Ono, “Selective etching of high-k HfO_2 films over Si in hydrogen-added fluorocarbon ($CF_4/Ar/H_2$ and $C_4F_8/Ar/H_2$) plasmas,” *Journal of Vacuum Science & Technology A: Vacuum, Surfaces, and Films*, vol. 24, no. 3, pp. 437–443, 2006, doi: 10.1116/1.2187997.

[237] J. Chen, W. J. Yoo, Z. Y. Tan, Y. Wang, and D. S. Chan, “Investigation of etching properties of HfO based high-k dielectrics using inductively coupled plasma,” *Journal of Vacuum Science & Technology A: Vacuum, Surfaces, and Films*, vol. 22, no. 4, pp. 1552–1558, 2004, doi: 10.1116/1.1705590.

[238] B. E. Thompson and H. H. Sawin, “Polysilicon etching in SF_6 RF discharges: Characteristics and diagnostic measurements,” *Journal of The Electrochemical Society*, vol. 133, no. 9, pp. 1887–1895, 1986, doi: 10.1149/1.2109042.

[239] P. Panduranga, A. Abdou, Z. Ren, R. H. Pedersen, and M. P. Nezhad, “Isotropic silicon etch characteristics in a purely inductively coupled SF_6 plasma,” *Journal of Vacuum Science & Technology B*, vol. 37, no. 6, pp. 061 206–1 – 061 206–7, 2019, doi: 10.1116/1.5116021.

[240] C. P. D’Emic, “Deep trench plasma etching of single crystal silicon using SF_6/O_2 gas mixtures,” *Journal of Vacuum Science & Technology B: Microelectronics and Nanometer Structures*, vol. 10, no. 3, pp. 1105 – 1110, 1992, doi: 10.1116/1.586085.

[241] R. D’Agostino and D. L. Flamm, “Plasma etching of Si and SiO_2 in SF_6–O_2 mixtures,” *Journal of Applied Physics*, vol. 52, no. 1, pp. 162–167, 1981, doi: 10.1063/1.328468.

[242] H. M. Anderson, J. A. Merson, and R. W. Light, “A kinetic model for plasma etching silicon in a SF_6/o_2 RF discharge,” *IEEE Transactions on Plasma Science*, vol. 14, no. 2, pp. 156–164, 1986, doi: 10.1109/TPS.1986.4316518.

[243] R. J. Belen, S. Gomez, D. Cooperberg, M. Kiehlbauch, and E. S. Aydil, “Feature-scale model of Si etching in SF_6/O_2 plasma and comparison with experiments,” *Journal of Vacuum Science & Technology A: Vacuum, Surfaces, and Films*, vol. 23, no. 5, pp. 1430–1439, 2005, doi: 10.1116/1.2013317.

[244] R. J. Belen and S. Gomez, “Feature scale model of Si etching in SF_6/O_2/HBr plasma and comparison with experiments,” *Journal of Vacuum Science & Technology A: Vacuum, Surfaces, and Films*, vol. 24, no. 2, pp. 350–361, 2006, doi: 10.1116/1.2173268.

[245] D. Shamiryan, A. Redolfi, and W. Boullart, “Dry etching process for bulk FinFET manufacturing,” *Microelectronic Engineering*, vol. 86, no. 1, pp. 96–98, 2009, doi: 10.1016/j.mee.2008.10.001.

[246] O. Luere, E. Pargon, L. Vallier, B. Pelissier, and O. Joubert, "Etch mechanisms of silicon gate structures patterned in $SF_6/CH_2F_2/Ar$ inductively coupled plasmas," *Journal of Vacuum Science & Technology B, Nanotechnology and Microelectronics: Materials, Processing, Measurement, and Phenomena*, vol. 29, no. 1, pp. 011 028–1 – 011 028–10, 2011, doi: 10.1116/1.3522656.

[247] T. Ohchi *et al.*, "Reducing damage to si substrates during gate etching processes," *Japanese Journal of Applied Physics*, vol. 47, no. 7 PART 1, pp. 5324–5326, 2008, doi: 10.1143/JJAP.47.5324.

[248] M. Vinet *et al.*, "Bonded planar double-metal-gate NMOS transistors down to 10 nm," *IEEE Electron Device Letters*, vol. 26, no. 5, pp. 317–319, 2005, doi: 10.1109/LED.2005.846580.

[249] L. Desvoivres, L. Vallier, and O. Joubert, "X-ray photoelectron spectroscopy investigation of sidewall passivation films formed during gate etch processes," *Journal of Vacuum Science & Technology B: Microelectronics and Nanometer Structures*, vol. 19, no. 2, pp. 420 – 426, 2001, doi: 10.1116/1.1352727.

[250] M. Tuda, K. Shintani, and H. Ootera, "Profile evolution during polysilicon gate etching with low-pressure high-density $Cl_2/HBr/O_2$ plasma chemistries," *Journal of Vacuum Science & Technology A: Vacuum, Surfaces, and Films*, vol. 19, no. 3, pp. 711–717, 2001, doi: 10.1116/1.1365135.

[251] M. Lemme *et al.*, "Nanoscale TiN metal gate technology for CMOS integration," *Microelectronic Engineering*, vol. 83, no. 4-9, pp. 1551–1554, 2006, doi: 10.1016/j.mee.2006.01.161.

[252] F. Lärmer and A. Schlip, "Method of anisotropically etching silicon," U.S. Patent 5 501 893, 1996, [Accessed: 2021, July]. [Online]. Available: https://patents.google.com/patent/US5501893A/en

[253] U. Soysal, F. Marty, E. Géhin, C. Motzkus, and E. Algré, "Fabrication, electrical characterization and sub-ng mass resolution of sub-μm air-gap bulk mode MEMS mass sensors for the detection of airborne particles," *Microelectronic Engineering*, vol. 221, pp. 111 190–1 – 111 190–9, 2020, doi: 10.1016/j.mee.2019.111190.

[254] M. Liu *et al.*, "A novel low-g MEMS bistable inertial switch with self-locking and reverse-unlocking functions," *Journal of Microelectromechanical Systems*, vol. 29, no. 6, pp. 1493–1503, 2020, doi: 10.1109/JMEMS.2020.3032586.

[255] M. Kawano, X. Y. Wang, and Q. Ren, "Trench isolation technology for cost-effective wafer-level 3D integration with one-step TSV," in *Proc. Electronic Components and Technology Conference (ECTC)*, vol. 2020-June. IEEE, 2020, pp. 1161–1166, doi: 10.1109/ECTC32862.2020.00186.

[256] P. Kumar, I. Dutta, Z. Huang, and P. Conway, "Materials and processing of TSV," in *3D Microelectronic Packaging*, 2021, pp. 47–70, doi: 10.1007/978-981-15-7090-2_3.

[257] K. Bae and J. Park, "Efficient TSV fault detection scheme for high bandwidth memory using pattern analysis," in *Proc. International SoC Design Conference (ISOCC)*. IEEE, 2020, pp. 19–20, doi: 10.1109/ISOCC50952.2020.9333115.

[258] K. C. Chun *et al.*, “A 16-GB 640-GB/s HBM2E DRAM with a data-bus window extension technique and a synergetic on-die ECC scheme,” *IEEE Journal of Solid-State Circuits*, vol. 56, no. 1, pp. 199–211, 2021, doi: 10.1109/JSSC.2020.3027360.

[259] V. Pano, I. Tekin, I. Yilmaz, Y. Liu, K. R. Dandekar, and B. Taskin, “TSV antennas for multi-band wireless communication,” *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 10, no. 1, pp. 100–113, 2020, doi: 10.1109/JETCAS.2020.2974236.

[260] H. V. Jansen, M. J. de Boer, S. Unnikrishnan, M. C. Louwerse, and M. C. Elwenspoek, “Black silicon method: X. A review on high speed and selective plasma etching of silicon with profile control: An in-depth comparison between bosch and cryostat DRIE processes as a roadmap to next generation equipment,” *Journal of Micromechanics and Microengineering*, vol. 19, no. 3, pp. 033 001–1 – 033 001–41, 2009, doi: 10.1088/0960-1317/19/3/033001.

[261] D. Chin, S. H. Dhong, and G. J. Long, “Structural effects on a submicron trench process,” *Journal of The Electrochemical Society*, vol. 132, no. 7, pp. 1705–1707, 1985, doi: 10.1149/1.2114195.

[262] D. Keil and E. Anderson, “Characterization of reactive ion etch lag scaling,” *Journal of Vacuum Science & Technology B: Microelectronics and Nanometer Structures*, vol. 19, no. 6, pp. 2082 – 2088, 2001, doi: 10.1116/1.1414116.

[263] K. J. Owen, B. VanDerElzen, R. L. Peterson, and K. Najafi, “High aspect ratio deep silicon etching,” in *Proc. Micro Electro Mechanical Systems (MEMS)*. IEEE, 2012, pp. 251–254, doi: 10.1109/MEMSYS.2012.6170138.

[264] M. A. Blauw, G. Craciun, W. G. Sloof, P. J. French, and E. van der Drift, “Advanced time-multiplexed plasma etching of high aspect ratio silicon structures,” *Journal of Vacuum Science & Technology B: Microelectronics and Nanometer Structures*, vol. 20, no. 6, pp. 3106 – 3110, 2002, doi: 10.1116/1.1518018.

[265] B. Chang, P. Leussink, F. Jensen, J. Hübner, and H. Jansen, “DREM: Infinite etch selectivity and optimized scallop size distribution with conventional photoresists in an adapted multiplexed bosch DRIE process,” *Microelectronic Engineering*, vol. 191, pp. 77–83, 2018, doi: 10.1016/j.mee.2018.01.034.

[266] B. Chang, F. Jensen, J. Hübner, and H. Jansen, “DREM2: A facile fabrication strategy for freestanding three dimensional silicon micro- and nanostructures by a modified bosch etch process,” *Journal of Micromechanics and Microengineering*, vol. 28, no. 10, pp. 105 012–1 – 105 012–10, 2018, doi: 10.1088/1361-6439/aad0c4.

[267] V. Thi Hoang Nguyen *et al.*, “The CORE sequence: A nanoscale fluorocarbon-free silicon plasma etch process based on SF_6/O_2 cycles with excellent 3D profile control at room temperature,” *ECS Journal of Solid State Science and Technology*, vol. 9, no. 2, pp. 24 002–24 013, 2020, doi: 10.1149/2162-8777/ab61ed.

[268] B. Chang, “Technology development of 3D silicon plasma etching processes for novel devices and applications,” Ph.D. dissertation, Technical University of Denmark, 2018, [Accessed: 2021, July]. [Online]. Available: https://backend.orbit.dtu.dk/ws/portalfiles/portal/179026583/Thesis_BingdongChang_Final.pdf

[269] X. Klemenschits, S. Selberherr, and L. Filipovic, "Geometric advection and its application in the emulation of high aspect ratio structures," *Computer Methods in Applied Mechanics and Engineering*, vol. 386, pp. 114 196–1 – 114 196–22, 2021, doi: 10.1016/j.cma.2021.114196.

[270] M. A. Blauw, T. Zijlstra, R. A. Bakker, and E. van der Drift, "Kinetics and crystal orientation dependence in high aspect ratio silicon dry etching," *Journal of Vacuum Science & Technology B: Microelectronics and Nanometer Structures*, vol. 18, no. 6, pp. 3453 – 3461, 2000, doi: 10.1116/1.1313578.

[271] W. Jacobs, A. Kersch, P. Moll, W. Sabisch, and G. S. Icking-Konert, "A feature scale model for trench capacitor etch rate and profile," in *Proc. International Electron Devices Meeting (IEDM).* IEEE, 2002, pp. 891–894, doi: 10.1109/iedm.2002.1175980.

[272] N. Roxhed, P. Griss, and G. Stemme, "A method for tapered deep reactive ion etching using a modified Bosch process," *Journal of Micromechanics and Microengineering*, vol. 17, no. 5, pp. 1087–1092, 2007, doi: 10.1088/0960-1317/17/5/031.

[273] S. L. Lai, D. Johnson, and R. Westerman, "Aspect ratio dependent etching lag reduction in deep silicon etch processes," *Journal of Vacuum Science & Technology A: Vacuum, Surfaces, and Films*, vol. 24, no. 4, pp. 1283–1288, 2006, doi: 10.1116/1.2172944.

[274] T. F. Dupont and Y. Liu, "Back and forth error compensation and correction methods for removing errors induced by uneven gradients of the level set function," *Journal of Computational Physics*, vol. 190, no. 1, pp. 311–324, 2003, doi: 10.1016/S0021-9991(03)00276-6.

[275] S. Bertini, M. Verotti, A. Bagolini, P. Bellutti, G. Ruta, and N. Belfiore, "Scalloping and stress concentration in DRIE-manufactured comb-drives," *Actuators*, vol. 7, no. 3, pp. 57–1 – 57–22, 2018, doi: 10.3390/act7030057.

[276] B. Chang *et al.*, "Large area three-dimensional photonic crystal membranes: Single-run fabrication and applications with embedded planar defects," *Advanced Optical Materials*, vol. 7, no. 2, pp. 1 801 176–1 – 1 801 176–9, 2019, doi: 10.1002/adom.201801176.

[277] C. Auth *et al.*, "A 22 nm high performance and low-power CMOS technology featuring fully-depleted tri-gate transistors, self-aligned contacts and high density MIM capacitors," in *Proc. VLSI Technology (VLSIT).* IEEE, 2012, pp. 131–132, doi: 10.1109/VLSIT.2012.6242496.

[278] J. Robertson and R. M. Wallace, "High-k materials and metal gates for CMOS applications," *Materials Science and Engineering: R: Reports*, vol. 88, pp. 1–41, 2015, doi: 10.1016/j.mser.2014.11.001.

[279] K. Mistry *et al.*, "A 45 nm logic technology with high-k+metal gate transistors, strained silicon, 9 Cu interconnect layers, 193 nm dry patterning, and 100% Pb-free packaging," in *Proc. International Electron Devices Meeting (IEDM).* IEEE, 2007, pp. 247–250, doi: 10.1109/IEDM.2007.4418914.

[280] H.-J. Lee *et al.*, "Intel 22 nm FinFET (22FFL) process technology for RF and mm wave applications and circuit design optimization for FinFET technology," in *Proc. International Electron Devices Meeting (IEDM).* IEEE, 2018, pp. 14.1.1–14.1.4, doi: 10.1109/IEDM.2018.8614490.

[281] A. Razavieh, P. Zeitzoff, and E. J. Nowak, "Challenges and limitations of CMOS scaling for FinFET and beyond architectures," *IEEE Transactions on Nanotechnology*, vol. 18, pp. 999–1004, 2019, doi: 10.1109/TNANO.2019.2942456.

[282] B.-R. Huang, F.-H. Meng, Y.-C. King, and C. J. Lin, "Investigation of parasitic resistance and capacitance effects in nanoscaled FinFETs and their impact on static random-access memory cells," *Japanese Journal of Applied Physics*, vol. 56, no. 4S, pp. 04CD11–1 – 04CD11–6, 2017, doi: 10.7567/JJAP.56.04CD11.

[283] X. Klemenschits, S. Selberherr, and L. Filipovic, "Combined process simulation and emulation of an SRAM cell of the 5 nm technology node," in *Proc. Simulation of Semiconductor Processes and Devices (SISPAD)*. IEEE, in press, pp. 1 – 5.

[284] H. Mertens *et al.*, "Gate-all-around MOSFETs based on vertically stacked horizontal Si nanowires in a replacement metal gate process on bulk Si substrates," in *Proc. VLSI Technology (VLSIT)*, no. 1. IEEE, 2016, doi: 10.1109/VLSIT.2016.7573416.

[285] N. Loubet *et al.*, "Stacked nanosheet gate-all-around transistor to enable scaling beyond FinFET," in *Proc. VLSI Technology (VLSIT)*, vol. 5, no. 1. IEEE, 2017, pp. T230–T231, doi: 10.23919/VLSIT.2017.7998183.

[286] Samsung Newsroom. Reduced size, increased performance: Samsung's GAA transistor, MBCFET. [Accessed: 2021, July]. [Online]. Available: https://news.samsung.com/global/infographic-reduced-size-increased-performance-samsungs-gaa-transistor-mbcfettm

[287] G. Bae *et al.*, "3 nm GAA technology featuring multi-bridge-channel FET for low power and high performance applications," in *Proc. International Electron Devices Meeting (IEDM)*. IEEE, 2018, pp. 28.7.1–28.7.4, doi: 10.1109/IEDM.2018.8614629.

[288] H. Mertens *et al.*, "Vertically stacked gate-all-around Si nanowire CMOS transistors with dual work function metal gates," in *Proc. International Electron Devices Meeting (IEDM)*. IEEE, 2016, pp. 19.7.1–19.7.4, doi: 10.1109/IEDM.2016.7838456.

[289] A. Yanguas-Gil, *Growth and Transport in Nanostructured Materials: Reactive Transport in PVD, CVD, and ALD*. Springer, 2016, doi: 10.1007/978-3-319-24672-7.

List of Publications

Journal Articles

[1] X. Klemenschits, S. Selberherr, L. Filipovic, "Geometric advection and its application in the emulation of high aspect ratio structures", *Computer Methods in Applied Mechanics and Engineering*, vol. 386, 114196-1 – 114196-22, 2021, doi: 10.1016/j.cma.2021.114196.

[2] A. Toifl, M. Quell, X. Klemenschits, P. Manstetten, A. Hössinger, S. Selberherr, J. Weinbub, "The level-set method for multi-material wet etching and non-planar selective epitaxy", *IEEE Access*, 8, 115406 – 115422, 2020, doi: 10.1109/ACCESS.2020.3004136.

[3] X. Klemenschits, S. Selberherr, L. Filipovic, "Modeling of gate stack patterning for advanced technology nodes: A review" *Micromachines*, vol. 9, (invited), 631-10 – 631-31, 2018, doi: 10.3390/mi9120631.

Book Contributions

[4] T. Reiter, X. Klemenschits, L. Filipovic, "Impact of high-aspect-ratio etching damage on selective epitaxial silicon growth in 3D NAND flash memory", in *Book of Abstracts for EuroSOI Workshop and International Conference on Ultimate Integration on Silicon (EuroSOI-ULIS)*. 2021-09-01–2021-09-03, pp. 54—57, 2021 (In press).

[5] X. Klemenschits, S. Selberherr, L. Filipovic, "Modeling of gate stack patterning for advanced technology nodes: A review", in *MDPI Miniaturized Transistors*. (invited), 105 - 135, 2019, doi: 10.3390/books978-3-03921-011-4.

[6] X. Klemenschits, S. Selberherr, L. Filipovic, "Unified feature scale model for etching in SF_6 and Cl plasma chemistries", *Book of Abstracts for EuroSOI Workshop and International Conference on Ultimate Integration on Silicon (EuroSOI-ULIS)*. 177 – 180, 2018, doi: 10.1109/ULIS.2018.8354763.

Conference Contributions

[7] T. Reiter, X. Klemenschits, L. Filipovic, “Impact of high-aspect-ratio etching damage on selective epitaxial silicon growth in 3D NAND flash memory”, in *Proc. EuroSOI Workshop and International Conference on Ultimate Integration on Silicon (EuroSOI-ULIS)*. 2021.09.01–2021.09.03, pp. 34–35, 2021 (In press).

[8] L. Filipovic, X. Klemenschits, “Fast model for deposition in trenches using geometric advection”, in *Proc. Simulation of Semiconductor Processes and Devices (SISPAD)*. 2021.09.27–2021.09.29, pp. 1–4, 2021 (In press).

[9] X. Klemenschits, S. Selberherr, L. Filipovic, “Combined process simulation and emulation of an SRAM cell of the 5 nm technology node”, in *Proc. Simulation of Semiconductor Processes and Devices (SISPAD)*. 2021.09.27–2021.09.29, pp. 1–4, 2021 (In press).

[10] X. Klemenschits, S. Selberherr, L. Filipovic, ”Geometric advection algorithm for process emulation”, in *Proc. Simulation of Semiconductor Processes and Devices (SISPAD)*. 2020.09.23–2020.10.06, pp.59–62, 2020, doi: 10.23919/SISPAD49475.2020.9241678.

[11] X. Klemenschits, P. Manstetten, L. Filipovic, S. Selberherr, ”Process simulation in the browser: Porting ViennaTS using WebAssembly”, in *Proc. Simulation of Semiconductor Processes and Devices (SISPAD)*. 2019-09-04 - 2019-09-06, pp. 339–342, 2019, doi: 10.1109/SISPAD.2019.8870374.

[12] X. Klemenschits, S. Selberherr, L. Filipovic, ”Fast Volume Evaluation on Sparse Level Sets”, in *Proc. International Workshop on Computational Nanotechnology (IWCN)*. pp. 113–114, 2019, ISBN: 978-3-9504738-0-3.

[13] X. Klemenschits, S. Selberherr, L. Filipovic, ”Unified feature scale model for etching in SF_6 and Cl plasma chemistries”, in *Proc. EuroSOI Workshop and International Conference on Ultimate Integration on Silicon (EuroSOI-ULIS)*. 2018.03.19–2018.03.21, pp. 65–66, 2018, ISBN: 978-1-5386-4810-0.

Curriculum Vitae

08/2017 – present

Doctoral Candidate and Research Assistant
Institute for Microelectronics, TU Wien
Vienna, Austria

09/2013 – 07/2017

Master's degree, Physics with Nanoscale Physics
Thesis titled "Manipulation of the Dielectric Function of Silicon"
University of Birmingham
Birmingham, UK

09/2012 – present

Engineering Company Commander (Militia)
Austrian Armed Forces
Vienna, Austria

Advection Bliss

I was enjoying advection with bliss,
when it occurs, some feature I miss.
What I see, oh it makes me perplex,
it seems the Hamiltonian was not convex.
How could I possibly be so obtuse!
A more robust scheme I must certainly choose.
Luckily Friedrichs and Lax came before,
so blissful advection shall be evermore.