

Copyright © 1991, by the author(s).  
All rights reserved.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission.

**ALGORITHMS FOR THREE-DIMENSIONAL  
SIMULATION OF ETCHING AND DEPOSITION  
PROCESSES IN INTEGRATED CIRCUIT  
FABRICATION**

by

Edward William Scheckler

Memorandum No. UCB/ERL M91/99

12 November 1991

**ALGORITHMS FOR THREE-DIMENSIONAL  
SIMULATION OF ETCHING AND DEPOSITION  
PROCESSES IN INTEGRATED CIRCUIT  
FABRICATION**

Copyright © 1991

by

Edward William Scheckler

Memorandum No. UCB/ERL M91/99

12 November 1991

**ELECTRONICS RESEARCH LABORATORY**

College of Engineering  
University of California, Berkeley  
94720

TITLE PAGE

**ALGORITHMS FOR THREE-DIMENSIONAL  
SIMULATION OF ETCHING AND DEPOSITION  
PROCESSES IN INTEGRATED CIRCUIT  
FABRICATION**

Copyright © 1991

by

Edward William Scheckler

Memorandum No. UCB/ERL M91/99

12 November 1991

**ELECTRONICS RESEARCH LABORATORY**

College of Engineering  
University of California, Berkeley  
94720



# **ALGORITHMS FOR THREE-DIMENSIONAL SIMULATION OF ETCHING AND DEPOSITION PROCESSES IN INTEGRATED CIRCUIT FABRICATION**

Ph.D.

Edward William Scheckler

EECS Department

## **ABSTRACT**

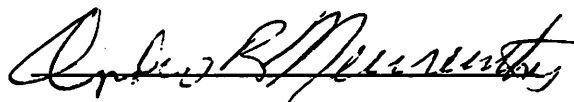
Algorithms for general surface advancement, three-dimensional visibility, and convolution over a surface have been developed and coupled with physical models for pattern transfer yielding efficient 3D topography simulation capability. The program, SAMPLE-3D, is compatible with engineering workstations and allows practical simulation of etch lag, deposition and etching with density variation, silylation, ion milling with a rotating source, and other 3D problems. The simulations require from 1 to 32 megabytes of memory and from 1 minute to a few hours of CPU time. A survey of other research provides direction for this and future work, as well as a comprehensive listing of the important literature.

An efficient cell-removal algorithm, requiring less than 10 megabytes and a few minutes for 100x100x100 cells, was implemented and found to be applicable to development. The cell algorithm is not suited to general simulation due to the difficulty in determining the surface orientation. A general surface advancement algorithm based on the motion of triangular facets was developed and tested for accuracy and efficiency. The method for calculating a

new surface from the advance facets uses the average intersection of facets, the solid angle swept out at the point, and the etch rate extrema directions. The algorithm maintains accuracy if no point advances further than 20% of the minimum mesh segment length.

The advancement is coupled with an array of rectangular prismatic cells to implement efficient algorithms for shadow detection, solid angle visibility, and loop identification. After sweeping out the cells based on the surface motion, the volume representation allows rapid determination of material properties at any point in the simulation region. Visibility determination then requires CPU time proportional to the surface area. Reflection and surface convolution require time proportional to the square of the surface area. Algorithm and data structure analysis gives rules for estimating CPU and memory requirements for specific simulations.

SAMPLE-3D is organized as a platform for continued model and algorithm development. Through the use of function calls, the modular structure allows the combination of process models and geometric operations to yield new simulation capability. A graphical user interface provides access to the simulator.

A handwritten signature in black ink, appearing to read 'Prof. A. R. Neureuther', written in a cursive style.

Prof. A. R. Neureuther

Committee Chairman

**Dedicated to Mom, Dad, Megan, Greg, and Mara.**

## Table of Contents

Dedication .....	ii
Table of Contents .....	iii
Acknowledgements .....	viii
CHAPTER 1 - INTRODUCTION .....	1
1.1 : Dissertation Overview .....	1
1.2 : Recent Trends in Integrated Circuit Electronics .....	1
1.3 : A Brief Introduction to Process Simulation .....	3
1.3.1 : The Disciplines of Process Modeling and Simulation .....	4
1.3.1 : The Role of Process Simulation in Integrated Circuit Development .....	5
1.4: The Need for 3D Topography Modeling .....	6
1.4 : Research Goals and Dissertation Outline .....	8
References for Chapter 1 .....	9
CHAPTER 2 - TOPOGRAPHY MODELING AND SIMULATION .....	12
2.1 : Etching and Deposition Processes .....	12
2.2 : Topography Process Modeling and Simulation .....	16
2.2.1 : A Simple Information Model for Topography Simulation .....	17
2.2.2 : A Slightly Modified Information Model for Topography Simulation .....	19
2.3 : A Survey of 3D Topography Simulation Tools .....	21
2.3.1 Photoresist Development Simulation with Volume-Removal Algorithms .....	21
2.3.2 Photoresist Development Simulation with Ray-trace Algorithms .....	24
2.3.3 Photoresist Development Simulation with a Network Method .....	25
2.3.4 General Topography Simulation with Cell-Based Algorithms .....	26
2.3.5 A Diffusion Equation Solution for Topography Simulation .....	26
2.3.6 Solid Modeling Applied to Topography Simulation .....	28
2.4 : A New Platform for 3D Topography Simulation, SAMPLE-3D .....	28
References for Chapter 2 .....	30
CHAPTER 3 - ANOTHER LOOK AT 3D CELL-REMOVAL ALGORITHMS .....	39
3.1 : Remaining Questions in 3D Cell-Removal Algorithms .....	39
3.2 : Cell-Removal With Spillover .....	39
3.3 : Alternate Methods For Volume-Removal Calculation .....	46
3.3.1 : Volume Removal Using Surface Orientation .....	46
3.3.2 : Volume Removal With Variable Material Density .....	49

3.4 : Conclusions on Cell-Based Simulation .....	53
References for Chapter 3 .....	54
<b>CHAPTER 4 - A GENERAL SURFACE ADVANCEMENT ALGORITHM FOR 3D TOPOGRAPHY SIMULATION .....</b>	<b>56</b>
4.1 : Algorithm Design Methodology .....	56
4.2 : General Surface Advancement and Time Discretization .....	56
4.3 : Surface Representation .....	57
4.4 : Surface Mesh Refinement .....	59
4.5 : Moving Nodes: The Facet Motion Algorithm .....	64
4.5.1 : The Simple Case: Moving Along Given Vectors .....	64
4.5.2 : Surface Normal Dependence: The 2D Segment Algorithm .....	64
4.5.3 : From 2D Segments to 3D Facets .....	65
4.5.4 : Facet Motion Algorithm for Isotropic Etching .....	68
4.5.5 : Facet Motion Algorithm for Simple Directional Process Models .....	73
4.5.6 : Facet Motion Algorithm for Strongly Direction Dependent Processes .....	80
4.5.7 : Border Points .....	81
4.5.8 : Material Boundaries .....	82
4.5.9 : Maximum Time Step in the Facet Motion Algorithm. ....	84
4.6 : Algorithm Implementation .....	84
4.6.1 : Data Structure .....	86
4.6.2 : Moving the Facets .....	88
4.6.3 : Fast Determination of Line-Triangle Intersections .....	88
4.6.4 : Finding the Fastest or Slowest Direction in a Solid Angle .....	91
4.6.5 : Rate Dependent Adaptive Time Step .....	93
4.6.6 : A Correction Factor for Inhomogeneous Rate Fields .....	93
4.6.7 : Putting It All Together .....	97
4.7 : Algorithm Evaluation .....	97
4.7.1 : Uniform Etch Rate .....	97
4.7.2 : Analytic Etch Rate: Gaussian .....	98
4.7.3 : Analytic Etch Rate: Triangular .....	101
4.7.4 : Inhomogeneous Rates in Lithography .....	103
4.7.5 : Isotropic Etching with a Mask .....	106
4.7.6 : Simple Directional Rate .....	108
4.7.7 : Highly Direction Dependent Rate .....	108
4.8 : Summary and Conclusion .....	111
References for Chapter 4 .....	114
<b>CHAPTER 5 - ALGORITHMS FOR 3D GEOMETRIC OPERATIONS .....</b>	<b>117</b>
5.1 : Geometric Operations in Process Simulation .....	117

5.2 : Visibility Problems in Computer Graphics .....	118
5.3 : The Surface-Cell Hybrid Data Representation .....	119
5.4 : Algorithm for Updating the Cells .....	120
5.5 : Cell Based Geometric Operations .....	127
5.5.1 : Efficient Shadow Test .....	128
5.5.2 : Efficient Solid Angle Visibility .....	128
5.5.3 : Surface Reflection .....	131
5.5.4 : Loop Identification .....	134
5.6 : Shadow and Solid Angle Performance Evaluation .....	137
5.7 : Commentary on the Surface-Cell Hybrid Method .....	139
References for Chapter 5 .....	141
<b>CHAPTER 6 - THREE-DIMENSIONAL ETCH MODELING AND SIMULATION</b>	
.....	142
6.1 : Geometric Etch Modeling For Process Simulation .....	142
6.2 : A Basic Model for Plasma Etching and Ion Milling .....	144
6.2.1 : Factors of Importance in Glow Discharge Etching .....	144
6.2.2 : Theoretical Basis for Etch Velocity Along the Surface Normal .....	145
6.2.3 : The Foundation of the 3D Unified Model .....	147
6.2.4 : Isotropic Etching in Plasma Processing .....	149
6.2.5 : Directly Incident Ions: Sputtering and Ion Milling .....	149
6.2.6 : Directly Incident Neutral Particles .....	153
6.2.7 : Indirectly Incident Etchant Flux .....	154
6.2.8 : Unifying the Model Terms and Constants .....	154
6.3 : Extensions to the Basic Model .....	158
6.3.1 : Sidewall Passivation .....	158
6.3.2 : Damage and Ion-Enhanced Etching .....	159
6.3.3 : Microelectric Field Effects .....	159
6.3.4 : A Surface Diffusion Model .....	161
6.3.5 : A 3D Extension of the Surface Diffusion Model .....	163
6.3.6 : An Approach to Solving the 2D Surface Diffusion Equation .....	164
6.4 : Plasma Etching and Ion Milling Simulations .....	167
6.4.1 : Coupling the Models and Algorithms .....	167
6.4.2 : Simulation Examples .....	169
6.5 : Crystal Etching .....	175
6.6 : Summary and Conclusion .....	177
References for Chapter 6 .....	179
<b>CHAPTER 7 - THREE-DIMENSIONAL DEPOSITION MODELING AND SIMULATION</b>	
.....	184
7.1 : Geometric Deposition Modeling for Process Simulation .....	184

7.2 : Basic Physical Vapor Deposition .....	185
7.3 : Extensions to the Basic Models .....	189
7.3.1 : Column Orientation .....	189
7.3.2 : Variable Film Density .....	193
7.3.3 : Surface Migration .....	196
7.3.4 : Reflection and CVD Models .....	202
7.4 : Summary and Conclusion .....	204
References for Chapter 7 .....	206
<b>CHAPTER 8 - SYSTEM ORGANIZATION, USER INTERFACE &amp; INTEGRATION .....</b>	<b>209</b>
8.1 : Topography Simulation Software Organization .....	209
8.2 : SAMPLE-3D Organization .....	210
8.2.1 : A High Level View .....	210
8.2.2 : Overall SAMPLE-3D Program Flow .....	212
8.2.3 : Operations in SAMPLE-3D .....	215
8.2.4 : Data Structure Organization .....	217
8.2.5 : Implementing Specific Models .....	218
8.2.6 : Estimating CPU Requirements .....	221
8.2.7 : Estimating Memory Requirements .....	222
8.3 : User Interface .....	223
8.3.1 : Command Interpreter .....	224
8.3.2 : Using the Command Interpreter .....	225
8.3.3 : Graphics and Visualization .....	227
8.3.4 : A Graphical User Interface .....	229
8.3.5 : Using the Graphical User Interface .....	230
8.4 : Technology CAD Integration .....	232
8.5 : Organizing SAMPLE-3D as a Functional Library .....	236
8.5 : Summary and Conclusion .....	240
References for Chapter 8 .....	241
<b>Chapter 9 - APPLIED SIMULATION .....</b>	<b>245</b>
9.1 : Introduction .....	245
9.2 : Deep-UV Lithography and Pattern Transfer .....	246
9.3 : Silylation .....	250
9.4 : Ion Milling with a Rotating Wafer .....	253
9.5 : Phase-Shift Lithography with Simple Deloop .....	255
9.6 : Deposition and Etching with Density Variations .....	258
9.7 : Lithography, Etching and Deposition .....	258
9.8 : Lithography Simulation with Rays and Cells .....	261
9.9 : Summary and Conclusion .....	265

References for Chapter 9 .....	267
Chapter 10 - CONCLUSION .....	269
10.1 : Looking Back and Looking Ahead .....	269
10.2 : Improving the Data Representation .....	274
10.3 : Improving the Algorithms .....	274
10.4 : Improving the Models .....	276
10.5 : Software Reorganization .....	277
10.6 : The Last Word .....	278
BIBLIOGRAPHY .....	279
APPENDIX - USER'S GUIDES .....	299
<b>CRATER</b> User's Guide .....	300
<b>NETCH</b> User's Guide .....	304
<b>SAMPLE3D-GUI v1.1</b> User's Guide .....	312
<b>PTOM</b> User's Guide .....	320
<b>CUTAWAY</b> User's Guide .....	321
<b>PDRAW</b> Updates .....	322
<b>CONTOUR</b> Updates .....	326



## ACKNOWLEDGEMENTS

A space for thanking. . .

First and foremost I must express my thanks to Professor Andrew R. Neureuther, my dissertation chair and research adviser. His enthusiasm for his work, his concern for the welfare of his students, his positive outlook, and his sense of fair play have made Professor Neureuther a success both in his career and in life. It has been my extreme good fortune to have been involved with the research program he directs.

Likewise, Professor William G. Oldham has played an important role in this work, not just as a dissertation committee member and careful reader of this dissertation, but also as a critical reviewer of my work over the last four years, and as a major force behind the acquisition of research funds. Professor Oldham has brought his wealth of experience to my education in more ways than he may know.

Appreciation is also extended to Professor H. Morrison for reviewing this dissertation and to Professor Costas Spanos for serving as the chair of my Ph.D. Qualifying Exam Committee.

Next, I would like to thank all of the other students who have contributed to an enjoyable and productive research environment, namely Rich Ferguson, Kenny Toh, Nelson Tam, Alfred Wong, Rob Wang, Dave Newmark, John Helmsen, John Hutchinson, Bill Partlo, Diane Hoffstetter, Don Lyons, Dr. Anton Pfau, Dr. Chris Spence, Annie Lai, John Camagna, Anita Chiu, Richard Hsu and especially Alex Wong - who arrived like me in the Fall of 1987 and has shared many similar experiences.

There are also many professionals in the field who with often nothing more than a simple question have contributed in countless small and varied ways to this work. Among them are Simon Polak, Nick Cowern, Jeff Hendrickson, Bob Hartzell, Anne G  rodolle, Joachim Pelka, Paco Leon, Masato Fujinaga, Tatsumi Ishizuka, Yoshihiko Hirai, Noboru Nomura, Masao Fukuma, Hiroo Masuda, Jim McVittie, Prof. Robert Dutton, Prof. Mike Brett, Prof. Tom Smy, and others too numerous to list here.

I would also like to thank those members of the Cal Jazz Choir who have provided a very pleasant and necessary diversion from my academic activities. Among them are Leroy, Al, Adam, Neal, Mark, Melissa, Susannah, Jen, Melinda, Jen  e, Leslie, Mary, Teri, Rachel, Kalene, Brian, Carol, Kevane, Mara, Michelle, Tom, Paul, Annie, Donnyelle, Angelique, Dan, Meaghan, Noemi, Tim, Beth Anne, and certainly directors Bill Ganz, Clinton Day, and Tony Pasqua. It has been my privilege to make music with you all.

Finally, the financial support of the Sematech Center of Excellence in Lithography and the Semiconductor Research Corporation is gratefully acknowledged.

## CHAPTER 1

### INTRODUCTION

*There is nothing more difficult to take in hand,  
more perilous to conduct or more uncertain in its  
success than to take the lead in the introduction  
of a new order of things.*

-Niccolo Machiavelli, *The Prince*

#### 1.1. DISSERTATION OVERVIEW

The purpose of this work is to advance the state of the art in practical three-dimensional simulation of fabrication processes used to make advanced integrated circuit computer chips. Specifically, this work will focus on simulation of processes for adding and removing materials on integrated circuit wafers. Process simulation has an important role to play in integrated circuit development because it promotes a better understanding of the relevant fundamental chemistry and physics and allows question to be investigated cheaply with a computer. To date, most of the research activity in process simulation has focused on two-dimensional cross-sections of integrated circuit elements. With the dramatic decrease in the size of devices today, full three-dimensional simulation is increasingly required. There are many unanswered questions in three-dimensional simulation waiting to be addressed. This chapter introduces recent trends in integrated circuit electronics and process simulation, and provides an overview of the work to be covered in this dissertation.

#### 1.2. RECENT TRENDS IN INTEGRATED CIRCUIT ELECTRONICS

Advances in computing and information processing power, over just the last decade, have heralded a new era that is being called the *Information Age*.<sup>1</sup> The changes wrought by this technological revolution are unparalleled by anything since the Industrial Revolution. A

primary catalyst for this new age is the remarkable pace with which integrated circuit (IC) electronics has increased in performance and decreased in cost. In 1963, a single solid-state transistor sold for \$10. Silicon memory chips that were widely available commercially in late 1980 contained about 20,000 transistors, had minimum access times of about 200 nanoseconds and sold for about \$8 per chip.<sup>†</sup> That translates as 0.05 cents per unit of memory storage (bit).<sup>2</sup> Now, in 1991, it is possible to purchase a single 4 megabit memory chip containing over 5 million transistors, with a minimum access time of 60 nanoseconds and at a cost of about \$30, or 0.0007 cents per bit. Even denser 16 megabit chips are currently available in small quantities.<sup>3</sup> Adjusted for inflation, the cost per bit has decreased more than a hundred-fold in just ten years. Going back to 1963, the unit cost has decreased over a million times!

The improvement in integrated circuit cost and performance is a result of both better circuit design and superior integrated circuit processing and manufacturing technology. One of the driving technologies in IC process advancement, which serves as a convenient indicator of the state of the technology, is the minimum manufacturable feature size. In 1980, the minimum pattern that could be produced on a production-worthy integrated circuit was about 2 millionths of a meter (microns). In 1991, chips are being sold with minimum features of less than 0.8 microns.<sup>4</sup> Equally remarkable is the fact that this trend is likely to continue. Experimental memory chips have been demonstrated with minimum feature sizes of less than 0.3 microns,<sup>5</sup> and continuous metal lines of less than 0.01 microns have been patterned using contamination resist lithography and ion beam etching.<sup>6</sup>

---

<sup>†</sup> The 1963 transistor price is from a Fairchild catalog. Typical prices for 16K and 4M DRAM chips reflect a survey of advertisers listed in the December 1980 and June 1991 issues of *Byte* magazine. The inflation rate for the last decade reflects the U.S. Consumer Price Index.

However, as feature dimensions have decreased, the corresponding complexity and cost of processing equipment has increased dramatically. For example, the original contact printing systems for patterning 10 micron features on a silicon wafer consisted of a simple light source, a mask made with colored plastic, and a basic mechanical focusing system.<sup>7</sup> Now, in order to produce 256 megabit memory chips, industry experts predict that sophisticated excimer laser light sources, masks that transmit different light phases in different regions, and complex projection, focusing and alignment systems will be required.<sup>8, 9</sup> As complexity has increased, so has cost. In the 1970's, a contact printer could be purchased for \$15,000. Commercial optical lithography systems now sell for more than \$3,000,000.<sup>10</sup> Given the complexity of manufacturing equipment, the cost to build a state-of-the-art fabrication facility is now approaching one billion dollars.

For integrated circuit technology to advance, and for companies to get a satisfactory return on their billion-dollar investments in manufacturing facilities, IC processing technology must be well understood. There are also important practical concerns such as the lead time to characterize and install new equipment, and the precision with which processes must be controlled to produce a high yield of quality products. Since the cost-benefit ratio of computers has come down dramatically with advances in integrated circuit technology, process simulation can be widely applied to help address these concerns.

### **1.3. A BRIEF INTRODUCTION TO PROCESS SIMULATION**

There are many complex physical and chemical interactions that must be understood in order to design and implement the next generation processes for integrated circuit fabrication. Fortunately, the computing power, that has been made possible by advances in IC technology, can also be applied to the problems of understanding and improving that technology.

Generically, this field is known as *technology computer-aided design* or *TCAD*. The field is further divided into several individual disciplines.

### 1.3.1. The Disciplines of Process Modeling and Simulation

Typically, it is impractical to derive the fundamental atomic processes involved in IC processing from first principles. Furthermore, process technology is advancing so rapidly that fundamental analysis is often obsolete by the time it is completed. Instead, technologists develop models to describe general process behavior. This field, known as *process modeling*, includes fundamental physical analysis, empirical study of processes, and macroscopic process behavior.

The field of applying computer power to the physics and chemistry of IC processing technology is known broadly as *process simulation*. Here, process models, which typically take the form of mathematical expressions, are solved with computers. The primary result of research efforts in process simulation is software which takes process conditions as input and generates information about the physical structure of integrated circuit devices.

Process modeling and simulation can be further broken down according to the type of processes which are modeled: 1) thermal and impurity processes, 2) pattern definition, and 3) pattern transfer. Thermal and impurity processing generally refers to techniques for introducing and redistributing impurity atoms in the silicon crystal lattice. These processes often occur at high temperature and include ion implantation, oxidation, diffusion, annealing, and epitaxy. Pattern definition processes delineate fine features in a coating on the wafer surface. These processes include optical, electron-beam, ion-beam and X-ray lithography. Pattern transfer processes include techniques for adding and removing material from the wafer surface, often

in the presence of a masking material defined by lithography. Pattern transfer processes include wet and dry etching, chemical vapor deposition, evaporation, and sputtering. Process simulation concerning the change in the shape of the wafer surface is called *topography simulation*. This term includes the simulation of pattern definition and pattern transfer and is sometimes extended to include epitaxy and oxidation processes.

### **1.3.2. The Role of Process Simulation In Integrated Circuit Development**

Process simulation has an important role to play in the development of modern integrated circuits. First, simulation allows process models to be tested. Different mathematical descriptions of IC processes may be readily analyzed and compared with experimental data. In this way, process engineers can discover which process conditions are of most importance to IC development. Second, process simulation provides a convenient vehicle for the transfer of information from research laboratory to design engineer. As processes are better understood, this knowledge can be incorporated into simulation software. This knowledge is then widely available to anyone with access to that software. Above all, process simulation allows the physics and chemistry of IC processing to be investigated and analyzed at low cost. Computer time is almost always cheaper than laboratory time, especially for highly speculative work. The main cost for process simulation is in the development of simulation software. But if process simulation results in better understanding of IC fabrication and in better technology transfer from research laboratory to design center, then the resulting product is likely to produce a higher return on initial investment. This will justify the expenditure to develop process simulation software.

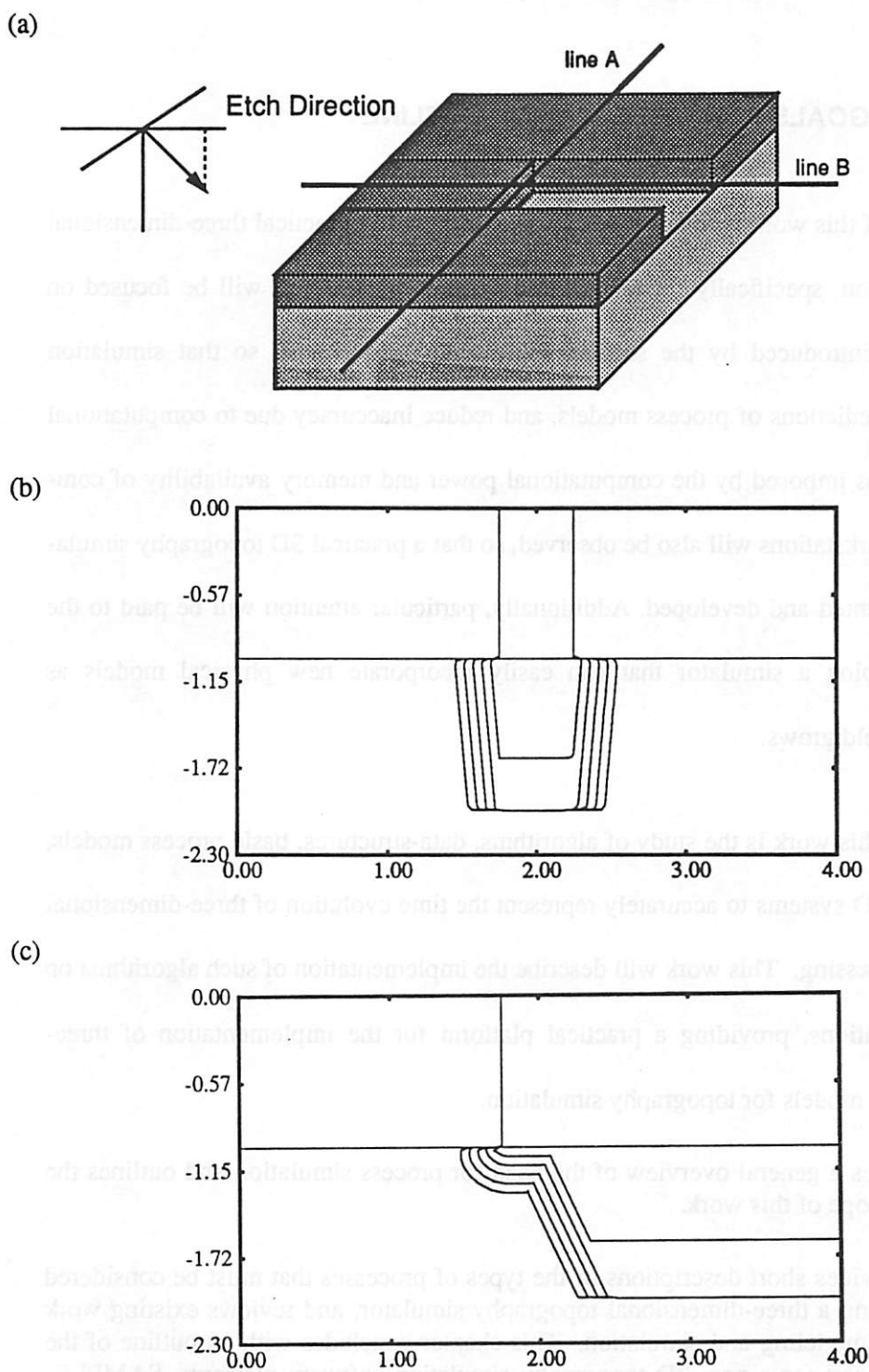
Indeed, many companies have already made successful use of simulation in technology development.<sup>11, 12</sup> These companies agree that effective development and use of process simulation capability has assisted in the design and production of competitive products.

#### **1.4. THE NEED FOR 3D TOPOGRAPHY MODELING**

In many respects, topography processes are the limiting steps in IC miniaturization. As already mentioned, the minimum feature size is a good indicator of the progress of IC technology. The minimum manufacturable feature limits the scaling of devices and circuits. Furthermore, devices that are packed densely together must also be interconnected. The reliability, performance, and cost of the interconnection is highly dependent on the technologies for patterning, etching, and deposition. The better these processes are understood, the better the process design will be.

Historically, it has often been sufficient to consider only the two-dimensional cross-section of an integrated circuit feature. The applicability of 2D simulation was based on the assumption that there was only a small change in the feature in the direction perpendicular to the simulation profile. This was often the case for features on the order of several microns. However, for smaller devices this is no longer true. As an illustration of this point, it is possible to simulate two different cross-sections of an essentially three-dimensional feature. Fig. 1.1 shows two cross section views of an etch simulation performed with the SAMPLE<sup>13</sup> program. Fig. 1.1a shows a view of the mask used to etch out the end of a trench and the orientation of a directional etch source. Fig. 1.1b is the result along line A and Fig 1.1c is the result along line B. The point where the two lines cross is in the middle of each simulation result. The distance etched is different at the center of the two plots, since neither simulation took the full 3D topography into account. This simple example only begins to reflect the





**Figure 1.1:** Two different 2D simulations of the end of a trench with SAMPLE. (a) 3D view of mask and etch directions. (b) Plasma etch simulation along line A. (c) Plasma etch simulation along line B.

complexities involved in modern integrated circuit processing.

## 1.5. RESEARCH GOALS AND DISSERTATION OUTLINE

The purpose of this work is to advance the state of the art in practical three-dimensional topography simulation, specifically for pattern transfer processes. Effort will be focused on reducing the error introduced by the surface advancement algorithms, so that simulation results reflect the predictions of process models, and reduce inaccuracy due to computational artifacts. Limitations imposed by the computational power and memory availability of common engineering workstations will also be observed, so that a practical 3D topography simulator may be implemented and developed. Additionally, particular attention will be paid to the problem of developing a simulator that can easily incorporate new physical models as knowledge in this field grows.

The scope of this work is the study of algorithms, data-structures, basic process models, and integrated TCAD systems to accurately represent the time evolution of three-dimensional surfaces during processing. This work will describe the implementation of such algorithms on engineering workstations, providing a practical platform for the implementation of three-dimensional process models for topography simulation.

*Chapter 1* gives a general overview of the need for process simulation and outlines the purpose and scope of this work.

*Chapter 2* provides short descriptions of the types of processes that must be considered when developing a three-dimensional topography simulator, and reviews existing work in topography modeling and simulation. This chapter concludes with an outline of the design philosophy for a new 3D topography simulation software platform, SAMPLE-3D.

*Chapter 3* provides further investigation of volume-removal methods for general topography simulation. Several key problems are identified and illustrated with examples. A fast and memory efficient cell removal algorithm is developed for photoresist

development simulation. Critical discussion of a diffusion equation approach to 3D topography simulation, and a "network" method is also offered.

*Chapter 4* describes algorithms and data structures for general surface advancement based on the motion of triangular surface facets. A theoretical basis for the correctness of the algorithms is provided for isotropic, directional, and surface orientation dependent processes. Inhomogeneous materials, masks, and material borders are also considered.

*Chapter 5* presents a hybrid surface-cell topography representation for efficient evaluation of several important geometric operations. The method for updating cells as the surface advances is given. Algorithms for the efficient calculation of shadows, flux integrals over solid angles, and other intra-geometry effects are given.

*Chapter 6* describes a variety of etch models which are implemented with the algorithms presented in previous chapters. A full 3D unified model for plasma etching and ion milling processes is developed. Techniques for including additional effects such as sidewall passivation, damage-induced etching, simple surface charging, and a surface diffusion model are also presented. Finally, anisotropic etching of crystals is investigated.

*Chapter 7* describes models for thin film deposition which are implemented with the algorithms presented in previous chapters. These models include isotropic deposition, evaporation, and sputtering. Surface migration, variable column orientation and orientation dependent density variation effects are also discussed.

*Chapter 8* describes the organization of an initial tool for 3D topography simulation, SAMPLE-3D, including an outline of how to implement new models and how to estimate performance. A command interpreter and graphical user interface is also presented. Additionally, future directions for 3D process simulation tool integration and continued development of SAMPLE-3D is discussed.

*Chapter 9* presents a variety of applications of the 3D models and algorithms given in previous chapters. Here, the emphasis is on capabilities unique to this work: deep-UV lithography and pattern transfer with plasma etching, photoresist silylation, ion milling of a rotating wafer, phase-shift lithography using the basic deloop routine, deposition and etch back with density variations, and a multistep wet etch, dry etch, and deposition simulation. Ray-trace and cell-removal lithography simulation are also compared.

*Chapter 10* concludes this dissertation by summarizing what was learned and discussing directions for future research.

*Appendixes* include a bibliography and user's guides.

## REFERENCES

1. J.S. Mayo, "Technology Requirements of the Information Age," *Bell. Lab. Rec.*, vol. 60, p. 55, 1982.
2. J.G. Posa, "Memories," *Electronics*, pp. 132-145, October 23, 1980.
3. G.F. Watson, "The Main Event," *IEEE Spectrum*, p. 30, January 1991.
4. K. Itoh, "Trends in Megabit DRAM Circuit Design," *IEEE Journal of Solid-State Circuits*, vol. 25, no. 6, pp. 778-789, June 1990.
5. Y. Nakagome, H. Tanaka, K. Takeuchi, E. Kume, Y. Watanbe, T. Kaga, Y. Kawamoto, F. Murai, R. Izawa, D. Hisamoto, T. Kisu, T. Nishida, E. Takeda, and K. Itoh, "An Experimental 1.5-V 64-Mb DRAM," *IEEE Journal of Solid-State Circuits*, vol. 26, no. 4, pp. 465-472, April 1991.
6. R.B. Laibowitz, A.N. Broers, J.T. C. Yeh, and J.M. Viggiano, *Applied Physics Letters*, vol. 35, p. 891, 1979.
7. J.G. Skinner, "Some Relative Merits of Contact, Near-Contact and Projection Printing," *Proceedings Kodak Interface*, vol. 73, p. 53, 1973.
8. *Integrated Circuits International*, p. 5, Elsevier Science Publishers, Ltd., Oxford, England, December 1990.
9. "i-Line Lithography vs. KrF Excimer Laser Lithography," *1991 Symposium on VLSI Technology - Evening Rump Session*, Oiso, Japan, May 29, 1991.
10. W.R. Runyan and K.E. Bean, *Semiconductor Integrated Circuit Processing Technology*, p. 18, Addison-Wesley, Reading, Massachusetts, 1990.
11. P. Lloyd, H.K. Dirks, E.J. Prendergast, and K. Singhal, "Technology CAD for Competitive Products," *IEEE Transactions on Computer-Aided Design of Integrated Circuits*

*and Systems*, vol. CAD-9, no. 11, pp. 1209-1216, November 1990.

12. D.C. Cole, E.M. Buturla, S.S. Furkay, K. Varahramyan, J. Slinkman, J.A. Mandelman, D.P. Foty, O. Bula, A.W. Strong, J.W. Park, T.D. Linton Jr., J.B. Johnson, M.V. Fischetti, S.E. Laux, P.E. Cottrell, H.G. Lustig, F. Pileggi, and D. Katcoff, "The Use of Simulation in Semiconductor Technology Development," *Solid-State Electronics*, vol. 33, no. 6, pp. 591-623, 1990.
13. *SAMPLE 1.8 User's Guide*, Electronics Research Laboratory, University of California, Berkeley, 1991.

## CHAPTER 2

### TOPOGRAPHY MODELING AND SIMULATION

*If you want to get somewhere. . .  
you had better know where you are.  
-Anon.*

#### 2.1. ETCHING AND DEPOSITION PROCESSES

Before developing a 3D topography simulator it is important to review the important relevant literature. Of fundamental importance is a review of the processes which affect topography. Several methods exist for adding and removing material from silicon wafers for integrated circuit fabrication. In all cases the process begins with a wafer (usually a wafer of crystalline silicon) in some initial state. This wafer enters a chamber in which the temperature, atmospheric conditions, and materials present can be controlled over some period of time. The result is a change in the state of the wafer, including changes in the shape of the wafer surface, and differences in the amount and type of materials present on the wafer. There are many general texts which describe most of the processes currently used in integrated circuit manufacturing.<sup>1, 2, 3, 4, 5</sup> Several important topography processes are listed and briefly described here, in order to provide a basis for the development of a general 3D topography simulator.

*Wet Etching* includes all those processes in which material is removed as it comes into contact with a liquid. Typically material is dissolved in a liquid solvent without any change in the chemical nature of the dissolved material.<sup>6, 7</sup> Photoresist development is one such process, in which material dissolves in the developer solution at a rate determined by previous exposure of the photoresist.<sup>8</sup> Another typical wet-etch process involves the formation of an oxide at the developer material interface, which then dissolves, exposing new material to the

etchant. This type of oxidation-reduction reaction is commonly used in the wet etching of Si and Al.<sup>9</sup> The etch rate is independent of the etch direction, except in the case of materials with well-defined crystal lattices.<sup>10</sup> Wet etching technology is popular because it is cheap, reliable, offers excellent selectivity between most mask and substrate materials, and is highly controllable.

*Plasma Etching* is a dry etch process in which a glow discharge produces a chemically reactive species from an inert molecular gas. This species then interacts with the material surface forming a volatile by-product which is desorbed from the surface and then diffused into the bulk of the gas. The gas is typically pressurized to a few torr in order to maintain a radio-frequency (rf) glow discharge.<sup>11, 12</sup> Different materials will etch at different rates depending on the type of gas used to generate the plasma. The pressure, temperature, and relative concentration of gases determine the etch rates of materials. Plasma etching, without directed particle bombardment, is isotropic but offers very high selectivity due to its sensitivity to specific chemical bonds.

*Reactive Sputter and Reactive Ion Etching (RSE and RIE)* † is similar to plasma etching except for the presence of ionic species which bombard or otherwise interact with the surface to enhance etching in the bottom of features.<sup>13, 14</sup> Features may be patterned directionally, *i. e.* without the significant undercut that exists in isotropic etch processes. *Reactive Ion Beam Etching (RIBE)* is a related process in which energetic ions are created by a variety of methods including electron cyclotron resonance (ECR).<sup>15</sup>

---

† There is some confusion in terminology between *RIE* and *RSE*. In this work, *RSE* will refer to processes where some etching due to physical sputtering occurs. *RIE* will refer to processes where ions damage the surface, or otherwise enhance etching due to neutrals, but do not necessarily remove material by physical momentum transfer.

*Ion Beam Milling* refers to processes in which energetic ions transfer momentum to the material surface to remove surface molecules.<sup>2</sup> Highly directional etching is possible but at the expense of low selectivity and potential for redeposition of material and surface damage. Ion milling processes run at low pressures and high excitation energies in order to generate fast moving particles, with a long mean free path.

*Evaporation* is a technique for depositing thin films. A material source is heated in a low pressure environment causing evaporation.<sup>16</sup> This vapor then condenses on the substrate which is typically located far from the source. Very pure films can be deposited at high rates, but the topography can shadow the flow of material from the source to the surface, thus resulting in reduced step coverage. Flash evaporation is a variation of this technique in which a thin wire of alloy is continuously fed to the vacuum chamber and immediately evaporated.<sup>17</sup>

*Sputter Deposition* has been known since 1852<sup>18</sup> and is among the most widely used deposition techniques for IC fabrication. Energetic particles are generated in a plasma and accelerated in order to dislodge (sputter) material from the surface of a source. The dislodged material travels to the wafer and condenses. From the point of view of a location on the wafer surface, sputtered material arrives from many different directions, reducing shadowing and resulting in good step coverage.<sup>19</sup> Sputtering suffers from the slow deposition rates of some materials. Many different variations on this technique exist including radio-frequency sputtering,<sup>20, 21</sup> magnetron sputtering,<sup>22</sup> and bias sputtering.<sup>23</sup> The primary differences among these methods are the mechanisms by which particles are energized or the relative electric potentials on various parts of the system (wafer, source, etc.). It is also possible to elevate the substrate temperature to improve step coverage.<sup>24</sup>



*Chemical Vapor Deposition* is typically used for amorphous and polycrystalline thin film formation<sup>25, 26</sup> but can also be used for metals.<sup>27</sup> Non-volatile solid films are the result of reactions of vapor phase chemicals which contain the required constituent elements. The gases are introduced into a reaction chamber and decomposed at a heated surface to form a thin film. With some materials (notably tungsten) it is possible to deposit material selectively on certain materials and not on others.<sup>28</sup> Atmospheric pressure (APCVD), low pressure (LPCVD),<sup>29</sup> plasma enhanced (PECVD),<sup>30</sup> photon enhanced,<sup>31</sup> and bias ECR techniques<sup>32</sup> are variations on this process.

*Electroless Plating* is another deposition process for metallization.<sup>33</sup> Unlike vapor deposition, the medium for transporting metal to the wafer surface is a liquid solution. The metal in the solution interacts chemically with certain materials on the wafer, leaving solid metal behind. This process allows the selective deposition of metal layers on certain materials only. Also, there is no step coverage problem since the metal film grows from the bottom surface. This process is not widely used for IC fabrication at present, but is being considered as a candidate for additive pattern transfer for copper interconnects.

*Epitaxy* in IC fabrication refers to the overgrowth of thin layers of single-crystal material onto a single-crystal substrate. Single-crystal silicon can be deposited with a vapor-phase CVD process.<sup>34</sup> It is also possible to control some epitaxial processes so that silicon is grown selectively.<sup>35</sup>

Many variations on the above processes exist, but nearly all processes of importance to topography evolution in modern IC fabrication are related to at least one of the above processes. A 3D topography simulator that can be applied to the majority of the above processes is likely to be useful well into the next century.

## 2.2. TOPOGRAPHY PROCESS MODELING AND SIMULATION

There are about as many process models as there are processes, and many of these have been incorporated into two-dimensional topography simulators. Fortunately, there are many good reviews of topography modeling and simulation.<sup>3, 36, 37, 38, 39, 40</sup> Many programs for simulating two-dimensional cross-sections of topography processes have been reported, including SAMPLE,<sup>41, 42</sup> BICEPS,<sup>43</sup> COMPOSITE,<sup>44</sup> DEER,<sup>45</sup> DEPICT,<sup>46</sup> ESPRIT,<sup>47</sup> FEDDS,<sup>48</sup> PROLITH,<sup>49</sup> SIMBAD,<sup>50</sup> SPEEDIE,<sup>51</sup> TITAN,<sup>52</sup> and FABRICS.<sup>53</sup> Two-dimensional simulation is maturing rapidly and many of these programs are widely available and widely used. Table 2.1 lists these programs and their capabilities.

Table 2.1: 2D Topography Simulation Programs					
Program	Date	Processes	Models and Algorithms	Availability	Comments
SAMPLE SAMPLE 1.8	1979 1991	Lithography Wet & Dry Etch Ion-Milling Deposition	String Advancement Gemoetric Depo & Etch Optical, E-beam, X-ray Advanced Resist Kinetics	widely available from UC-Berkeley	
BICEPS BICEPS 5.0	1983 1990	All Si Processes		AT&T internal	
COMPOSITE	1985	All Si Processes	String Advancement New Dry Etch models	widely available from Fraunhofer Gesellschaft	includes SAMPLE 1.6a for lithography
DEER	1989	Lithography Etching Deposition incl. ECR	Segment Advancement by 'virtual planes.' Unified velocity expression	NT&T internal	6 parameters used to fit model to any topo. process
DEPICT-2	1990	Lithography Deposition Etching	Improved Segment Advancement	Commercial product from TMA	Uses SPLAT and segment algorithms
ESPRIT	1987	Etching	Improved String	Hitachi int.	
FEDDS FEDDS-2	1985	All Si Processes	String Advancement	IBM, now sold	Uses SAMPLE Algorithms
PROLITH PROLITH/2	1985 1990	Lithography	String Advancement	Commercially available from FINLE Tech.	
SIMBAD	1988	Deposition: Evaporation Sputtering Magnetron, rf	Monte Carlo ballistic deposition. Some wet-etching	Univ. of Alberta internal	Allows density modeling. Quasi 3D for contacts
SPEEDIE	1990	Plasma Etching LPCVD	String Advancement Rigorous plasma physics	Stanford beta release	
TITAN	1989	General	String, Analytic	CNET	
FABRICS FABRICS-II	1985 1988	Statistical simulator	Basic topography models	from CMU	designed for statistical analysis

### 2.2.1. A Simple Information Model for Topography Simulation

Despite the variety of implementations, it is possible to develop a simple information model which describes the operation of most 2D topography simulators and highlights several important points. The information flow diagram presented in Fig. 2.1 is derived from knowledge of existing two-dimensional simulation, but it will be readily applicable to the development of a 3D topography simulator.

Most topography simulators process information in a straightforward manner. Information about the wafer state and process conditions is provided as input to mathematical models of process physics. A variety of algorithms and numerical methods are used to solve the models, which in turn provide information about the change in the wafer state over time. In the real world, the wafer state evolves continuously until the process is complete. In the realm of simulation however, time is necessarily broken down into steps of finite duration. To achieve physical accuracy the models and algorithms must be correct over the given time step. For most topography process models, the maximum allowed time step is at least a few orders of magnitude smaller than the actual process time, thus several iterations around this loop are required to complete the simulation.

The first, and most fundamental element of this information model is a representation of the wafer state. For topography simulation, two pieces of information are of interest as depicted in Fig. 2.2: 1) material boundaries, and 2) inhomogeneous attributes distributed throughout a material. Many 2D programs, such as SAMPLE and COMPOSITE, use ordered lists of coordinate pairs, called *strings*, to represent material layers or polygonal material boundaries. When inhomogeneous information is required, it can be represented by numerical values attached to a mesh or grid which is superimposed on the material region. SIMBAD

takes a different approach and represents the structure as an aggregation of small discs, each with its own location. The surface boundary is not explicitly represented. This method allows the microstructure of thin films to be represented. The surface can be extracted by locating the zero material density contour. FABRICS follows yet a different approach and represents inhomogeneous information as fitting parameters to analytic functions, allowing for a very compact wafer state representation. Different topography simulators tailor the particular wafer state representation to the needs of the physical models, however it is always possible to extract the surface topography.

The second element of this information flow diagram is the connection of the process model to the change in the wafer state in the form of state evolution algorithms. For string-based representations of surface geometries, the state evolution algorithms take the form of mechanisms for advancing the string. Some of the earliest string advancement algorithms<sup>54</sup> are still widely used in programs such as SAMPLE, COMPOSITE, and FEDDS. Variations and improvements on this technique are used in other programs like SPEEDIE<sup>55</sup> and DEPICT.<sup>56</sup> In these algorithms, the string segments or points are advanced at a rate and direction determined by the location on the surface, the surface normal, and the process physics. As the surface evolves, string points must be added or deleted to maintain an accurate surface description.

The third part of the information flow is the modeling of global phenomena that occur far from the wafer surface but nevertheless influence and are influenced by the wafer state. In the case of lithography simulation, this includes modeling of the illumination system and exposure in the photoresist before development begins. In dry-etching simulation, this includes modeling of the plasma and ion transport to the wafer. For deposition simulation such phenomena include the flux distribution of incoming atoms. These models are solved in a

variety of ways. SPEEDIE and SIMBAD use Monte Carlo methods to calculate the transport of materials to the wafer. SAMPLE solves analytic models of light illumination to find the image at the wafer surface.

### **2.2.2. A Slightly Modified Information Model for Topography Simulation**

Typically, topography simulation begins with the modeling of global phenomena. However, it is not always necessary to recalculate this information every time the wafer state is updated. For example, once the flux distribution of incoming material for metal deposition is determined, that information may be considered constant as the metal film grows. The only new information that must be included is the changing shape of the film, which restricts the arrival angle, but does not change the overall flux distribution. Given this fact, the information model of Fig. 2.1 may be slightly modified as shown in Fig. 2.3. This diagram illustrates the idea that in many cases the physics important to topography simulation can be presented in the form of a constant input to the algorithms for combining the surface geometry with the process model.

This way of looking at topography simulation reveals the following important insight: A critical element of topography simulation is the wafer state evolution algorithm. Surface advancement algorithms must be investigated thoroughly and implemented carefully or there can be no way of knowing whether the physical process models used will ultimately result in a correct simulation. Given this point, a large part of the development of a 3D topography simulator should be devoted to the study and investigation of algorithms for connecting physical models to topography evolution. This task will be undertaken in subsequent chapters. Additionally, some of the models mentioned in this section will be presented in more detail and investigated for their applicability to full 3D simulation.

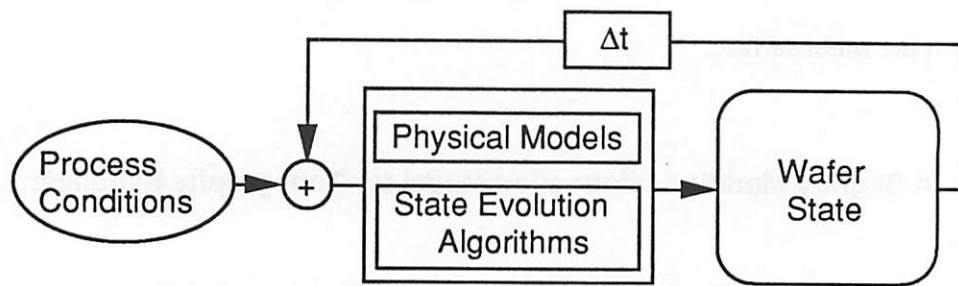


Fig. 2.1: Generic information flow model for topography simulation

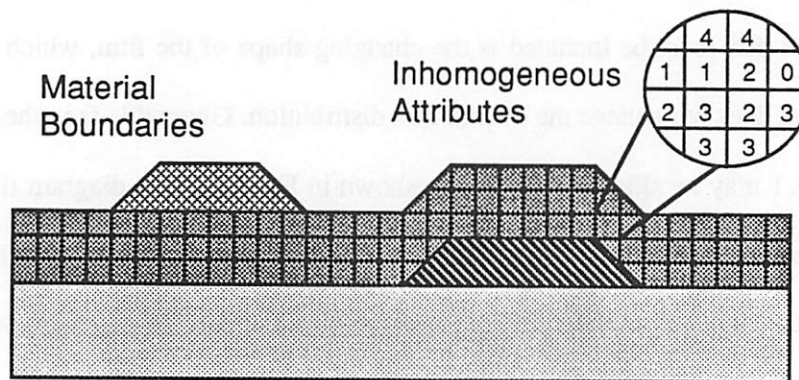


Fig. 2.2: Important wafer state information for topography simulation

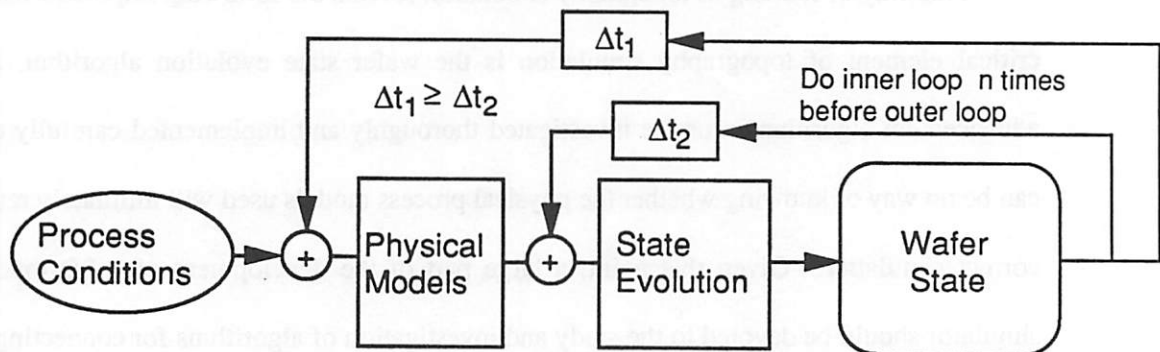


Fig. 2.3: A modified information flow model for topography simulation

### 2.3. A SURVEY OF 3D TOPOGRAPHY SIMULATION

Many three-dimensional topography simulators have been reported for lithography simulation. A few other simulators have been reported for dry-etching and deposition simulation. A variety of surface evolution algorithms have been demonstrated, including *volume-removal methods*, *ray-trace algorithms*, a *network method*, topography simulation via a solution of the diffusion equation, and solid modeling approaches to topography simulation. Table 2.2 summarizes the research surveyed in this section.

Table 2.2: 3D Topography Simulation Programs					
Program	Date	Processes	Models and Algorithms	Availability	Comments
LITHSIM	1980/91	Lithography	Cell Method	E. Germany	
RD3D	1980	Lithography	Cell Method	IBM internal	
PEACE	1987	Lithography	Cell Method	Matsushita internal	
SOLID	1990	Lithography	Cell with Spillover	Fraunhofer Sold by Silvaco	integer math for speed
XMAS	1990	X-Ray Lith.	Cell with Spillover	Fraunhofer	
CARPS	1990	Lithography	Cell Method	Toshiba int.	Monte Carlo Surface Diffusion
TRIPS-I	1985/87	Lithography	Ray-Trace	Hitachi internal	
Jia et. al.	1987	E-Beam Lith.	Ray-Trace		
Barouch et. al.	1989	Lithography	Ray-Trace		B-spline interpolation
SAMPLE-3D	1990	Lithography	Ray-trace	UC-Berkeley beta release	Recursive Euler
3D-EBLS	1991	E-Beam Lith.	Ray-Trace	Samsung internal	
Ishizuka	1990	Lithography	Network Method	Fuji RIC internal	
FINE	1990	Lith., Etch, Depo	Cell Method	Fujitsu internal	
ADEPT	1990	Dry Etching	Cell Method	Fraunhofer internal	Models surface kinetics
3D-MULSS	1988/91	Lith., Etch, Depo	Concentration Contour Diffusion Equation	Mitsubishi internal	
Oyster	1983	Basic Processes	Solid Model	IBM internal	geometric only
Footc	1990	Si Crystal Etch	Solid Model	UC-Berkeley	predicts facets

#### 2.3.1. Photoresist Development Simulation with Volume-Removal Algorithms.

The first three-dimensional topography simulators focused on the simplest of topography processes: wet etching for photoresist development. Three-dimensional simulation of photoresist development is almost as old as the field of process simulation itself, with work published in Germany as early as 1980,<sup>57</sup> which was closely followed by work at IBM on RD3D,

published in 1981.<sup>58</sup> Both of these simulators employ variations on volume-removal algorithms which are still being used today in the simulator PEACE,<sup>59, 60</sup> developed at Matsushita Electric Industries, and in SOLID,<sup>61</sup> developed at the Fraunhofer Gesellschaft in Berlin. XMAS<sup>62</sup> uses the same algorithm as SOLID to simulate development in X-Ray lithography. CARPS,<sup>63</sup> from Toshiba uses a Monte Carlo dissolution method to model the surface reactions.

These simulators established the usefulness of volume-removal or cell-based algorithms for practical 3D etching simulation. In this method, the volume of material is divided into a large array of regular prismatic elements or cells. The cells that come in contact with the developer are removed one by one, according to the local etch rate at the cell and the number of cell faces exposed.

The primary advantage of this algorithm is that the topography representation is absolutely stable and can support structures with tunnels, or regions of material which are completely disconnected from other regions. Furthermore, the basic implementation is relatively straightforward, resulting in a robust simulator with a relatively modest investment in software engineering.

RD3D and PEACE remove cells by searching for the cell with the minimum time for removal, deleting that cell and updating a global clock, evaluating the new time to removal for all the surface cells, and then searching again for the cell with the minimum time for removal. However, this double loop results in a slow algorithm. In PEACE, the CPU time increases proportionally with the 1.78 power of the number of cells removed. A similar trend is seen in RD3D, in which doubling the number of surface cells nearly quadruples the CPU time.



It is possible to improve the run time dramatically by eliminating the double loop in the algorithm. Instead, the time step duration is fixed. At each time step in the simulation, the time to removal (or volume remaining) of every surface cell is updated. Any cell with a time to removal (or volume remaining) of less than zero is removed from the surface. In this way, an entire surface layer may be removed in one time step, resulting in a run time nearly linear with the surface area at each time step, and thus proportional to the volume removed. This is the method used in SOLID. However, as pointed out by M. Fujinaga<sup>64</sup> and J. Pelka,<sup>65</sup> the fixed time step can result in a cell's clock being decremented to a value less than zero. It is necessary to redistribute this negative value among the adjacent cells.

Despite the practicality of cell-removal algorithms for 3D simulation, they tend to suffer from a lack of accuracy. This effect was demonstrated by Hirai *et. al.*<sup>60</sup> for the case of uniform spherical etching. As the simulation proceeded, facets began to appear. This is a direct result of the algorithm which alters the time to removal based on the number of faces exposed. Regardless of the true surface orientation a set of cells is intended to represent, the local etch rate will be enhanced along certain preferred directions: the cell diagonal, or one of the cell face diagonals. It is well known that any time preferred etch rates exist, facets will appear. K.K.H. Toh discovered a similar result in a modified cell-removal algorithm.<sup>66</sup>

Despite the problems with accuracy, volume-removal methods may indeed be sufficient for practical development simulation. The advantage of stability and robustness is significant; therefore, it is worth investigating this approach further. Additionally, highly accurate results have been demonstrated with SOLID. Chapter 3 provides more investigation of this topic.

### 2.3.2. Photoresist Development Simulation with Ray-trace Algorithms

Fermat's Principle states that a light ray traveling through a medium of spatially-varying index of refraction will follow a trajectory that minimizes the transit length.<sup>†</sup> A similar phenomenon holds for a point on a material surface advancing through a medium of spatially-varying etch rate. Since this physical principle of ray-tracing can be solved exactly, it can be used for accurate simulation of photoresist development.

Paul Hagouel was the first to derive the full mathematics of the ray-trace algorithm for 3D development simulation.<sup>68</sup> The first reported 3D implementation of a ray-trace algorithm, was TRIPS-I presented by Hitachi in 1985,<sup>69</sup> and subsequently improved in 1987.<sup>70</sup> In TRIPS-I, the ray-trace equation was solved numerically with the Runge-Kutta-Gill method. A similar approach was applied to electron-beam lithography development in 1987 by Jia *et al.*,<sup>71</sup> and in 1991 by K. Y. Lee *et al.*<sup>72</sup> A further refinement was introduced by Barouch *et al.*,<sup>73</sup> which represented the Principle of Least Time (which is physically equivalent to Fermat's Principle) as a system of seven ordinary differential equations which were solved using a specialized fourth-order Runge-Kutta method. After each time step, a tensor product B-spline interpolating function was used to obtain a smooth surface representation and ensure mesh integrity.

K. K. H. Toh implemented a highly accurate and efficient version of the ray-trace algorithm by using a recursive Euler method to solve the ray-trace equation at each surface point.<sup>66</sup> After a point is advanced, the direction vector associated with that point is updated according to the ray-trace equation. If the vector orientation changes by more than 5 degrees, the time step for that point is halved and the point advancement is recalculated for the two

---

<sup>†</sup> This is proven in Born & Wolf,<sup>67</sup> Section 3.3.2

smaller time steps. This process may proceed recursively. Since the maximum allowed change in vector orientation is held to less than a few degrees, the Euler calculation results in only a small numerical error.

A major difficulty with ray-trace algorithms is the control of the surface mesh. First, as the mesh expands and contracts, points must be added and deleted. It is a non-trivial software engineering problem to ensure that the surface mesh data structure maintains its integrity. Second, it is possible for rays to cross, resulting in erroneous surface loops. These loops describe a 'negative volume' which simply would not exist in the real world, but does remain as a simulation artifact. The loops do not affect the correctness of the rest of the surface, but do take up additional memory and CPU time. The B-spline interpolation implemented by Barouch *et. al.* reduces the problem of loops, but at the expense of the significant extra computation time needed for the interpolation. Nevertheless, despite these implementation difficulties, the ray-trace algorithm can be a fast, accurate, and efficient way to simulate photoresist development.

### **2.3.3. Photoresist Development Simulation with a Network Method**

Recently, T. Ishizuka has introduced an algorithm for development simulation called the network method.<sup>74</sup> The photoresist region is divided up into small tetrahedrons. The progress of the development is expressed by the movement of points along the edges of the tetrahedrons. The collection of edge points describing the surface is called the network. With the network, it is possible to determine the surface normal at each point. The distance moved along each edge is then related to the surface normal. As the surface passes through a volume element, those points are simply removed from the network and replaced by points on the edges of a new volume element. In this way, the problem of surface control is greatly

simplified. The problem of loop formation is eliminated. This algorithm has only recently been reported, but it seems to show great promise as an effective topography simulation method. It has only been applied to photoresist development, but could be generalized to anisotropic etching and development.

#### **2.3.4. General Topography Simulation with Cell-Based Algorithms**

A full 3D topography simulator FINE, developed by Fujitsu, uses cell-based algorithms.<sup>75</sup> It appears to use macroscopic etch rate models similar to those in SOLID and PEACE, but has been extended to deposition, dry-etching, and oxidation.

The same group that developed SOLID has also reported a dry-etching simulator, ADEPT.<sup>76</sup> An important feature of this algorithm is that it does not use macroscopic rates to remove cells, as in the previously mentioned development simulators. Instead, the cell sizes are close to the same order of magnitude as elementary crystal lattice cells, or even surface atoms. ADEPT solves elementary surface processes combined with Monte-Carlo methods to simulate chemically enhanced physical sputtering, and damage induced etching in three dimensions. In principle this method provides a highly accurate solution of physical models for dry-etch processes, since the calculations are at the near-atomic level.

#### **2.3.5. A Diffusion Equation Solution for Topography Simulation**

One of the only reported 3D simulators for lithography, deposition, and etching is 3D-MULSS, developed by M. Fujinaga *et. al.* at Mitsubishi Electric.<sup>64, 77, 78, 79</sup> This simulator has three key elements. The first is a topography representation model that employs the concentration of material. Bulk material has a concentration of 1.0. Air or vacuum has a

concentration of 0.0. The surface topography is determined by locating the region where the magnitude of the concentration gradient is much greater than zero. This describes the transition from empty space to the bulk material. The concentration gradient is also oriented in the direction of the surface normal. The second key element is a solution of the mass transfer problem with a diffusion equation. The change in concentration of material in a small volume element is related to the flow of material in or out of that element by the continuity equation. Additionally, the flux of etchant is related to the concentration. Combining these relations gives a diffusion equation for the concentration which is easily solved since the diffusion coefficient is constant. By comparing the evolution of the uniform concentration contour with the velocity of the surface due to uniform etching, it is seen that the diffusion time is actually the square of the etch time. The third key element, is that as the surface advances and material is removed, the cell concentrations must be set to 1.0. The diffusion equation must then be reinitialized with new Dirichlet boundary conditions.

This formulation has been applied to photoresist development, simple directional etching, and metal deposition via sputtering, with impressive results. However, some difficulties are apparent. The diffusion calculation must be performed over a very large region. Also, to perform a 5 second etch, a 25 second diffusion calculation must be performed, since the etch time is the square root of the diffusion time. Taken together, these two facts mean the algorithm is likely to be slow. Furthermore, it is not clear how the current implementation would be applied to directional etching that depends on the surface orientation. In that case, the diffusion coefficient would vary with the surface shape and thus with time, significantly complicating the numerical solution of the diffusion equation.

### **2.3.6. Solid Modeling Applied to Topography Simulation**

Solid modeling packages are readily applicable to the problem of representing 3D IC device structures. The OYSTER system developed at IBM<sup>80, 81, 82</sup> allows the creation of device structure solid models from a process description and mask-level information. However, OYSTER only allows basic geometric actions to be performed and full topography simulation of structures with rugged terrains has not been reported.

A crystal etching model for silicon has been applied to the Unigrafix modeling system.<sup>83</sup> A model for determining when facets appear was developed and incorporated in the Unigrafix etching algorithm. Several interesting shape evolutions have been demonstrated with this system.

Solid modeling systems hold promise for 3D topography simulation, but so far have found limited application. This is probably due to the difficulty with which most currently existing solid models represent surfaces described by thousands of points, and the enormous computational and memory requirements of these programs.

## **2.4. A NEW PLATFORM FOR 3D TOPOGRAPHY SIMULATION, SAMPLE-3D.**

Clearly there are many areas still open for investigation in the field of three-dimensional topography simulation. The analysis of section 2.2 showed that algorithms for connecting process models to the topography evolution are of fundamental importance. Many powerful 2D topography simulation tools profited from the original work that went into the string algorithm. However, no 3D topography simulator uses a 3D version of the string algorithm. Many of the other approaches that are used offer certain advantages, but they suffer from either inac-

curacy, applicability to a limited number of models, implementation complexity, or computational inefficiency.

SAMPLE-3D is a platform for investigating many of these issues. The implementation of surface motion algorithms will be coupled with physical models to provide a general 3D topography simulator. Throughout the development of SAMPLE-3D, several important points are kept in mind:

- **Accuracy:** The surface evolution algorithms must be accurate, and the accuracy should be controllable by varying the time step or the density of elements describing the surface.
- **Efficiency:** When possible, all algorithms affecting the surface evolution should be approximately linear with the surface area. In other words, if the surface area doubles, the run time should merely double and not quadruple.
- **Compatibility:** SAMPLE-3D will be first of all compatible with the 3D lithography simulation work of K. K. H. Toh at UC-Berkeley.<sup>†</sup> Second, SAMPLE-3D should be compatible with a variety of topography process models. Third, SAMPLE-3D should be compatible with engineering workstations - a criterion that will be met if run time and memory requirements are held down.
- **Expandability:** As new algorithms are developed, and as new process models become available, it should be possible to incorporate these in SAMPLE-3D
- **Usability:** Mechanisms for human interaction with SAMPLE-3D should be developed to enhance the usability of the software.

---

<sup>†</sup> Kenny Toh calls his 3D resist development simulator *SAMPLE-3D*. This work will build on Dr. Toh's software to provide full 3D topography simulation in *SAMPLE-3D*.

## REFERENCES

1. D.J. Eliot, *Integrated Circuit Fabrication Technology*, McGraw-Hill, New York, 1982.
2. I. Brodie and J.J. Muray, *The Physics of Microfabrication*, Plenum Press, New York, 1982.
3. in *VLSI Technology*, ed. S.M. Sze, McGraw-Hill Book Co., New York, 1988.
4. S. Wolf and R.N. Tauber, *Silicon Processing for the VLSI Era, 1 - Process Technology*, Lattice Press, Sunset Beach, California, 1986, reprint June, 1987.
5. W.R. Runyan and K.E. Bean, *Semiconductor Integrated Circuit Processing Technology*, Addison-Wesley, Reading, Massachusetts, 1990.
6. P. Burggraaf, "Wet Etching Today," *Semiconductor International*, p. 48, February 1983.
7. C. Murray, "Wet Etching Update," *Semiconductor International*, pp. 80-85, May 1986.
8. F.H. Dill, "Optical Lithography," *IEEE Transactions on Electron Devices*, vol. ED-22, no. 7, pp. 440-444, July 1975.
9. D. MacArthur, "Chemical Etching of Metals," in *Etching for Pattern Definition*, p. 76, Electrochemical Society, 1976.
10. K.E. Bean, "Anisotropic Etching of Si," *IEEE Transactions on Electron Devices*, vol. ED-25, p. 1185, 1978.
11. D.L. Flamm, "Basic Chemistry and Mechanisms of Plasma Etching," *Journal of Vacuum Science Technology B1*, pp. 23-30, 1983.
12. R.A. Morgan, *Plasma Etching in Semiconductor Fabrication*, Elsevier Science Publishers, B. V., New York, 1985.



13. J.W. Coburn, *Plasma Etching and Reactive Ion Etching*, p. 15, American Vacuum Society, New York, 1982.
14. D. L. Flamm and G. K. Herb, "Plasma Etching Technology - an Overview," in *Plasma Etching, an Introduction*, ed. D.M. Manos and D.L. Flamm, pp. 1-90, Academic Press, Boston, 1989.
15. S. Matsuo and Y. Adachi, "Reactive Ion Beam Etching Using a Broad Beam ECR Ion Source," *Japan Journal of Applied Physics*, vol. 21, p. L4, 1982.
16. R. Glang, "Vacuum Evaporation," in *Handbook of Thin Film Technology*, ed. L. Maissel and R. Glang, pp. 1-130, McGraw-Hill, New York, 1970.
17. T. Strahl, "Flash Evaporation, An Alternative to Magnetron Sputtering in the Production of High-Quality Aluminum Alloy Films," *Solid State Technology*, pp. 78-82, December 1978.
18. W.R. Grove, *Philosophical Transactions of the Royal Society*, p. 87, London, 1852.
19. J.L. Vossen and J.J. Cuomo, "Glow Discharge Sputter Deposition," in *Thin Film Processes*, ed. J.L. Vossen and W. Kern, pp. Chapter II-1, Academic Press, New York, 1978.
20. R.S. Nowicki, "The RF Diode Co-Deposition of Refractory Metal Silicides," *Solid State Technology*, p. 95, November 1980.
21. N. Kumar and et. al., "Fabrication of RF Reactively Sputtered TiN Thin Films," *Semiconductor International*, pp. 100-104, April 1987.
22. V. Hoffman, "High Rate Magnetron Sputtering for Metallizing Semiconductor Devices," *Solid State Technology*, pp. 57-61, December 1976.
23. L. Hartsough, "Resistivity of Bias-Sputtered Ti-W Films," *Thin Solid Films*, vol. 64, p. 17, 1979.

24. K. Kikuta, T. Kikkawa, and H. Aoki, "0.25  $\mu$ m Contact Hole Filling by Al-Ge Reflow Sputtering," *1991 Symposium on VLSI Technology, Digest of Technical Papers*, pp. 35-36, Oiso, Japan, May 28-30, 1991.
25. W. Kern and V.S. Ban, "Chemical Vapor Deposition of Inorganic Thin Films," in *Thin Film Processes*, ed. J. L. Vossen and W. Kern, pp. 257-331, New York, 1978.
26. W. Kern, "Deposited Dielectrics for VLSI," *Semiconductor International*, vol. 8, no. 7, p. 122, July 1985.
27. R.A. Levy and M.L. Green, "Low Pressure Chemical Vapor Deposition of Tungsten and Aluminum for VLSI Applications," *Journal of the Electrochemical Society*, vol. 134, pp. 37C-49C, 1987.
28. R.H. Wilson, R.W. Stoll, and M.A. Calacone, *Tungsten and Other Refractory Metals for VLSI Applications*, 1, p. 35, 1986.
29. P. Singer, "Techniques of Low Pressure CVD," *Semiconductor International*, p. 72, May 1984.
30. B. Gorowitz, T.B. Gorczyca, and R.J. Saia, "Applications of PECVD in VLSI," *Solid State Technology*, p. 197, June 1985.
31. J.Y. Chen and R. Henderson, "Photo-CVD for VLSI," *Journal of the Electrochemical Society*, vol. 131, p. 2147, September 1984.
32. T. Gocho, Y. Morita, and J. Sato, "Trench Isolation Technology for 0.35 $\mu$ m Device by Bias ECR CVD," *1991 Symposium on VLSI Technology, Digest of Technical Papers*, pp. 87-88, Oiso, Japan, May 28-30, 1991.
33. P.-L. Pai and C. Ting, *Proceedings IEEE VLSI Multilevel Interconnection Conference*, pp. 258-354, Santa Clara, California, June 1990.

34. J. Bloem and L.J. Gilling, "Epitaxial Growth by Chemical Vapor Deposition," in *VLSI Electronics*, ed. N. G. Einspruch and H. Huffs, vol. 12, p. 89, Academic Press, Orlando, 1985.
35. H.M. Liaw, "Surface Morphology of Selective Epitaxial Silicon," in *Semiconductor Silicon*, p. 260, Electrochemical Society, Pennington, New Jersey, 1986.
36. A.R. Neureuther and W.G. Oldham, "Simulation of Optical Lithography," in *Advances in CAD for VLSI Process and Devices Simulation*, ed. W. L. Engl, Elsevier Science Publishers, B.V. , Amsterdam, 1985.
37. A.R. Neureuther, "Topography Simulation Tools," *Solid State Technology*, pp. 71-75, March 1986.
38. *Process and Device Modeling*, Elsevier Science Publishers, B.V., Amsterdam, 1986.
39. W. Fichtner, "Physics of VLSI Processing and Process Simulation," in *Silicon Integrated Circuits Part C*, ed. Dawon Kahng, Academic Press, Orlando, Florida, 1985.
40. S. Wolf, *Silicon Processing for the VLSI Era*, 2 - Process Integration, Chapter 9 , Lattice Press, Sunset Beach, California, 1990.
41. W.G. Oldham, S.N. Nandgaonkar, A.R. Neureuther, and M.M. O'Toole, "A General Simulator for VLSI Lithography and Etching Processes: Part I - Application to Projection Lithography," *IEEE Transactions on Electron Devices*, vol. ED-26, no. 4, pp. 717-722, April 1979.
42. W.G. Oldham, A.R. Neureuther, C. Sung, J.L. Reynolds, and S.N. Nandgaonkar, "A General Simulator for VLSI Lithography and Etching Processes: Part II - Application to Deposition and Etching," *IEEE Transactions on Electron Devices*, vol. ED-27, no. 8, pp. 1455-1459, August 1980.

43. B.R. Penumalli, "A Comprehensive Two-Dimensional VLSI Process Simulation Program, BICEPS," *IEEE Transactions on Electron Devices*, vol. ED-30, September 1983.
44. J. Lorenz, J. Pelka, H. Ryssel, A. Sachs, A. Seidl, and M. Svoboda, "COMPOSITE - A Complete Modeling Program of Silicon Technology," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. CAD-4, no. 4, pp. 421-430, April 1985.
45. S. Tazawa, S. Matsuo, and K. Saito, "Unified Topography Simulator for Complex Reaction Including Both Deposition and Etching," *1989 Symposium on VLSI Technology, Digest of Technical Papers*, pp. 45-46, May 1989.
46. "DEPICT-2," *Technology Modeling Associates*, 1990.
47. S. Yamamoto, T. Kume, M. Ohgo, T. Matsuzawa, S. Tachi, and H. Sunami, "A Two-Dimensional Etching Profile Simulator: ESPRIT," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. CAD-6, no. 3, pp. 417-421, March 1987.
48. L. Borucki, H.H. Hansen, and K. Varahramyan, "FEDDS - A 2D semiconductor fabrication process simulator," *IBM Journal of Research and Development*, vol. 29, no. 3, pp. 263-276, May 1985.
49. C.A. Mack, "PROLITH: A Comprehensive Optical Lithography Model," *Proceedings of SPIE: Optical Microlithography IV*, vol. 538, pp. 207-220, March 1985.
50. T. Smy, K.L. Westra, and M.J. Brett, "Simulation of Density Variation and Step Coverage for a Variety of Via/Contact Geometries Using SIMBAD," *IEEE Transactions on Electron Devices*, vol. ED-37, no. 3, pp. 591-598, March 1990.
51. J. McVittie, J. Rey, L.-Y. Cheng, A. Bariya, S. Ravi, and K. Saraswat, "SPEEDIE: A Profile Simulator for Etching and Deposition," *TECHCON '90, Extended Abstract*

- Volume*, pp. 16-19, Semiconductor Research Corporation, San Jose, California, October 16-18, 1990.
52. A. Gerodolle, C. Corbex, A. Poncet, T. Pedron, and S. Martin, "TITAN 5: a two-dimensional process and device simulator," in *Software Tools for Process, Device and Circuit Modelling*, pp. 56-67, Boole Press, Dublin, Ireland, 1989.
  53. S. Nassif, A.J. Strojwas, and S.W. Director, "FABRICS II: A Statistically Based IC Fabrication Process Simulator," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. CAD-3, pp. 20-46, January 1984.
  54. R.E. Jewett, P.I. Hagouel, A.R. Neureuther, and T. Van Duzer, "Line-Profile Resist Development Simulation Techniques," *Polymer Eng. Sci.*, vol. 17, no. 6, pp. 381-384, June 1977.
  55. J.I. Ulacia-Fresnedo, "Theoretical and Experimental Considerations Necessary to Build a Dry-etching Process Simulator," Technical report No. G833-1, Integrated Circuits Laboratory, Stanford University, 1988.
  56. T. Thurgate, "Segment Based Etch Algorithm and Modeling," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. CAD-10, no. 9, pp. 1101-1109, September 1991.
  57. J. Bauer, "Modelle fuer den fotolithografischen Prozess," *Feingeraetetechnik*, vol. 29, p. 127ff, 1980.
  58. F. Jones and J. Paraszczak, "RD3D (Computer Simulation of Resist Development in Three Dimensions)," *IEEE Transactions on Electron Devices*, vol. ED-28, no. 12, pp. 1544-1552, December 1981.
  59. Y. Hirai, M. Sasugo, M. Endo, K. Ikeda, S. Tomida, and S. Hayama, "Three Dimensional Process Simulation for Photo and Electron Beam Lithography and Estimations of

- Proximity Effects," *Symposium on VLSI Technology, Digest of Technical Papers*, p. 15, 1987.
60. Y. Hirai, S. Tomida, K. Ikeda, M. Sasago, M. Endo, S. Hayama, and N. Nomura, "Three-Dimensional Resist Process Simulator PEACE (Photo and Electron Beam Lithography Analyzing Computer Engineering System)," *IEEE Transactions on Computer-Aided Design of Integrated Circuits*, vol. CAD-10, no. 6, pp. 802-807, June 1991.
  61. W. Henke, D. Mewes, M. Weiss, G. Czech, and R. Schiessl-Hoyler, "Simulation of Defects in 3-Dimensional Resist Profiles in Optical Lithography," *Microelectronic Engineering*, vol. 13, pp. 497-501, 1991.
  62. H.K. Oertel, M. Weiss, J. Chlebek, H.L. Huber, R. Dammel, C.R. Lindley, J. Lingnau, and J. Theis, "Resist Modeling near Resolution and Sensitivity Limits in X-Ray Lithography," *Proceedings SPIE*, vol. 1089, p. 283ff., 1989.
  63. T. Ushirogouchi, Y. Onishi, and T. Tada, "Resist profile simulation for photoresist composition optimization," *Journal of Vacuum Science Technology B*, vol. 8, no. 6, pp. 1418-1422, November/December 1990.
  64. M. Fujinaga, T. Kunikiyo, T. Uchida, N. Kotani, A. Osaki, and Y. Akasaka, "New Topography Expression Model and 3D-Topography Simulation of Al-Sputter Deposition, Etching, and Photolithography," *IEDM Technical Digest*, pp. 905-908, San Francisco, December 9-12, 1990.
  65. J. Pelka, *personal communication*, 1990. Berkeley, CA
  66. K.K.H. Toh, "Algorithms for Three-Dimensional Simulation of Photoresist Development," Memo. No. UCB/ERL M90/123, p. 31, Ph.D. Dissertation, University of California, Berkeley, December 14, 1990.

67. M. Born and E. Wolf, *Principles of Optics, Sixth Edition*, Pergamon Press, London 1980.
68. P.I. Hagouel, *X-ray Lithographic Fabrication of Blazed Diffraction Gratings*, Ph.D. Dissertation, University of California, Berkeley, 1976.
69. T. Matsuzawa, T. Ito, and H. Sunami, "Three-dimensional Photoresist Image Simulation on Flat Surfaces," *IEEE Transactions on Electron Devices*, vol. ED-32, no. 9, pp. 1781-1783, September 1985.
70. A. Moniwa, T. Matsuzawa, T. Ito, and H. Sunami, "A Three-Dimensional Photoresist Imaging Process Simulator for Strong Standing-Wave Effect Environment," *IEEE Transactions on Computer Aided Design of Integrated Circuits*, vol. CAD-6, no. 3, pp. 431-437, May 1987.
71. L. Jia, W. Jian-kun, and W. Shao-jun, "Three-Dimensional Development of Electron Beam Exposed Resist Patterns Simulated by Using Ray Tracing Model," *Microelectronic Engineering*, vol. 6, pp. 147-151, 1987.
72. K.Y. Lee, Y.H. Kim, and C.G. Hwang, "New Three-Dimensional Simulator for Electron Beam Lithography," *1991 International Workshop on VLSI Process and Device Modeling*, pp. 44-45, Oiso, Japan, May 26-27, 1991.
73. E. Barouch, B. Bradie, H. Fowler, and S.V. Babu, "Three-Dimensional Modeling of Optical Lithography for Positive Photoresists," *Interface'89 : Proceedings of KTI Microelectronics Seminar*, pp. 123-136, November 1989.
74. T. Ishizuka, "Three-Dimensional Simulation in Photoresist Development," *IEICE (In Japanese)*, vol. J73-C-II, no. 11, pp. 775-785, November 1990.
75. S. Yamaguchi, "Three-Dimensional Simulation Technology," *Semiconductor World (In Japanese)*, pp. 149-153, January 1990.

76. J. Pelka, "Simulation of Ion-Enhanced Dry-etch Processes," *Microelectronic Engineering*, vol. 13, pp. 487-491, 1991.
77. M. Fujinaga, N. Kotani, H. Oda, M. Shirahata, H. Genjo, T. Katayama, T. Ogawa, and Y. Akasaka, "Three Dimensional Topography Simulation model Using Diffusion Equation," *IEDM Technical Digest*, pp. 332-335, San Francisco, CA, December 11-14, 1988.
78. M. Fujinaga, N. Kotani, T. Kunikiyo, H. Oda, M. Shirahata, and Y. Akasaka, "Three-Dimensional Topography Simulation Model : Etching and Lithography," *IEEE Transactions on Electron Devices*, vol. ED-37, no. 10, pp. 2183-2192, October 1990.
79. M. Fujinaga, "3D-Topography Simulator for Sequential Processes," *1991 International Workshop on VLSI Process and Device Modeling*, pp. 80-83, Oiso, Japan, May 26-27, 1991.
80. G.M. Koppelman and M.A. Wesley, "Oyster: A Study of Integrated Circuits as Three-Dimensional Structures," *IBM Journal of Research and Development*, vol. 27, p. 149, 1983.
81. R.N. Wolf, M.A. Wesley, J.C. Kyle Jr., F. Gracer, and W.J. Fitzgerald, "Solid Modeling for Production Design," *IBM Journal of Research and Development*, vol. 31, no. 3, pp. 277-295, May 1987.
82. R.C. Evans, G. Koppelman, and V.T. Rajan, "Shaping geometric objects by cumulative translational sweeps," *IBM Journal of Research and Development*, vol. 31, no. 3, pp. 343-360, May 1987.
83. W.F. Foote, "Simulation of Anisotropic Crystal Etching," *M.S. Project Report*, University of California, Berkeley, September 12, 1990.



## CHAPTER 3

### ANOTHER LOOK AT 3D CELL-REMOVAL ALGORITHMS

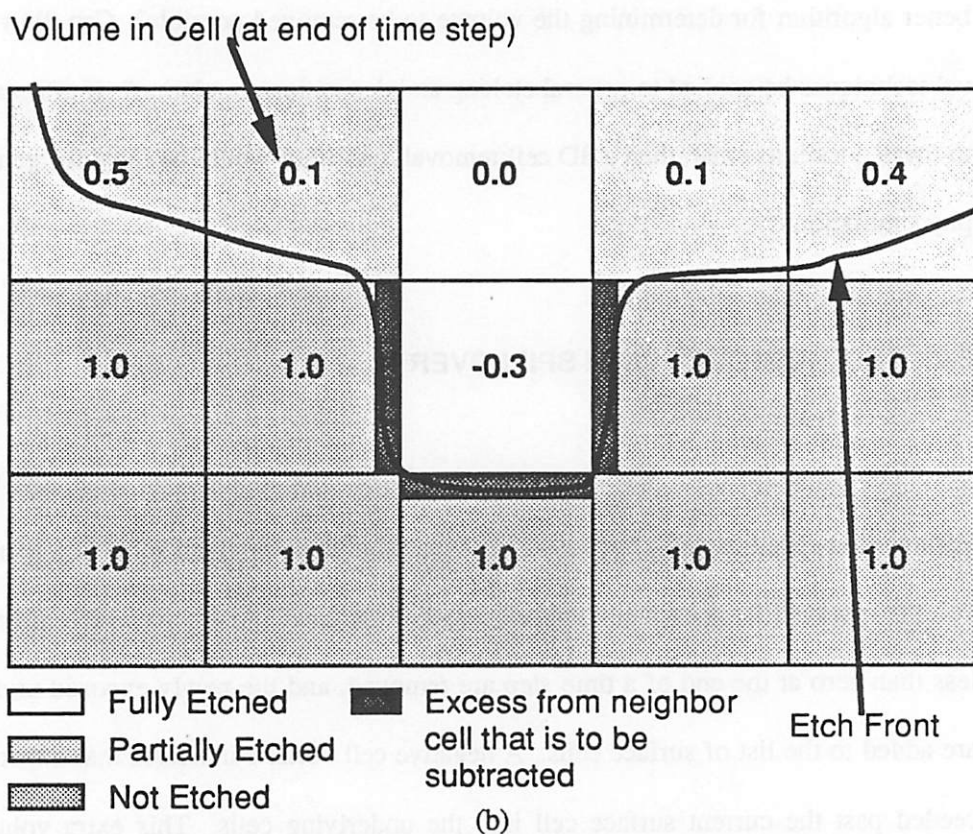
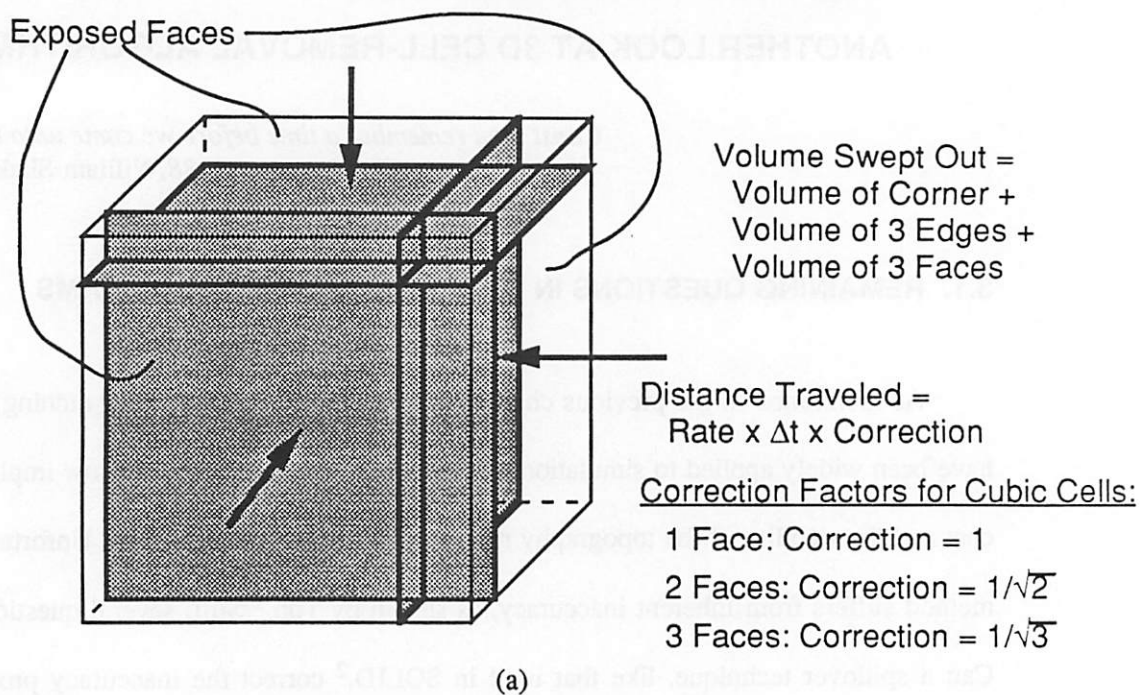
*Canst thou remember a time before we came unto this cell?*  
-The Tempest I.II.38, William Shakespeare

#### 3.1. REMAINING QUESTIONS IN 3D CELL-REMOVAL ALGORITHMS

As mentioned in the previous chapter, 3D cell-based algorithms for etching simulation have been widely applied to simulation of lithography development. The low implementation cost and the stability of the topography representation motivate this trend. Unfortunately, this method suffers from inherent inaccuracy, as shown by Toh.<sup>1</sup> Still, several questions remain. Can a spillover technique, like that used in SOLID,<sup>2</sup> correct the inaccuracy problem? Is a better algorithm for determining the volume to be removed possible? Can 3D volume removal techniques be applied to general etching and deposition simulation? If affirmative answers to these questions exist, then a 3D cell removal algorithm would be ideal for general topography simulation.

#### 3.2. CELL-REMOVAL WITH SPILLOVER

The basic spillover technique was developed, not to improve accuracy, but rather to improve the speed of cell-removal algorithms. If a predetermined time step is used, then the volume removed from each surface cell may be calculated. Any cells which have a volume of less than zero at the end of a time step are removed, and the newly exposed underlying cells are added to the list of surface cells. A negative cell volume indicates that the etch front proceeded past the current surface cell into the underlying cells. This extra volume must be deducted from the newly exposed cells to preserve a valid surface representation. This tech-



**Fig. 3.1:** A volume removal algorithm for etching simulation. (a) Method for determining volume to be removed. (b) A spillover technique to correct for fixed time step.

nique avoids the need to locate the one cell with minimum time to removal at each time step, thereby improving the algorithm run time from  $O(N^3)$  to  $O(N^2)$ , where  $N$  is the number of cells per unit of linear dimension.<sup>†</sup> To save memory, full information is allocated only for surface cells. Other cells point to a dummy structure indicating whether the cell is full or empty. The full memory requirement is thus:  $Bytes = (int\ size) \times (N_{cells} + 5N_{surfacecells}) + (float\ size)N_{rate}$ .

A method for approximating the volume removed during a time step is shown in Fig. 3.1a. If a neighboring cell is devoid of material, the cell face shared with that neighbor is exposed. All exposed faces are moved into the cell, with a distance given by the product of the etch rate at the cell and the time step. If only one face is exposed, the calculation is straightforward. The volume removed is the product of the face area and the distance traveled. If two edge-sharing faces are exposed, a correction factor of  $1/\sqrt{2}$  for cubic cells is include in the rate to slow the advance of the planes. If three vertex-sharing planes are exposed the correction factor is  $1/\sqrt{3}$  for cubic cells. The volume removed is estimated as the amount swept out by all of the moving planes:

$$\begin{aligned}
 Volume = & d_{xl}d_{yl}d_{zl} + d_{xl}d_{yl}d_{zh} + d_{xl}d_{yh}d_{zl} + d_{xl}d_{yh}d_{zh} + d_{xh}d_{yl}d_{zl} + d_{xh}d_{yl}d_{zh} + d_{xh}d_{yh}d_{zl} + d_{xh}d_{yh}d_{zh} \\
 & + (\Delta x - d_{xh} - d_{xl}) \cdot (d_{yl}d_{zl} + d_{yl}d_{zh} + d_{yh}d_{zl} + d_{yh}d_{zh}) \\
 & + (\Delta y - d_{yh} - d_{yl}) \cdot (d_{xl}d_{zl} + d_{xl}d_{zh} + d_{xh}d_{zl} + d_{xh}d_{zh}) \\
 & + (\Delta z - d_{zh} - d_{zl}) \cdot (d_{xl}d_{yl} + d_{xl}d_{yh} + d_{xh}d_{yl} + d_{xh}d_{yh}) \\
 & + (\Delta x - d_{xh} - d_{xl}) \cdot (\Delta z - d_{zh} - d_{zl}) \cdot (d_{yl} + d_{yh}) \\
 & + (\Delta y - d_{yh} - d_{yl}) \cdot (\Delta z - d_{zh} - d_{zl}) \cdot (d_{xl} + d_{xh}) \\
 & + (\Delta x - d_{xh} - d_{xl}) \cdot (\Delta y - d_{yh} - d_{yl}) \cdot (d_{zl} + d_{zh})
 \end{aligned} \tag{3.1}$$

where  $d_{il}, d_{ih}$  are the etch distances ( $rate \times \Delta t$ ) of the two planes perpendicular to the  $i$ -axis, and  $\Delta x, \Delta y, \Delta z$  are the cell dimensions. The correction factors for one, two, or three adjacent faces

<sup>†</sup> The "O-notation" was first introduced by P. Bachman in *Analytische Zahlentheorie* (1892). It will be used to represent the approximate run time of algorithms.  $O(f(N))$ , where  $N$  is a positive integer, means there exists a positive constant  $M$  such that  $|O(f(N))| \geq M|f(N)|$ , for all  $N \geq N_0$ .

are derived as if the etch front were advancing along an inclined plane placed on top of the cells, and are given as follows, where each of  $i$ ,  $j$ , and  $k$ , represent three different coordinate axes:

$$\begin{aligned}
 \text{face } i: \quad & \text{correction}_i = 1.0; \\
 \text{faces } i \text{ and } j: \quad & \text{correction}_i = \sin(\arctan(\Delta i / \Delta j)) \\
 & \text{correction}_j = \cos(\arctan(\Delta i / \Delta j)) \\
 \text{faces } i, j, \text{ and } k: \quad & \text{correction}_i = \sin(\arctan(\Delta k \sqrt{\Delta i^2 + \Delta j^2} / \Delta i \Delta j)) \cdot \cos(\arctan(\Delta i / \Delta j)) \\
 & \text{correction}_j = \sin(\arctan(\Delta k \sqrt{\Delta i^2 + \Delta j^2} / \Delta i \Delta j)) \cdot \sin(\arctan(\Delta i / \Delta j)) \\
 & \text{correction}_k = \cos(\arctan(\Delta k \sqrt{\Delta i^2 + \Delta j^2} / \Delta i \Delta j))
 \end{aligned} \tag{3.2}$$

The volume removal calculation is always performed as if the cell were completely full, until it is removed entirely. This tends to overestimate the amount of material removed from the surface cell, but the underlying cells do not etch until the surface cell is completely empty. Thus the underlying cells are underetched. The combination of overetch and underetch yields the correct overall amount of removed material.

The spillover technique is illustrated in Fig. 3.1b. Excess volume removed from a surface cell is evenly divided among all the cells that share a face with the original cell, and have not been etched. For a small time step, this excess volume is typically only a few percent of the total cell volume.

Fig 3.2a shows a result for uniform spherical etching, using this algorithm. The etch rate was 2.5 cells per second, the etch time was 16 seconds and the time step was 0.02 seconds. The full calculation required only 26.4 seconds on an IBM Powerstation 530 and the total memory used was 500Kbytes. This is about an order of magnitude better than a similar calculation with the PEACE program on a FACOM VP-100 supercomputer.<sup>3</sup> The number of cells removed is 32270. A perfect spherical etch with a radius of 40 cells would result in a removed

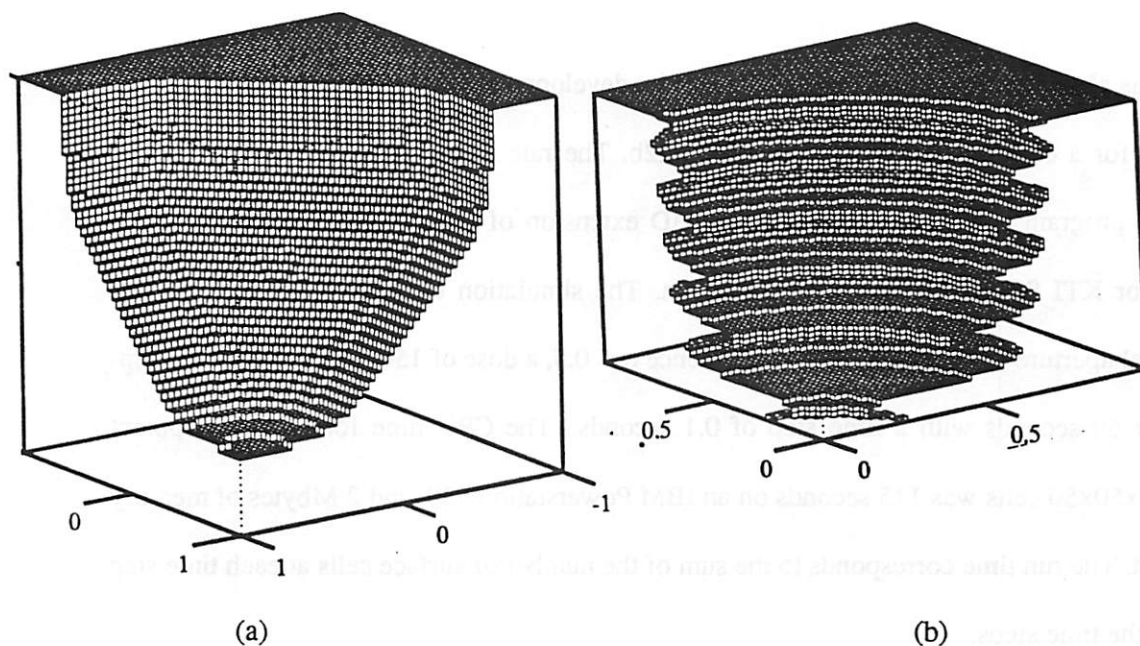


Fig. 3.2: Cell-removal algorithm with spillover. (a) Facet formation in spherical etching  
(b) KTI 820 contact hole, development for 60 s, g-line, N.A.=0.28,  $\sigma=0.7$ ,  $150 \text{ mJ/cm}^2$

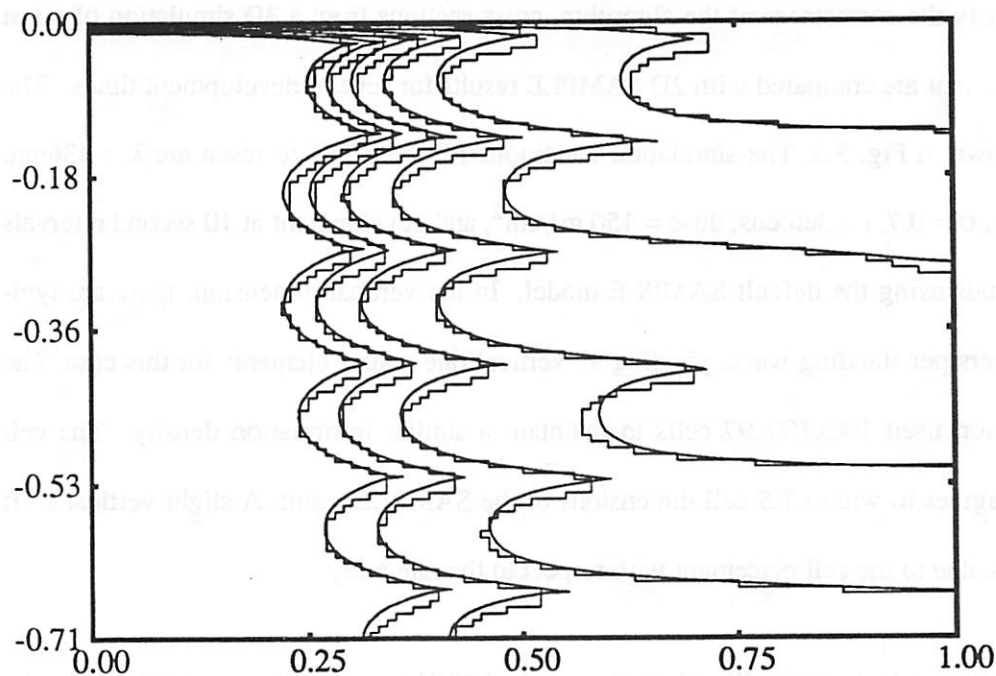


Fig. 3.3: Cross-section of resist line development. Cell-removal results are superimposed on SAMPLE 1.8a results. KTI 820 resist,  $\lambda=436\text{nm}$ , N.A.=0.28, no defocus, dose= $150\text{mJ/cm}^2$ . Development at 10s intervals to 60s using Kim model:  $R1=0.23$ ,  $R2=0.0016$ ,  $R3=5.6$ ,  $R4=0.25$ ,  $R5=0.62$ ,  $R6=0.08$ .  $100 \times 100 \times 97$  cells.

volume totaling 33510.3 cells. Although facets are evident, the error in the volume removed is only 3.7%.

This algorithm can be applied to lithography development with quite satisfactory results. A result for a contact hole is shown in Fig. 3.2b. The rate matrix was generated using the SPLAT<sup>4</sup> program for aerial imaging, and a 3D extension of the SAMPLE bleaching algorithm<sup>1</sup> for KTI 820 resist exposure simulation. The simulation conditions are  $\lambda = 436\text{nm}$ , numerical aperture N.A. = 0.28, partial coherence  $\sigma = 0.5$ , a dose of  $150\text{ mJ/cm}^2$ , and development for 60 seconds with a time step of 0.1 seconds. The CPU time for the development using  $50 \times 50 \times 50$  cells was 115 seconds on an IBM Powerstation 530, and 2 Mbytes of memory was used. The run time corresponds to the sum of the number of surface cells at each time step over all the time steps.

To verify the correctness of the algorithm, cross-sections from a 3D simulation of resist line development are compared with 2D SAMPLE results for several development times. The result is shown in Fig. 3.3. The simulation conditions for the KTI 820 resist are  $\lambda = 436\text{nm}$ , N.A. = 0.28,  $\sigma = 0.7$ , no defocus, dose =  $150\text{ mJ/cm}^2$ , and development at 10 second intervals for 60 seconds using the default SAMPLE model. In the vertical dimension, there are typically 16 layers per standing wave, yielding 97 vertical rate matrix elements for this case. The cell simulation used  $100 \times 100 \times 97$  cells to maintain a similar information density. The cell simulation agrees to within 1.5 cell dimensions of the SAMPLE result. A slight vertical shift also appears, due to the cell placement with respect to the rate array.

The spillover technique will reduce the required CPU time, but it cannot correct for the inaccuracy inherent in the cell-removal algorithm. The algorithm can be applied to lithography development, since the varying etch rate throughout the exposed photoresist tends to

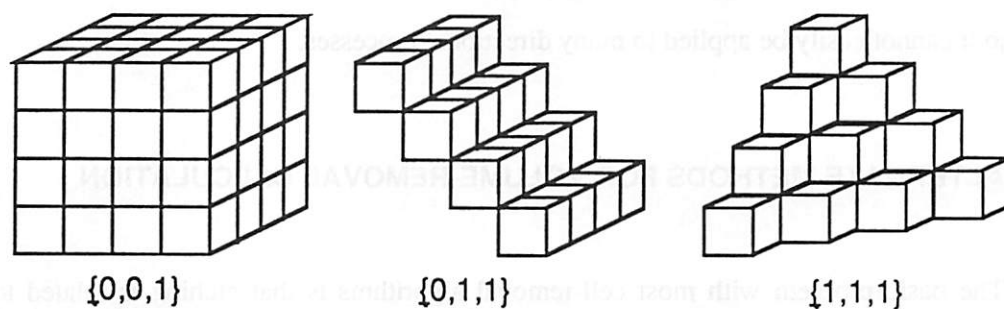


Fig. 3.4: Preferred crystallographic planes in basic cell-removal algorithm.

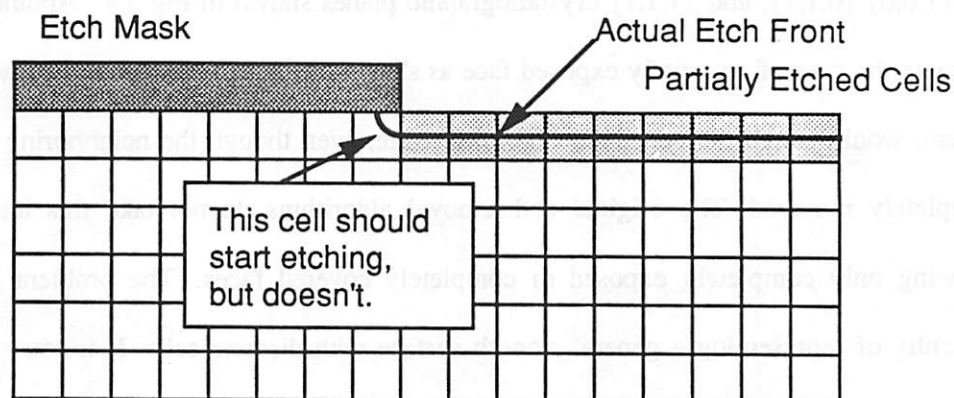


Fig. 3.5: Partially exposed cell under etch mask.

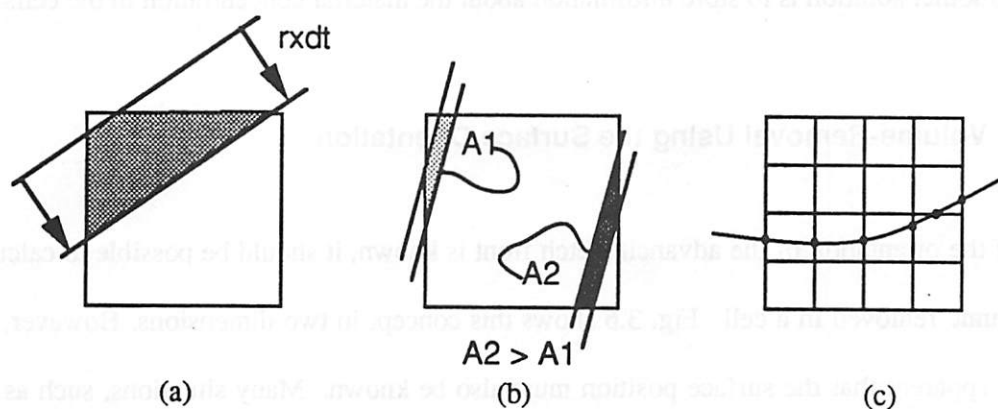


Fig. 3.6: Volume-removal using the surface orientation (2D example). (a) Region is swept out by advancing surface. (b) Different amount of material is swept out depending on starting location. (c) Represent surface as intersections along cell edges.

smear out the rigid crystal structure. For the general etch problem however, faceting cannot be ignored. This algorithm also does not provide information about the true surface orientation, so it cannot easily be applied to many directional processes.

### **3.3. ALTERNATE METHODS FOR VOLUME-REMOVAL CALCULATION**

The basic problem with most cell-removal algorithms is that etching is related to the movement of cell faces and not to the surface normal of the advancing etch front. This favors the  $\{1,0,0\}$ ,  $\{0,1,1\}$ , and  $\{1,1,1\}$  crystallographic planes shown in Fig 3.4. Another problem arises in the case of a partially exposed face as shown in Fig. 3.5. In the case of wet etching, etchant would attack the cell under the mask edge, even though the neighboring cell is not completely removed. The original cell-removal algorithms do not take this into account, allowing only completely exposed or completely covered faces. The problem lies in the difficulty of representing a general smooth surface with discrete cells. It follows that better methods for calculating the cell removal require a better surface representation. One possible solution is to store information about the orientation and location of the actual etch front in the cells. Another solution is to store information about the material concentration in the cells.

#### **3.3.1. Volume-Removal Using the Surface Orientation**

If the orientation of the advancing etch front is known, it should be possible to calculate the volume removed in a cell. Fig. 3.6 shows this concept in two dimensions. However, it is readily apparent that the surface position must also be known. Many situations, such as that shown in Fig. 3.6b can occur in which the volume removed is as much a function of position as orientation. Furthermore, if the etch front is curved, the surface orientation must be carefully calculated to approximate the actual surface as closely as possible. Ultimately, the best



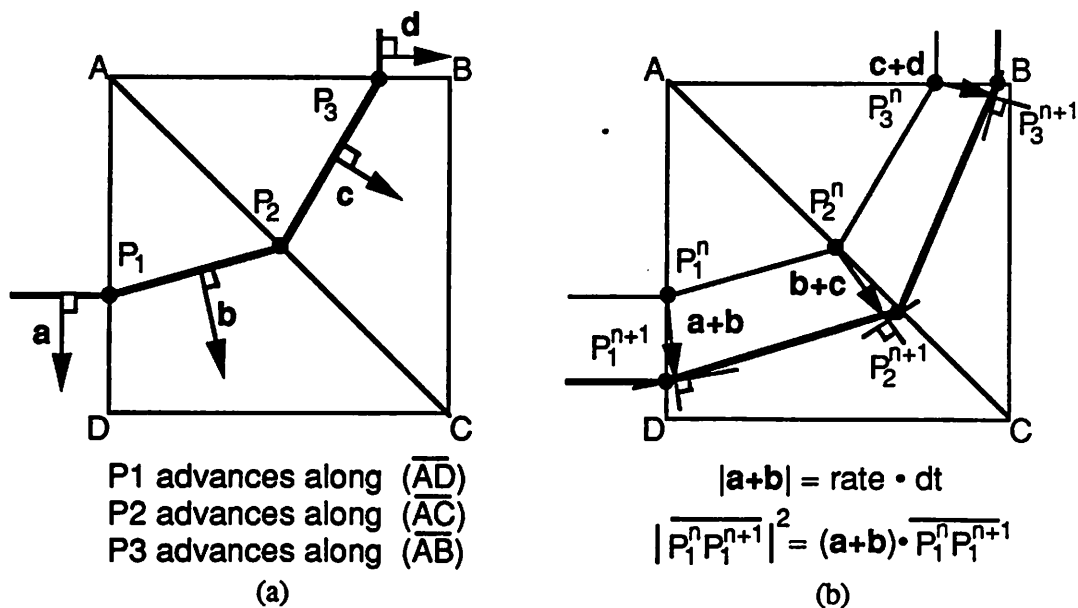


Fig. 3.7: A 2D network method (similar to *Ishizuka*<sup>5</sup>). (a) Cell divided into triangles. (b) Front motion for one time step, using network method with uniform rate.

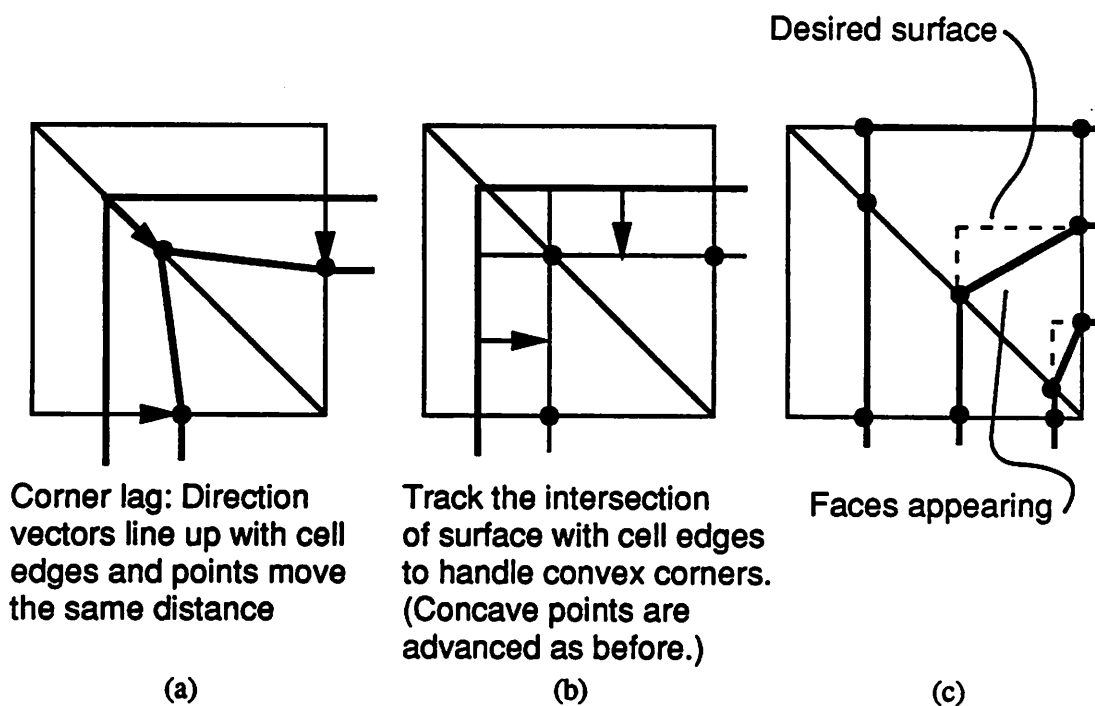


Fig. 3.8: Convex corners and the network method. (a) Corner lag. (b) Using segment advancement to correct for corner lag. (c) Faces appearing in method of (b).

way to represent the surface is to keep track of its intersection with the edges of the cells as shown in Fig. 3.6c. This is the basis of the network method by Ishizuka,<sup>5</sup> which was mentioned in Section 2.3.3.

For best results, The network method divides up cells even further, including the cell body and face diagonals. In 2D, a square cell would be divided into smaller triangles and the surface would be represented by its intersection with the triangle edges. The surface advancement for a simple 2D case proceeds as shown in Fig. 3.7a. A surface point is advanced by first averaging the orientations of all the surface edges at that point to calculate the direction of movement. The front motion along that direction is related back to motion along the cell edges and the surface advances as shown in Fig. 3.7b. Since the edge orientations are known, it is not difficult to calculate the distance each point travels.

This method improves the accuracy of the volume-removal calculation by retaining surface information. However, some difficulties remain. One problem is the way this algorithm handles etching of convex corners. Corner lag occurs as shown in Fig. 3.8a. In 2D it is possible to correct for this by locating the intersection of two advancing segments as shown in Fig. 3.8b. However, even using this technique new faces may appear and disappear as shown in Fig. 3.8c. This is probably not a problem for isotropic etching, but care should be taken if this technique is applied to directional etching. In 3D the problem is more difficult, since now the location of advancing planes must be determined. Unlike two line segments in 2D, the intersection of several planes is not necessarily a unique point. Fortunately, this problem does not propagate beyond one cell, as long as no surface point advances past a cell vertex in one time step. This restriction will increase the algorithm run time, since the maximum allowed time step must now be determined at each advancement step. Run times for 60x60x50 cells in the 3D implementation by Ishizuka are from 3 to 8 hours on a CONVEX-C210 high-speed

vector-processing CPU! The amount of information that must be stored in each cell has also increased significantly. For practical 3D simulations with 250,000 to 1,000,000 cells, the parameters necessary to describe the surface intersections with the cell edges require an additional several megabytes of memory storage. Dynamic allocation of the active cells would be needed for most engineering workstations. Nevertheless, methods which combine the accuracy of surface advancement with the stability of cell-based topography representations may prove quite useful. This issue is explored further in Chapter 5.

### 3.3.2. Volume Removal With Variable Material Density

The above cell-based algorithms all view the material boundary as an abrupt change in material density from zero to some finite value. An alternate view is to consider a varying change in material concentration. At the etch front, material is reduced in density until enough is removed that a cell is considered deleted. As the density is reduced, the material can be considered porous allowing underlying cells to be affected. This raises two questions: how far into the material should the porosity be allowed, and how exactly is the etch front related to the change in material density?

To solve this problem accurately, a firm mathematical foundation must be provided. The principle of mass-conservation can be expressed as

$$\frac{\partial C}{\partial t} + \nabla \cdot J = G \quad (3.3)$$

where  $C$  is the material concentration,  $J$  is the flux density of material and  $G$  is the generation rate (typically  $G=0$ ). The flux  $J$  can be related to the material concentration:

$$J = -D \nabla C \quad (3.4)$$

Together, equations 3.3 and 3.4 result in a diffusion equation:

$$\frac{\partial C}{\partial t} = \nabla^2(DC) \quad (3.5)$$

The method of Fujinaga *et. al.* mentioned in Section 2.3.5,<sup>6</sup> relates the solution of this diffusion equation to the expected etch front velocity. The boundary conditions at the interface of the etchant and the material require that the etchant concentration (in this case  $C$ ) be equal on both sides of the interface. The etchant concentration deep in the bulk material  $C_2$  is 0. It is also assumed that the etchant concentration  $C_1$  at the surface is less than the etchant concentration  $C_0$  in the developer. The etchant concentration along a vector perpendicular to the etch front can be given as follows:<sup>7</sup>

$$C(r, t) = C_0 \operatorname{erfc}(r/(2\sqrt{D_r t})) \quad (3.6)$$

The movement of the etch front where  $C(r, t) = C_1$  is given by solving for  $r$ :

$$r = 2\sqrt{D_r t} \operatorname{erfc}^{-1}(C_1/C_0) \quad (3.7)$$

The etch rate is simply related to the velocity of the moving etch front:

$$V_r = \frac{dr}{dt} = \sqrt{D_r/t} \operatorname{erfc}^{-1}(C_1/C_0) \quad (3.8)$$

From these relations, the etch front velocity  $V_r$  and the diffusion constant  $D_r$  satisfy:

$$r = V_r \cdot T = k\sqrt{D_r \cdot t} \quad (3.9)$$

where  $T$  is the etch time and  $t$  is the diffusion time, and  $k$  is a proportionality constant. If  $k$  is set to 1.0<sup>†</sup> and  $C_0$  is normalized to 1.0, then the concentration of the etch front surface is:

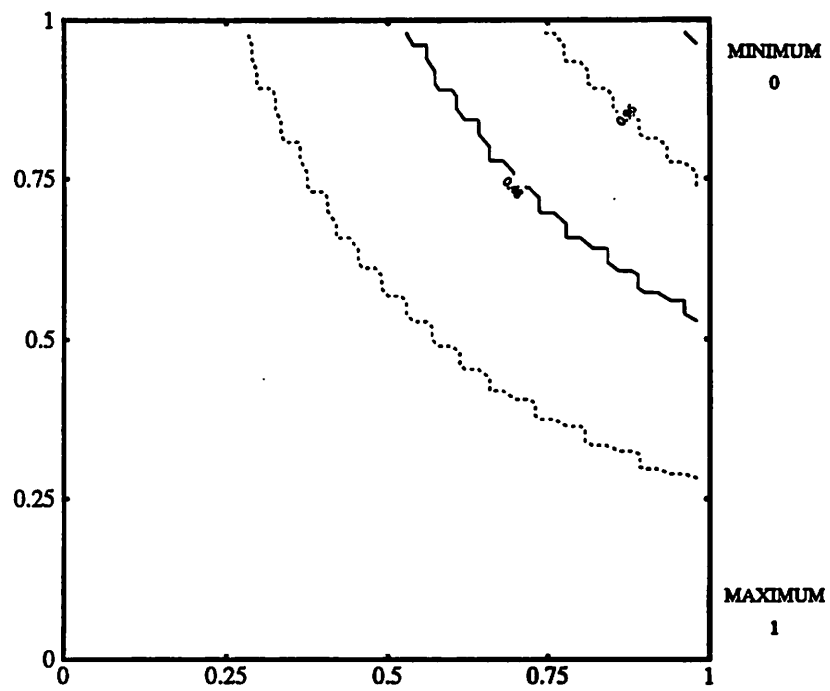
$$C_1 = \operatorname{erfc}[r/(2\sqrt{D_r t})] = \operatorname{erfc}(0.5) = 0.479 \quad (3.10)$$

Equation 3.5 can be discretized and expanded into three dimensions as:

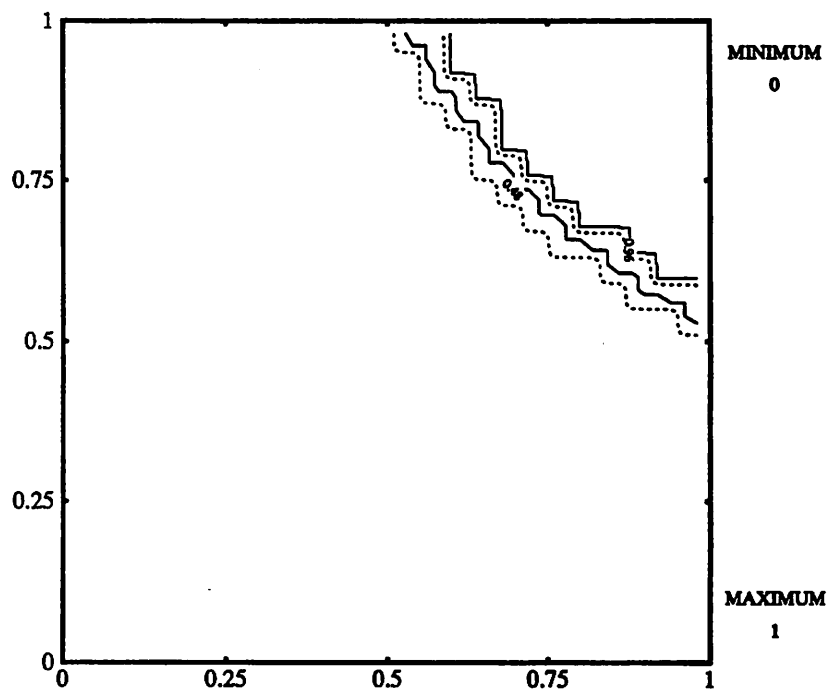
$$\frac{\Delta C}{\Delta t} = [V_x^2 \frac{\partial^2}{\partial x^2} C + V_y^2 \frac{\partial^2}{\partial y^2} C + V_z^2 \frac{\partial^2}{\partial z^2} C] \quad (3.11)$$

where  $V_x, V_y$ , and  $V_z$  are the etch rates along the Cartesian coordinate axes. This equation must be solved by iterating  $\Delta t$  until  $\sum_i \Delta t_i = t = T^2$ .

<sup>†</sup> Note that changing  $k$  will result in a different value for  $C_1$ . However, the position of contours determined using different  $k$  values will be the same, since the relationship between  $V$  and  $D$  and between  $T$  and  $t$  is similarly altered.



(a)



(b)

Fig. 3.9: Diffusion Equation Etch Algorithm. (a) 5.0 seconds etching at  $0.1\mu\text{m/s}$ .  
 (b) Same result, after reassigning bulk cells as 0.0, and air cells as 1.0.

A simple 2D implementation which estimates the partial derivatives of the concentration as the slope of the concentration over the cell width, gives marginally satisfactory results for a small enough time step. Uniform circular etching for 5 seconds at a rate of 0.1/s is shown in Fig. 3.9a. However, discretization error is still present, and it is difficult to properly reassign the cells as material or air, once the contour is determined. Fig. 3.9b shows the result when all cells that do not bound the 0.479 contour are reset as either 0.0 or 1.0.

Table 3.1 gives the number of cells removed for 2D isotropic etching from a seed point, with a simple diffusion-based etch model. The area is given in units of cells, and the CPU time in seconds is for an IBM Powerstation 530.

Table 3.1: No. Cells Removed with 2D Diffusion Model				
rate=5 cells/s time step	1.0 s r=5 cells	2.0 s r=10 cells	3.0 s r=15 cells	4.0 s r=20 cells
0.05s (CPU)	17 cells 0.91s	2301 cells 2.72s	N/A	N/A
0.04s (CPU)	17 cells 1.125s	81 cells 5.0s	177 cells 11.31s	317 cells 20.29s
0.02s (CPU)	17 cells 2.45s	81 cells 10.10s	177 cells 27.78s	317 cells 40.6s
0.01s (CPU)	17 cells 5.0s	81 cells 20.0s	177 cells 45.1s	317 cells 80.2s
theory	19.6 cells	78.5 cells	176.7 cells	314.2 cells

The CPU time is significant, even for a 50x50 cell 2D simulation. For a 50x50x50 3D simulation the CPU time would be increased at least 50 times, without even accounting for the longer time to calculate gradients and divergences in 3D! It should also be noted that the etch rates in the above expressions are independent of the concentration and thus of the surface orientation. This would not be the case for many direction dependent topography processes. Etch rates that are functions of surface orientation (as in ion milling) would complicate the problem significantly since equation 3.6 would no longer be valid.

The implementation of Fujinaga *et. al.* in 3D-MULSS achieves accuracy by solving the diffusion equation with a finite difference method and periodically reinitializing the simulation region by retaining only those cells needed to describe the surface and assigning everything else to be bulk or air. This method greatly improves the accuracy of the simulation but now the problem is expanded to a system of  $N$  linear equations, where  $N$  is the number of nodes in the simulation region. Even the basic forward finite-difference method requires the inversion of an  $N \times N$  matrix.<sup>8</sup> Similar variable-density volume-removal algorithms which do not solve the diffusion equation are not likely to improve on the results of Fujinaga *et. al.* The only exception would be cell-based simulator that represent material structures on the atomic level and which solve surface kinetics and transport phenomena explicitly.<sup>9, 10</sup>

### 3.4. CONCLUSIONS ON CELL-BASED SIMULATION

Basic cell-removal algorithms can be adequate for lithography development simulation. General topography simulation, however, requires the ability to relate the surface orientation to etching and deposition models. The network method provides surface orientation, but the problem of etching convex corners must still be addressed adequately. The diffusion equation method does not have a problem with convex corners, and the concentration gradient gives the surface normal. However, it requires the diffusion equation to be solved numerically in three dimensions and is difficult to apply to processes with etch rates dependent on surface orientation.

In summary, cell-based topography simulation is not the panacea that it might appear on first encounter. All of the above methods can be applied to process technology problems, but there is still room for investigating alternate approaches.

## REFERENCES

1. K.K.H. Toh, "Algorithms for Three-Dimensional Simulation of Photoresist Development," Memo. No. UCB/ERL M90/123, Ph.D. Dissertation, University of California, Berkeley, December 14, 1990.
2. W. Henke, D. Mewes, M. Weiss, G. Czech, and R. Schiessl-Hoyler, "Simulation of Defects in 3-Dimensional Resist Profiles in Optical Lithography," *Microelectronic Engineering*, vol. 13, pp. 497-501, 1991.
3. Y. Hirai, S. Tomida, K. Ikeda, M. Sasago, M. Endo, S. Hayama, and N. Nomura, "Three-Dimensional Resist Process Simulator PEACE (Photo and Electron Beam Lithography Analyzing Computer Engineering System)," *IEEE Transactions on Computer-Aided Design of Integrated Circuits*, vol. CAD-10, no. 6, pp. 802-807, June 1991.
4. K.K.H. Toh, "Two-Dimensional Images with Effects of Lens Aberrations in Optical Lithography," *M.S. Thesis, Memorandum No. UCB/ERL M88/30*, University of California, Berkeley, May 20, 1988.
5. T. Ishizuka, "Three-Dimensional Simulation in Photoresist Development," *IEICE (In Japanese)*, vol. J73-C-II, no. 11, pp. 775-785, November 1990.
6. M. Fujinaga, N. Kotani, T. Kunikiyo, H. Oda, M. Shirahata, and Y. Akasaka, "Three-Dimensional Topography Simulation Model : Etching and Lithography," *IEEE Transactions on Electron Devices*, vol. ED-37, no. 10, pp. 2183-2192, October 1990.
7. J. Crank, *The Mathematics of Diffusion*, p. 12, Clarendon Press, Oxford, UK, 1956.
8. E.B. Becker, G.F. Carey, and J.T. Oden, *Finite Elements, An Introduction*, I, p. 246, Prentice-Hall, Englewood Cliffs, New Jersey, 1981.



9. J. Pelka, "Simulation of Ion-Enhanced Dry-etch Processes," *Microelectronic Engineering*, vol. 13, pp. 487-491, 1991.
10. R.N. Tait, S.K. Dew, T. Smy, and M.J. Brett, "Density Variation of Tungsten Films Sputtered Over Topography," *Journal of Applied Physics (submitted)*, 1991.

## CHAPTER 4

### A GENERAL SURFACE ADVANCEMENT ALGORITHM FOR 3D TOPOGRAPHY SIMULATION

*"Either this madness or it is Hell."  
"It is neither," calmly replied the  
voice of the Sphere, "it is knowledge;  
it is Three Dimensions: open your eye  
once again and try to look steadily."  
-Flatland, Edwin A. Abbot*

#### 4.1. ALGORITHM DESIGN METHODOLOGY

Previous chapters have discussed the general topics of process simulation and modeling, have established the need for more work in 3D surface advancement algorithms, and have outlined a general design philosophy. This chapter focuses on the development, implementation and evaluation of a general surface advancement algorithm. First, discussions of the problem and of previous work in surface representations and mesh refinement begin the chapter. Next, geometric arguments provide a mathematical foundation for the correctness and accuracy of a surface evolution algorithm based on the motion of surface facets. The geometric arguments consider isotropic surface advancement, simple directional etching, and etching with strong dependence on surface orientation. Finally, several simulation examples establish the credibility of the algorithm and its implementation.

#### 4.2. GENERAL SURFACE ADVANCEMENT AND TIME DISCRETIZATION

The basic question for topography simulation is: given a surface  $S_t$ , a set of process conditions  $P$ , and a time of evolution  $\Delta t$ , what is the resulting surface  $S_{t+\Delta t}$ ? A mapping function  $f$  performs the transformation.

$$S_{t+\Delta t} = f(P, S, \Delta t) \quad (4.1)$$

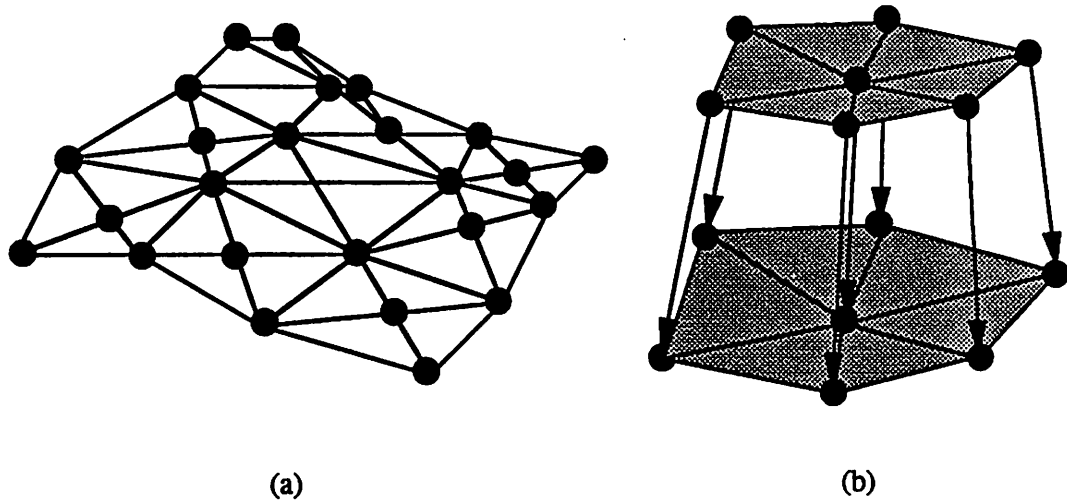
The mapping function often includes the shape of the surface which is continuously evolving over the time interval  $\Delta t$ . On a finite-state machine, the surface must be discretized into a finite number of geometric elements and time must be discretized into finite intervals of  $\Delta t$ . The discretization necessarily introduces an error term.

$$S_{t+\Delta t} = f(P, S(t), \Delta t) + \text{Error}(P, S_{t_0}, \Delta t) \quad (4.2)$$

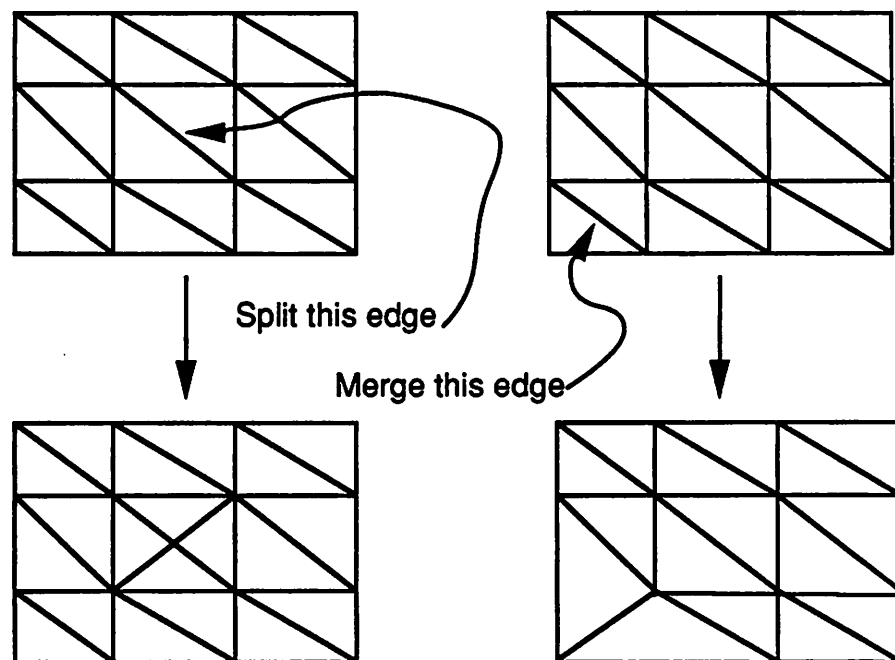
Fortunately, as will be demonstrated in this chapter, this error term can often be reduced to well within an acceptable limit by careful selection of  $\Delta t$  and the algorithm for implementing function  $f$ . Equation 4.2 also assumes that the process conditions  $P$  are independent of time. These conditions may actually vary, but this change is assumed small over  $\Delta t$ , as was discussed in chapter 2, section 2.2.

### 4.3. SURFACE REPRESENTATION

The way surfaces are represented has a major effect on the implementation of algorithms for topography simulation. There are many possible ways to represent curved surfaces,<sup>1</sup> but most describe a mesh of control points containing information about the location and curvature at several points on the surface. A simple, yet powerful, surface representation consists of a mesh of points, which are connected by line segments to form triangular facets as shown in Fig. 4.1a. The surface curvature is implicit in the orientation of the facets. Increasing the accuracy of the representation simply requires a greater density of surface points. The problem of advancing the surface is now one of how to advance each surface point as depicted in Fig. 4.1b.



**Fig. 4.1:** Surface representation. (a) Basic surface points, lines, triangles. (b) moving points.



**Fig. 4.2:** Mesh refinement by splitting or deleting triangle edges.

#### 4.4. SURFACE MESH REFINEMENT

As the surface evolves in time, its area increases or decreases. A method for adding and deleting surface points is a prerequisite for the correct operation of surface advancement algorithms. First, criteria for mesh refinement must be established: 1) What criteria must all valid surface meshes satisfy? 2) What criteria define an accurate mesh? 3) What is a "good" mesh for topography simulation?

The first question is easy to answer. All valid surface meshes must satisfy the following constraint: Exactly two surface triangles share one edge. Any mesh with more than two triangles sharing an edge no longer describes a valid surface. One triangle is allowed for the special case of edges that lie on the boundary of the simulation region.

Two criteria define an accurate mesh. First, the total volume between the surface mesh and the surface being represented is a minimum. Ideally, this volume is zero. Indeed, this is true for any surface comprised only of flat polygonal regions. For general curved surfaces this volume is not zero, but it is reduced by increasing the number of points representing the surface. Second, the orientation of the faces should be as close to the actual surface orientation as possible. Increasing the number of points and placing them correctly achieves this goal.

The most difficult concept to formalize is the idea of a "good" surface. Obviously, it must satisfy the criteria of validity and accuracy. For topography simulation a few other considerations make for a good mesh. The density of information must be reduced, so that the simulation run time and memory requirements are within practical limits. The shape of the triangles must also allow surface advancement algorithms to give correct results. The actual shape requirements are purposely vague at this point, but at the very least it should be possible to

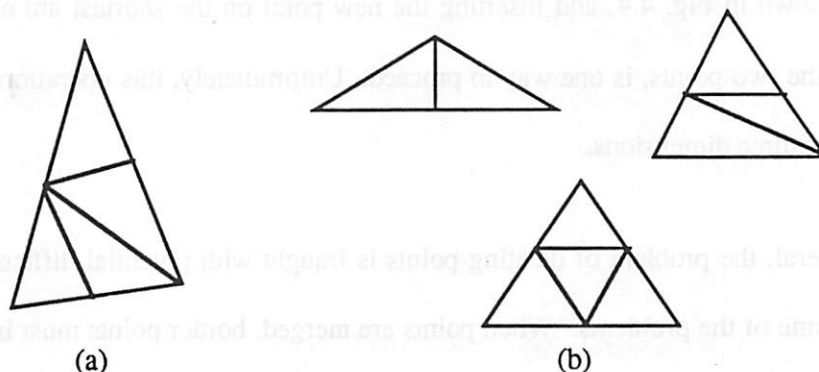
extract the normal vector of a triangle from knowledge of its vertexes. The restriction of near equilateral triangles often needed in finite element mesh generation is NOT established as a requirement at this stage.

Surface mesh refinement has been considered in lithography simulation by Moniwa *et al.*,<sup>2</sup> and further investigated by Toh.<sup>3</sup> These results apply to general topography simulation. The important results from this work and some additional observations are summarized here.

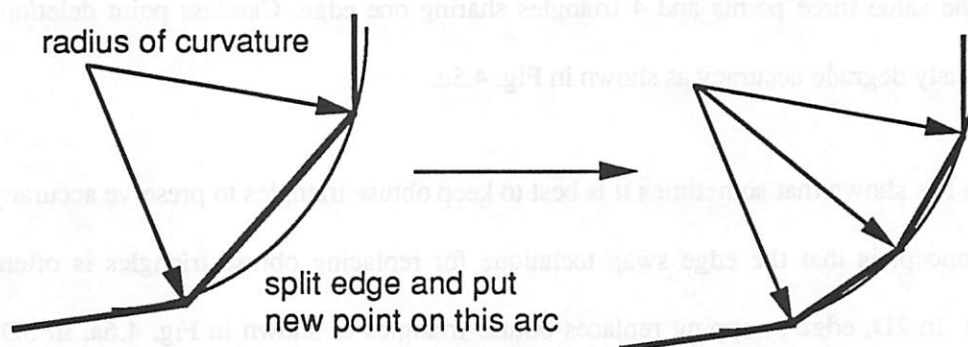
A convenient approach to point insertion and deletion is to test the length of all the triangle edges. All edges that are too long are split by the insertion of a point, and all edges that are too short are deleted by merging the end points as depicted in Fig. 4.2. The adjacent triangles are also either divided up or removed. Good results are obtained when the minimum allowed edge length is less than about 20% of some nominal ideal edge length. Maximum allowed edges are about 1.2 times longer than the ideal edge.

Satisfactory results can often be achieved by testing and splitting edges in random order, but this does not always split the triangles in the best way. It is better to test the triangles one by one. Toh chose to split the longest edge in each triangle first, and then proceed to shorter edges as shown in Fig. 4.3a. A more general approach is shown in Figure 4.3b, giving the 3 possible ways to split triangles. The last method is however more difficult to implement, since more information about the status of the triangles must be maintained.

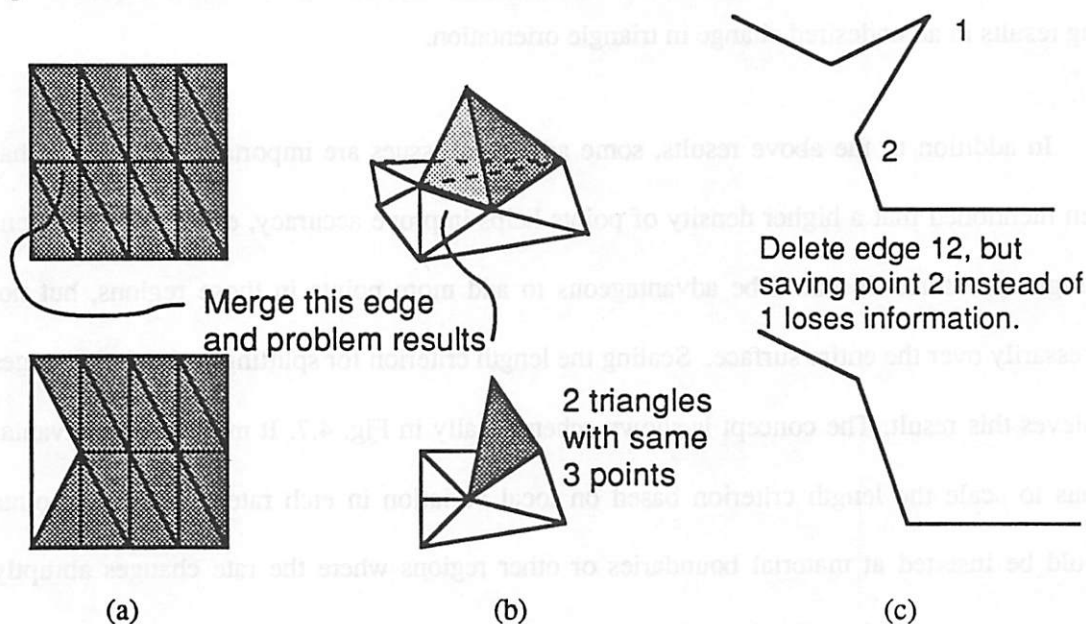
When inserting a mesh point, simply putting the new point on the original edge gives acceptable results if the density of mesh points is high enough. This method is also easy to implement. However, best accuracy is obtained if the new point is located as closely as possible to the surface being represented. Estimating the curvature of the surface between two



**Fig. 4.3:** Splitting triangles. (a) Splitting triangle edges in order from longest to shortest. (b) Dividing up triangle based on number of edges affected.



**Fig. 4.4:** Point insertion on the arc of curvature in two dimensions.



**Fig. 4.5:** Problems with point deletion and edge merging: (a) result of not maintaining border point. (b) Mesh leading to invalid surface. (c) Accuracy degradation.

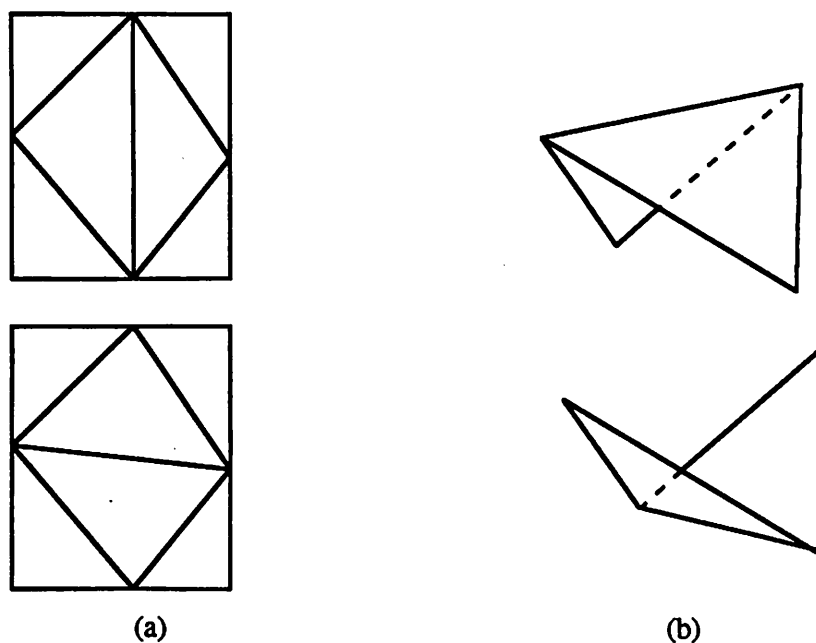
points, as shown in Fig. 4.4, and inserting the new point on the shortest arc on that surface connecting the two points, is one way to proceed. Unfortunately, this operation is difficult to implement in three dimensions.

In general, the problem of deleting points is fraught with potential difficulties. Fig. 4.5 illustrates some of the problems. When points are merged, border points must be maintained, or the surface will contract into the simulation region. For pyramid-like structures as in Fig. 4.5b, certain edges may not be deleted or an invalid surface will result, with two triangles sharing the same three points and 4 triangles sharing one edge. Careless point deletion can also seriously degrade accuracy as shown in Fig. 4.5c.

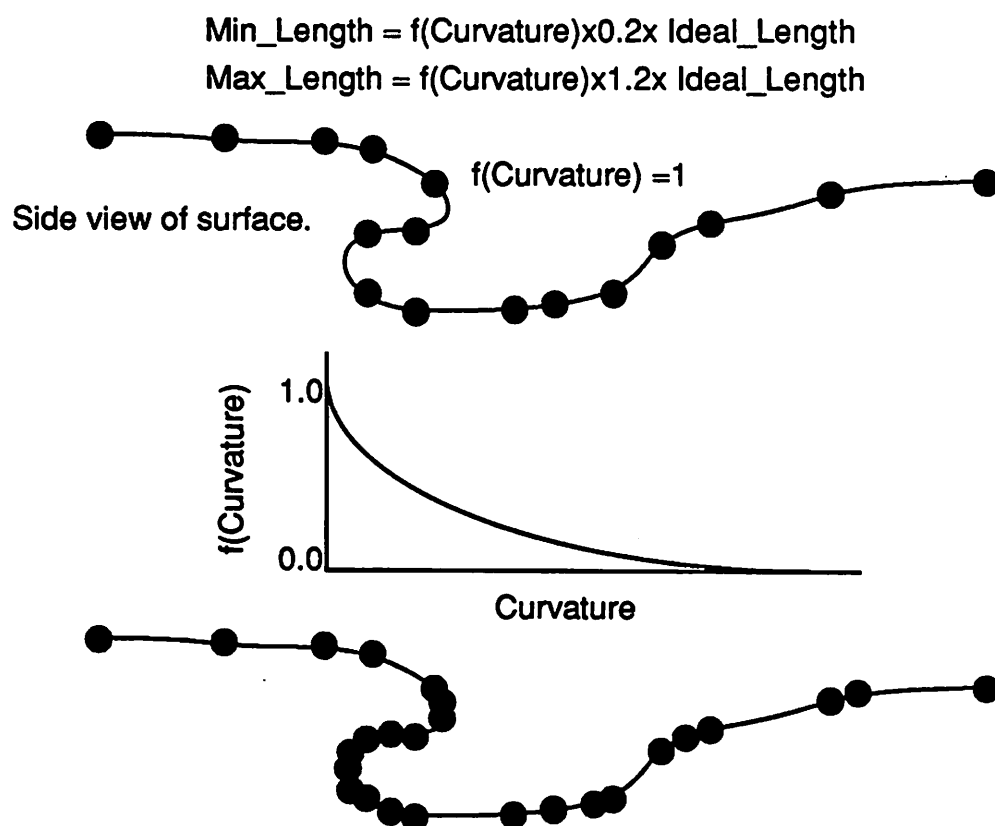
Toh has shown that sometimes it is best to keep obtuse triangles to preserve accuracy. A related concept is that the edge swap technique for replacing obtuse triangles is often not beneficial. In 2D, edge swapping replaces obtuse triangles as shown in Fig. 4.6a. In 3D this could result in an inaccurate surface representation. Fig. 4.6b shows a case where edge swapping results in an undesired change in triangle orientation.

In addition to the above results, some additional issues are important. Already, it has been mentioned that a higher density of points helps improve accuracy, especially in regions of high curvature. It would be advantageous to add more points in those regions, but not necessarily over the entire surface. Scaling the length criterion for splitting and deleting edges achieves this result. The concept is shown schematically in Fig. 4.7. It may also be advantageous to scale the length criterion based on local variation in etch rate. Above all, points should be inserted at material boundaries or other regions where the rate changes abruptly from zero to some value. Section 4.7 compares these approaches.





**Fig. 4.7:** Edge swapping to replace obtuse triangles. (a) 2D case. (b) Surface shape changes for edge swap in 3D.



**Fig. 4.7:** Adjusting the minimum and maximum allowed edge lengths according to local surface curvature.

## **4.5. MOVING NODES: THE FACET MOTION ALGORITHM**

### **4.5.1. The Simple Case: Moving Along Given Vectors.**

Each point on the surface moves along some vector to its location on the new surface. Many physical models give this vector directly. For example, that is the case for metal evaporation according to Blech's model.<sup>4</sup> That is also the method used by ray-trace lithography simulation in which the motion of the surface is calculated based on the spatial distribution of etch rates in the vicinity of each surface point. Models that advance the surface independent of the surface orientation are desirable since the effect of surface inaccuracy is reduced. A general surface motion algorithm should support the degenerate case of moving nodes along given directions.

### **4.5.2. Surface Normal Dependence: The 2D Segment Method**

Many process models cannot be implemented in such a straightforward way as the ray-trace method for isotropic etching. For example, the ray-trace equation cannot be applied to topography simulation models that strongly depend on the shape of the surface, since the 3D spatial distribution of etch rates is not known. Many physical topography evolution models have their foundation in the following hypothesis: Surfaces advance along the surface normal at a velocity related to the orientation of the surface. This approach to modeling has been used for 2D simulation of many etching and deposition processes.<sup>5, 6, 7, 8, 9, 10</sup> Some models also move the surface points along a specific direction that is related to the surface normal, as is the case in some models for columnar film growth.<sup>11</sup> This approach to surface advancement is quite general and can be easily implemented in 2D as string or segment advancement algorithms. The segment concept is illustrated in Fig. 4.8. The end points of segments advance

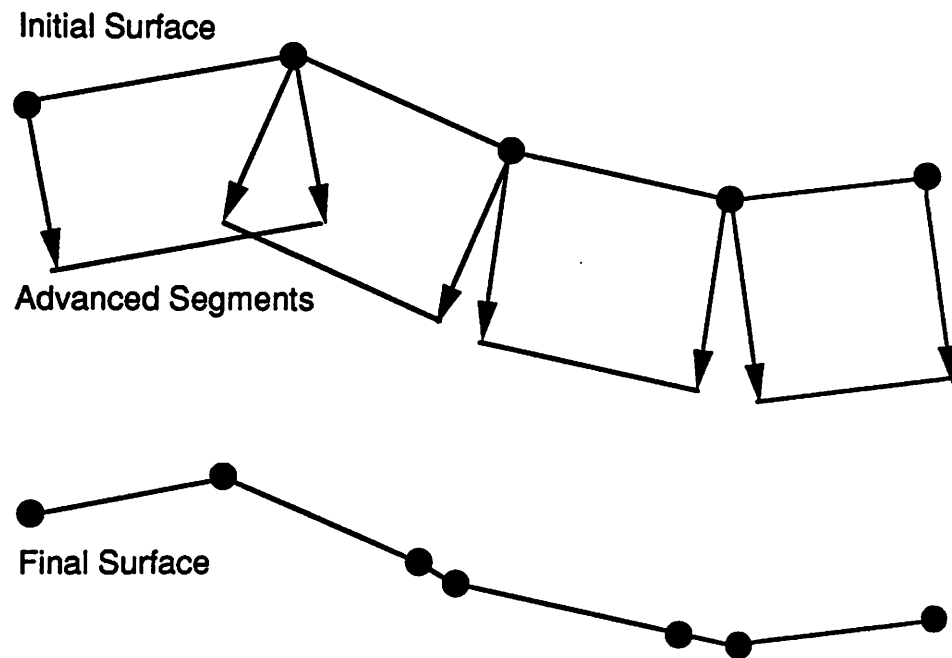
according to the local rate and the orientation of the segments. As segments pull apart, new segments are inserted. As segments collide, the intersection is determined and used as the new point joining the two segments.

Thurgate has refined segment based etch modeling with point allocation based on surface curvature, accurate handling of material and shadow boundaries, and full consideration of angle dependent effects.<sup>12</sup> Another method by Tazawa *et. al.* also achieves a high degree of accuracy by carefully considering varying etch rates along different directions. This method calculates the distance traveled by all the virtual planes at a surface corner. The actual point advancement vector ends at the intersection of the surface plane and the bisector of the angle created by the two limiting planes that is closest to the original point, as shown in Fig. 4.9.<sup>10</sup>

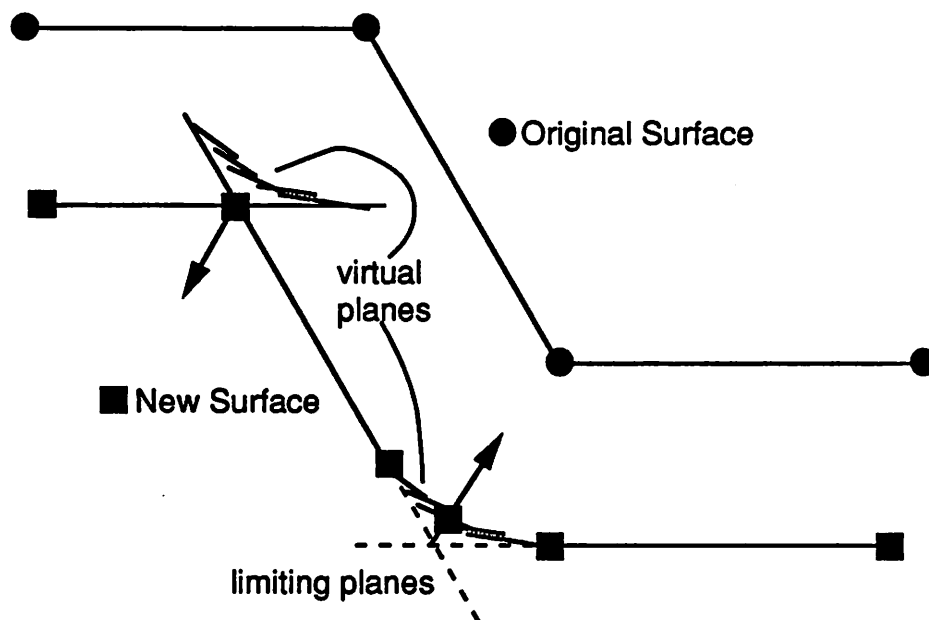
#### 4.5.3. From 2D Segments to 3D Facets

The 2D segment advancement concept satisfies the needs of general 2D topography simulation. Such an approach would also satisfy the needs of general 3D topography simulation if a method exists to advance a surface based on the motion of its constituent triangular facets. The motion of the facets themselves is not difficult. The three facet vertexes advance along given directions (typically along the surface normal) according to the local rate and the surface orientation, as shown in Fig. 4.10. As long as the time step is small enough, the algorithm handles even spatially varying rates, although a correction factor might allow for a larger time step.

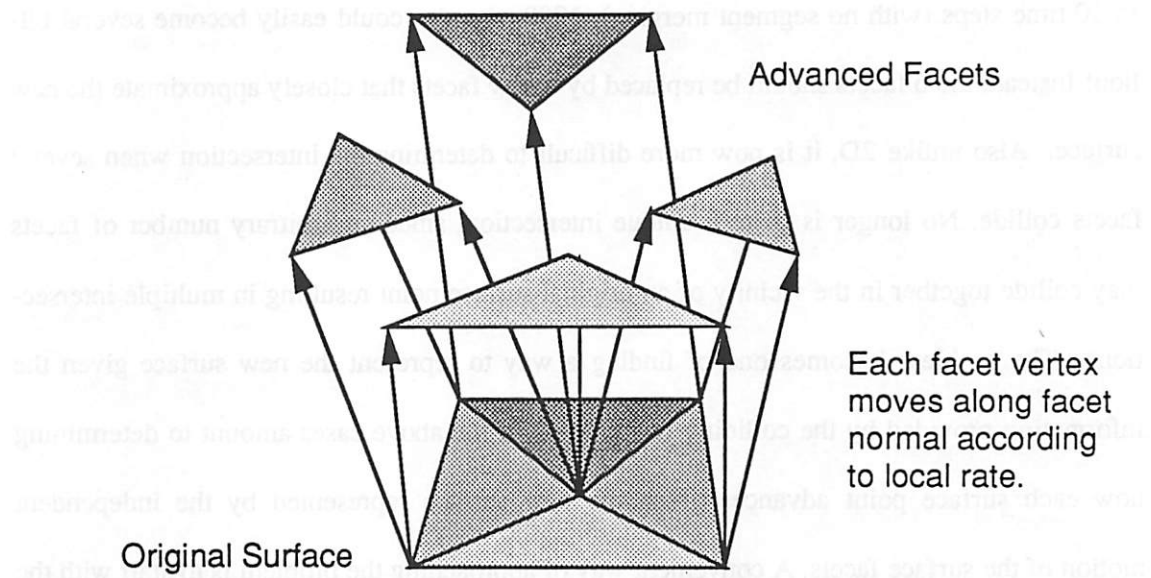
The difficulty comes in representing the new surface given the information present in the facets. Unlike 2D segment algorithms, in 3D it is impractical to add facets everywhere they pull apart. In Fig. 4.11, pulling apart 6 facets requires the insertion of 18 new facets. Such a



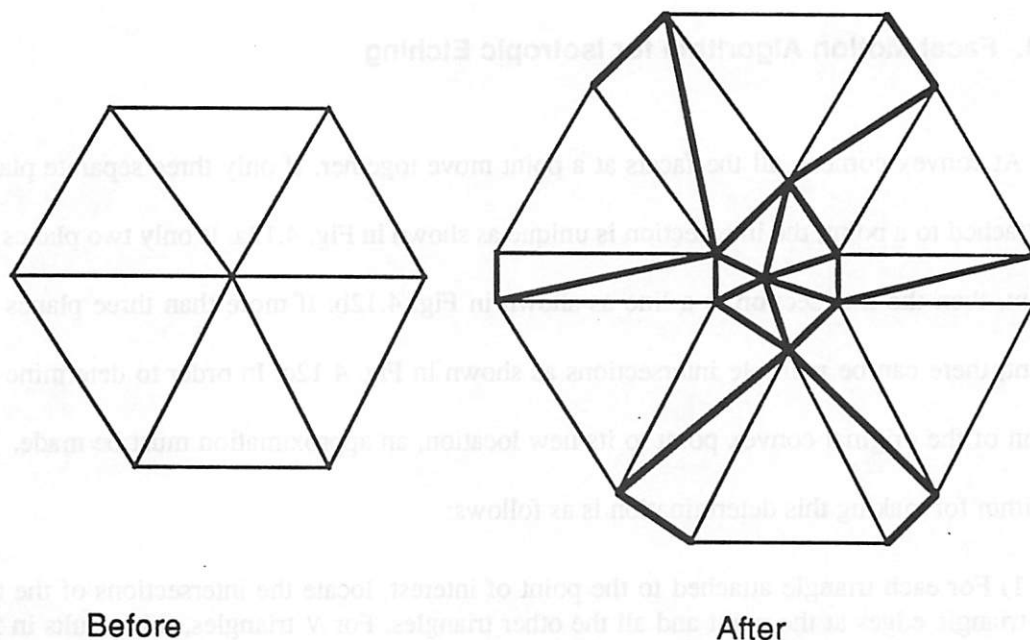
**Fig. 4.8:** The basic 2D segment advancement algorithm.



**Fig. 4.9:** Segment advancement algorithm according to Tazawa *et al.*<sup>10</sup>



**Fig. 4.10:** Advancing surface facets according to local rate and surface orientation.



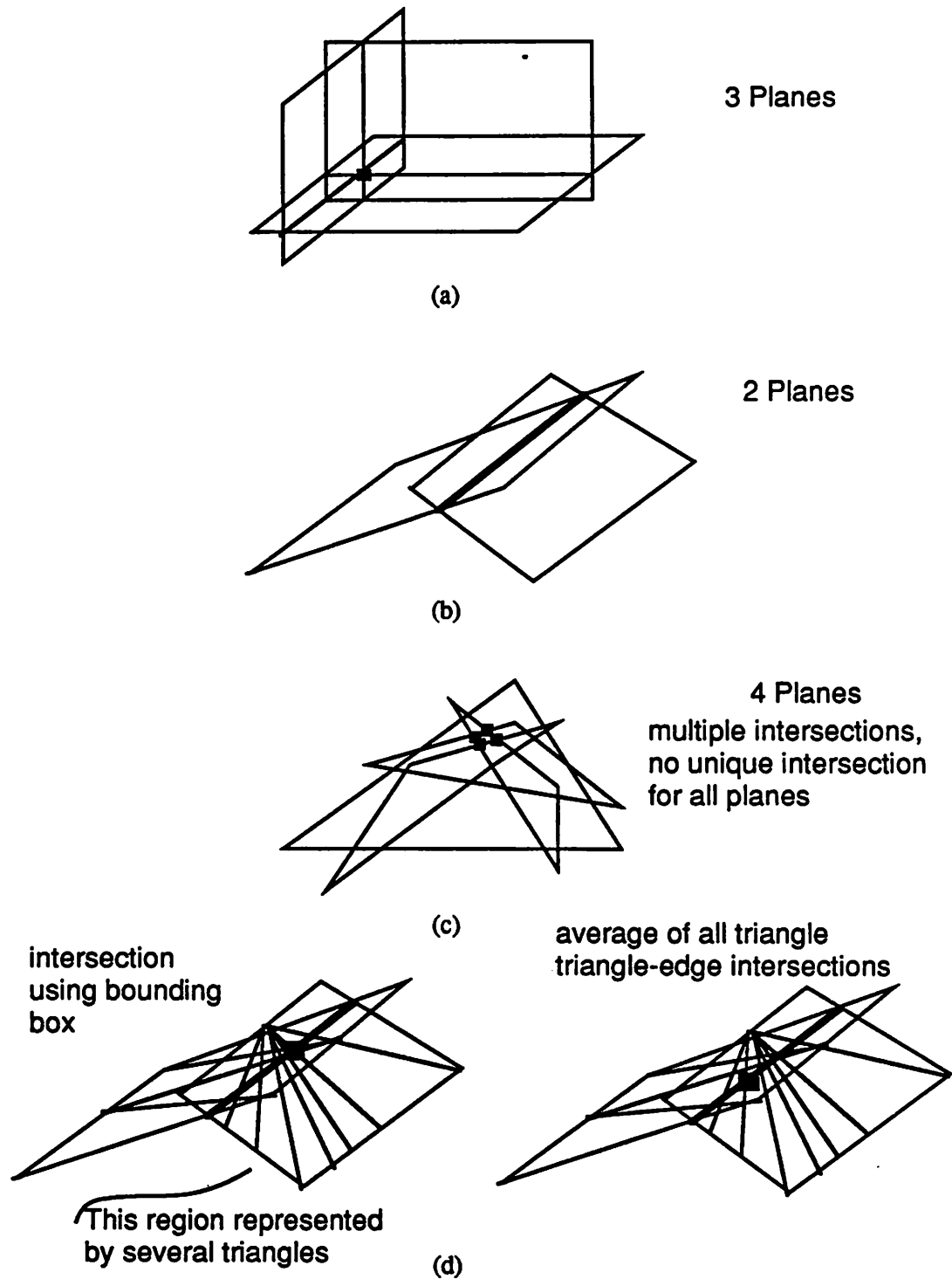
**Fig. 4.11:** Why the 2D concept of filling in segment gaps is difficult to implement in 3D.

geometric growth in the number of surface triangles would create havoc on most computers. In 10 time steps (with no segment merging), 1000 triangles could easily become several billion! Instead, the 6 facets should be replaced by 6 new facets that closely approximate the new surface. Also unlike 2D, it is now more difficult to determine the intersection when several facets collide. No longer is there a unique intersection, since an arbitrary number of facets may collide together in the vicinity of an original surface point resulting in multiple intersections. The problem becomes one of finding a way to represent the new surface given the information provided by the colliding faces. Both of the above cases amount to determining how each surface point advances given the new surface represented by the independent motion of the surface facets. A convenient way of approaching the problem is to start with the simple case of isotropic etching and then proceed to more difficult directionally dependent processes.

#### **4.5.4. Facet Motion Algorithm for Isotropic Etching**

At convex corners, all the facets at a point move together. If only three separate planes are attached to a point, the intersection is unique as shown in Fig. 4.12a. If only two planes are present, then the intersection is a line as shown in Fig. 4.12b. If more than three planes are present, there can be multiple intersections as shown in Fig. 4.12c. In order to determine the motion of the original convex point to its new location, an approximation must be made. An algorithm for making this determination is as follows:

- 1) For each triangle attached to the point of interest, locate the intersections of the two triangle edges at the point and all the other triangles. For  $N$  triangles, this results in  $2 \cdot N$  intersection tests.
- 2) Determine the box bounding the intersections by searching for the minimum and maximum  $x, y$ , and  $z$  coordinates of all the intersection points.



**Fig. 4.12:** Facet motion at convex corners (isotropic): (a) Three planes at a node. (b) Two planes at a node. (c) Multiple planes at a node. (d) Multiple triangles at a plane: result of averaging intersections vs. locating center of box bounding intersections.

- 3) Take the center of the box as the new point location.

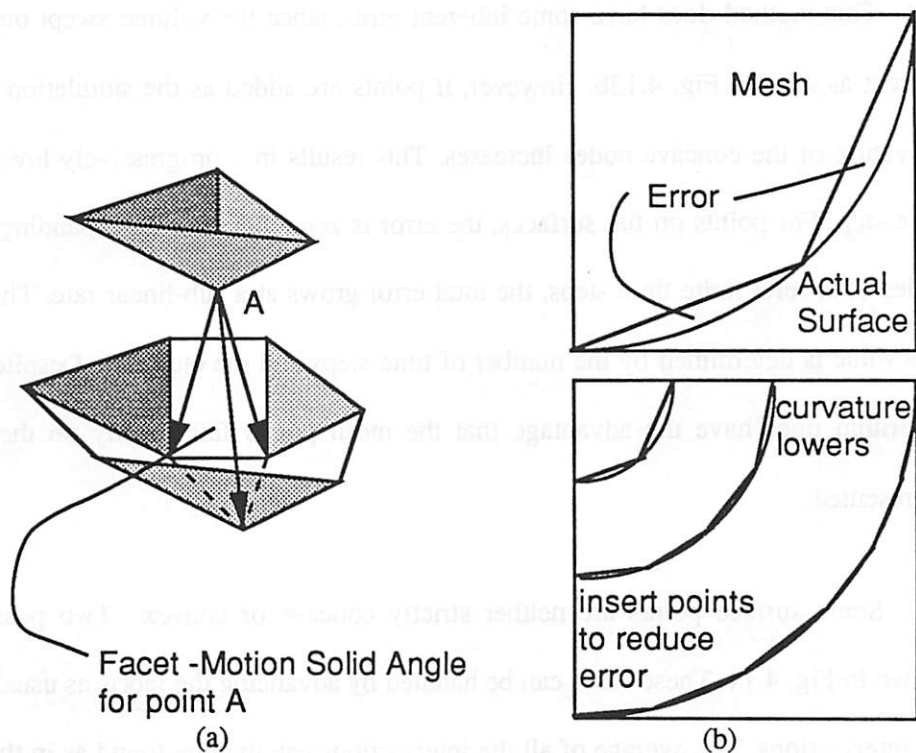
This algorithm finds the exact intersection point for two and three planes connected at a convex point. It also finds an approximate intersection that is the average of all possible intersections, but is not affected by cases when more than one triangle facet shares a plane. This is important as illustrated in Fig. 4.12d. One of the planes at the convex point is represented by several facets. Simply taking the average would result in a new point shifted too far towards the plane with multiple triangles. Taking the average of the bounding box gives the same result as the case when each plane is represented by only one triangle.

At concave corners, the facets at a point move away from each other. For uniform isotropic etching, the distance traveled is the same for all facets. The vectors from the starting point to the corresponding points on the expanded facets trace out the border of a solid angle (which will henceforth be called the *facet-motion solid angle*). In the isotropic case, all possible vectors within this solid angle trace out the surface of part of a sphere, bordered by the solid angle depicted in Fig. 4.13a. The problem of determining where to move the original point, is now one of where to locate the new point on the surface of the sphere bounded by the facet-motion solid angle. An algorithm for this is as follows:

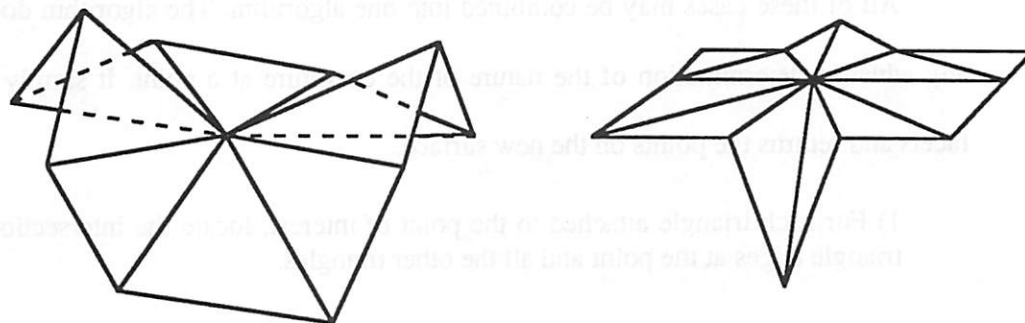
- 1) Determine the box bounding all the endpoints of the vectors to the new facets.
- 2) Determine the vector from the original point through the centroid of the box.
- 3) Advance the original point along the vector through the centroid by a distance equal to the distances of all the other vectors emanating from the original point.

Since the facets all move equal distances, the center of the bounding box is almost exactly the same as the centroid of the convex hull of the facet points. Moving along the vector through the centroid provides a very good approximation of the new point location.





**Fig. 4.13:** Facet motion at convex corners (isotropic): (a) The facet motion solid angle. (b) Volume error : comparing actual surface with mesh (2D view).



**Fig. 4.14:** Two cases that are neither concave nor convex.

This method does have some inherent error, since the volume swept out is not exactly correct as seen in Fig. 4.13b. However, if points are added as the simulation progresses, the curvature of the concave nodes increases. This results in a progressively lower error at each time step. For points on flat surfaces, the error is zero. Thus for an expanding surface over a series of several finite time steps, the total error grows at a sub-linear rate. The magnitude of this value is determined by the number of time steps and the etch rate. Despite this error, the algorithm does have the advantage that the mesh points fall exactly on the surface being represented.

Some surface points are neither strictly concave or convex. Two possible cases are shown in Fig. 4.14. These cases can be handled by advancing the facets as usual and searching for intersections. The average of all the intersections can then be found as in the case for convex nodes. Next, the box bounding all the relevant points on non-intersecting facet edges is found. If the average intersection is inside this box, the point is treated as a concave point. If the average intersection is outside the box, it is taken as the new point.

All of these cases may be combined into one algorithm. The algorithm does not require any advance determination of the nature of the curvature at a point. It simply advances the facets and returns the points on the new surface.

- 1) For each triangle attached to the point of interest, locate the intersections of the two triangle edges at the point and all the other triangles.
- 2) Determine the box bounding the intersections by searching for the minimum and maximum x,y, and z coordinates of all the intersection points. If no intersections exist, skip this step.
- 3) If there are line segments that do not intersect with any of the triangles, make a second box bounding all the endpoints of those vectors to the new facets.

- 4) If the center of the first box is outside of the second box, take the center to be the new point location. Otherwise, determine the center of the second box, and find the new point by advancing the original point through the center of the second box according to the local rate.

This combined algorithm does not determine the concavity or convexity of a point in advance, thereby reducing one possible source of numerical error. Treating flat points as a special case and avoiding the need to calculate intersections improves the algorithm speed. The algorithm handles topography simulation with spatially inhomogeneous rates, since the rates for all the facets at one point are the same in all directions. It should be noted that if a correction factor is used, which alters the distance traveled according to the etch rate variation along different directions, the new facets share many aspects in common with directional etching discussed in the next sections.

#### 4.5.5. Facet Motion Algorithm for Simple Directional Process Models

Simple directional processes are modeled by adding the isotropic rate to the directional rate projected onto the facet normal, as shown in Fig. 4.15. The rate of point movement along a facet normal is expressed as

$$rate_{facet-normal} = \begin{cases} rate_i + rate_d \cos(\alpha) & 0 \leq \alpha \leq \pi/2 \\ 0 & \alpha > \pi/2 \end{cases} \quad (4.3)$$

where  $r_i$  and  $r_d$  are the isotropic and directional rates, and  $\alpha$  is the angle between the facet normal and a predetermined etch direction. For  $\alpha > \pi/2$  the etch rate is 0 since the facet shadows itself in that case. Facets are moved as in the isotropic case, except different facets at a point now move different distances.

Convex nodes can be handled exactly as in the isotropic case. Proving that the distance to the intersection is greater than the maximum distance that could be traveled along any

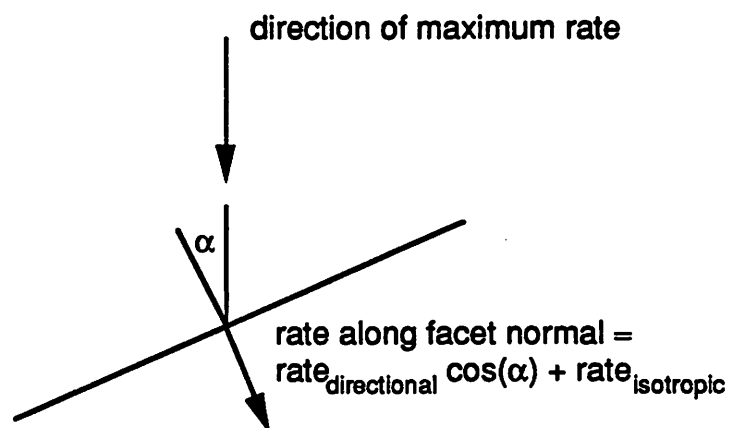


Fig. 4.15: Simple directional etching using isotropic and directional rate components.

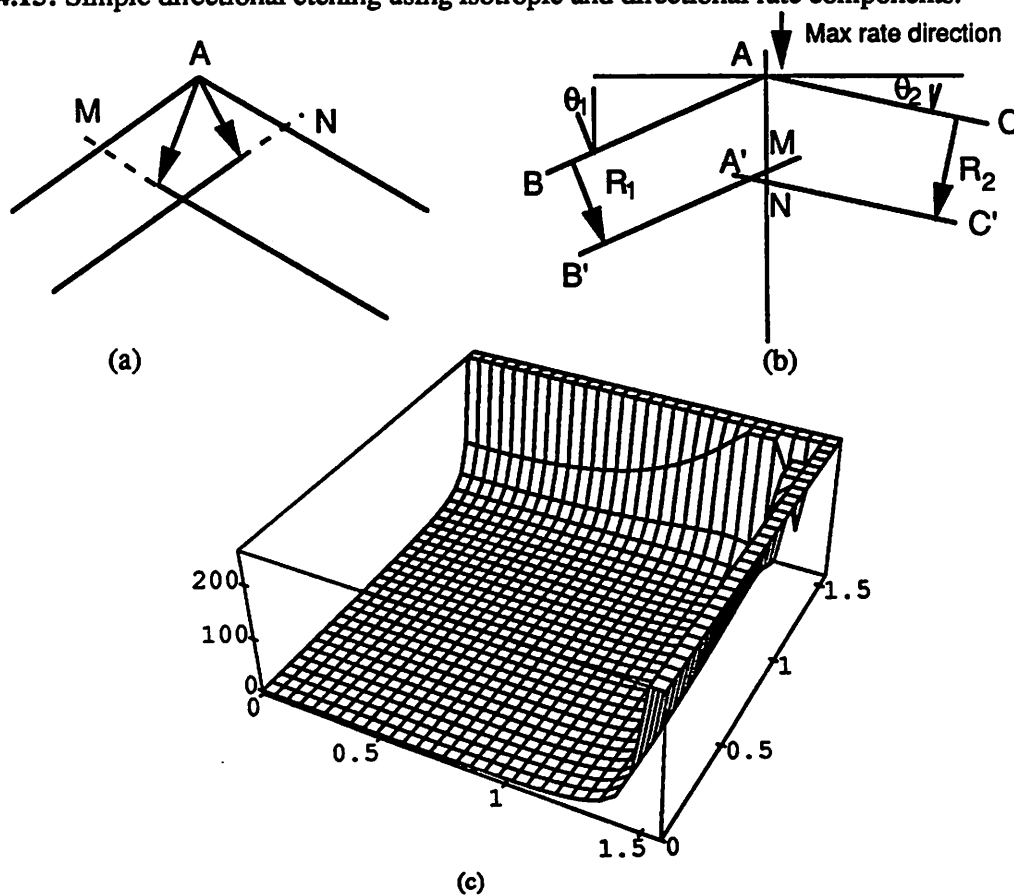


Fig. 4.16: Convex node in simple directional etching (2D): (a) Fastest direction not in angle. (b) General 2D case for proof of equivalence with isotropic algorithm. (c) Plot proving statement (4.5) for  $\text{rate}_d = \text{rate}_i$ .

direction within the facet-motion solid angle verifies the previous statement. It is sufficient to prove this in two dimensions. This can be seen by imagining several planes with an axis on the line from the original point to the new intersection. For one such plane, it is possible to project the etch direction, and the surface normals of the intersecting facets onto that plane so that the facet motion problem is reduced to 2D. The entire 3D case is represented by the facet motion on all of the 2D planes. A proof need only exist for one of the 2D planes since it holds for all planes, and thus by extension to the 3D case. The facet-motion solid angle must also contain the direction where  $\alpha=0$ , otherwise the the fastest direction is along the border of the solid angle. The intersection when the fastest direction is not in the solid angle is always farther way from the original point than any point on the facet-motion solid angle boundary. This is shown in Fig. 4.16a for the 2D case where  $|\vec{AN}| + |\vec{AM}| \geq |\vec{V}_{\max}|$ .

Fig. 4.16b is a diagram of the 2D geometry for the general proof. The proof shows that the distance from  $A$  to  $A'$  is greater than the maximum possible distance  $R_{\max} = rate_i + rate_d$ . In the given coordinate system, both  $\overline{BA}$  and  $\overline{B'A'}$  have a slope of  $\cot(\theta_1)$  and both  $\overline{AC}$  and  $\overline{A'C'}$  have a slope of  $-\cot(\theta_2)$ . The y-intercepts  $N$  and  $M$  are also easy to find:  $M_y = -R_1 \cdot \sec(\theta_1)$  and  $N_y = -R_2 \cdot \sec(\theta_2)$  where  $R_1 = rate_i + rate_d \cdot \cos(\theta_1)$ . The coordinates of the intersection  $A'$  are found by solving for the intersection of the lines  $\overline{NC'}$  and  $\overline{B'M}$ :

$$\begin{aligned} x &= \frac{R_1 \cdot \sec(\theta_1) - R_2 \cdot \sec(\theta_2)}{\cot(\theta_1) + \cot(\theta_2)} \\ y &= \cot(\theta_1) \cdot \frac{R_1 \cdot \sec(\theta_1) - R_2 \cdot \sec(\theta_2)}{\cot(\theta_1) + \cot(\theta_2)} - R_1 \cdot \sec(\theta_1) \end{aligned} \quad (4.4)$$

The proof now reduces to a verification of the following:

$$x^2 + y^2 \geq R_{\max}^2 \quad (4.5)$$

The algebra to prove this statement is quite tedious, however it is not difficult for a computer to plot  $x^2 + y^2 - R_{\max}^2$  for  $0 < \theta_1 < \pi/2, 0 < \theta_2 < \pi/2$ . If this quantity is greater than zero for  $rate_i \leq rate_d, rate_i = rate_d$ , and  $rate_i \geq rate_d$ , then statement 4.5 is proven and simple directional

etching at convex corners is exactly the problem of locating the intersection of the colliding facets. Fig. 4.16c is a plot generated by Mathematica<sup>TM 13</sup> for  $rate_i = rate_d$  showing that statement 4.5 is indeed true for that case. Similar plots result for  $rate_i \leq rate_d$  and  $rate_i \geq rate_d$ . †

Concave nodes are a different matter. Since, the different facets move at different rates, motion through the centroid of the bounding box as in isotropic etching no longer gives a good approximation. The reason for this is seen in Fig. 4.17 which compares 2D isotropic and directional etching. In the directional case, the orientation of the faster moving facet changes significantly. Since the etch rate is direction dependent, the inaccuracy in surface orientation has an adverse cumulative effect over several time steps.

Another problem occurs when the angle  $\alpha$  approaches  $\pi/2$ . Applying the isotropic etch algorithm to directional etching in Fig 4.18 results in a kink in the surface. This kink is incorrect since the angle between facet  $\overline{A'C'}$  and the etch direction is greater than  $\pi/2$ , and thus that facet should not have moved faster than the perpendicular facet  $\overline{AB}$ . The correct new location for A is the intersection of the advancing plane  $\overline{AB}$  with the moving facet  $\overline{AC}$ .

Two simple modifications to the algorithm for isotropic concave points provide a considerable degree of improvement. Instead of moving through the centroid of the box bounding all the ends of the vectors to the new facets, determine first which facet vector is longest and move the the original point along that vector. This has the effect of preserving the accuracy of the fastest moving plane. Second, any point which is attached to an immobile facet should not be moved. If the isotropic rate is zero, this prevents erroneous kinks from occurring. If the isotropic rate is nonzero, this compensates for kinks if they occur. The cumulative effect of

---

† Mathematicians might object to the use of the term "proof" here. The above analysis is more correctly a "verification". Purists might also object to the use of a computer to do the verification. However, the analysis is entirely reproducible and anyone wishing to disprove statement 4.5 is welcome to redo the analysis.

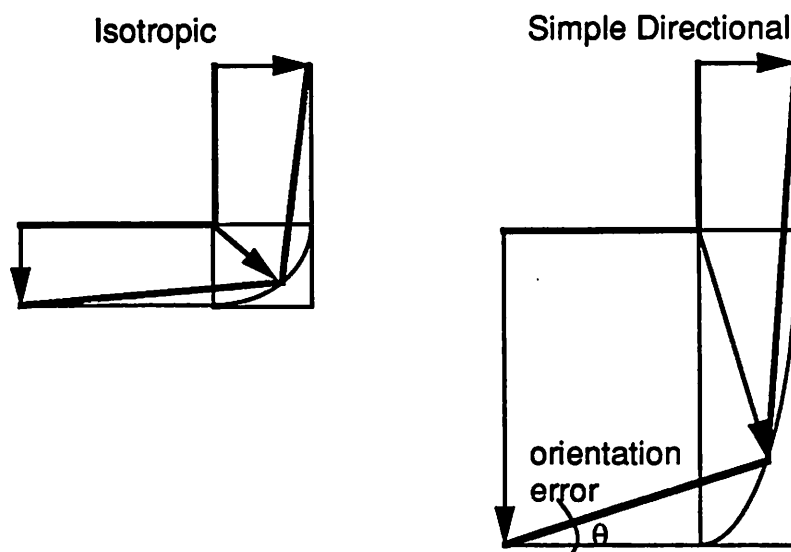


Fig. 4.17: 2D isotropic vs. simple directional etching using algorithm of section 4.5.4.

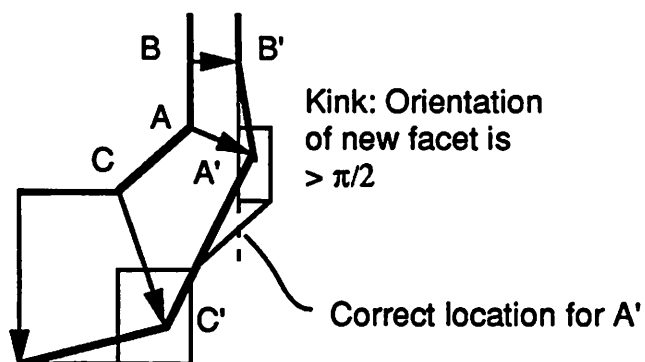


Fig. 4.18: Kink formation with isotropic etch algorithm for simple directional etching.

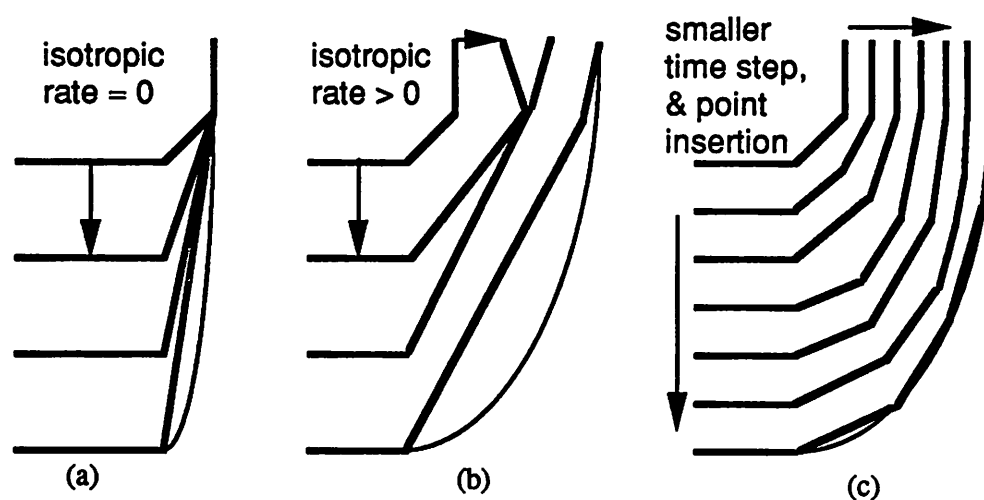


Fig. 4.19: Modifications to isotropic algorithm for simple directional etching. (a) cumulative effect, isotropic rate = 0. (b) isotropic rate > 0. (c) including point insertion.

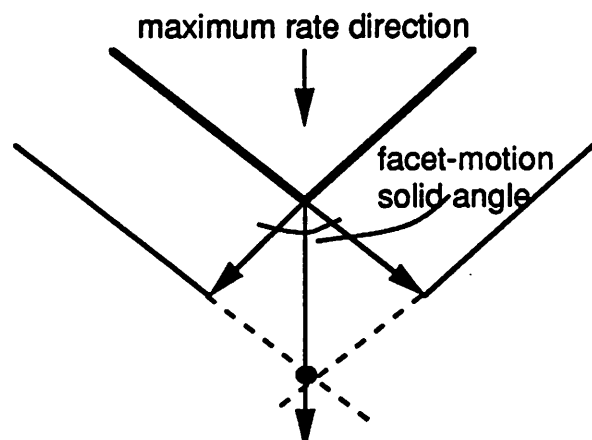
these two modifications is illustrated in 2D in Fig. 4.19a and b. Reducing the time step and regularly inserting points results in an even better approximate surface as shown in Fig. 4.19c.

A special case occurs when the facets all move different distances but the maximum possible rate in the facet-motion solid angle is not along one of the bounding vectors. This case is shown in 2D in Fig. 4.20. Here the new point is the intersection of the fastest direction with the first facet it hits. The fastest direction is always further than any of the planes, but the surface does not grow beyond the expanded facets.

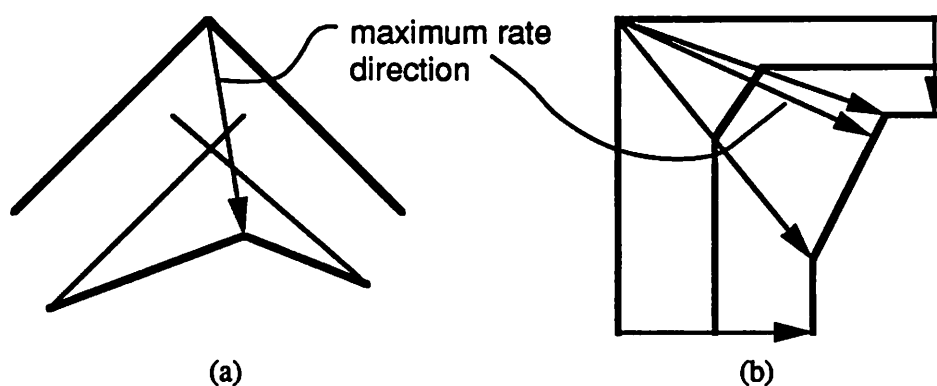
This algorithm can be combined with the isotropic algorithm in such a way that the algorithm is independent of the etch model.

- 1) For each triangle attached to the point of interest, locate the intersections of the two triangle edges at the point and all the other triangles.
- 2) Determine the box bounding the intersections by searching for the minimum and maximum x,y, and z coordinates of all the intersection points. If no intersections exist, skip this step.
- 3) If there are line segments that do not intersect with any of the triangles, make a second box bounding bounding all the endpoints of those vectors at the new facets.
- 4) If any facets attached to the point are immobile, do not move the point. Stop.
- 5) If the center of the first box is outside of the second box, take the center to be the new point location. Stop.
- 6) If the length ratio of the longest possible vector within the facet-motion solid angle to the the shortest vector traveled is greater than some predetermined factor (greater than 1), take the intersection of the longest vector with the first facet it hits as the new point location. Stop.
- 7) Determine the center of the second box, and find the new point by advancing the original point through the center of the second box according to the local rate along that direction.

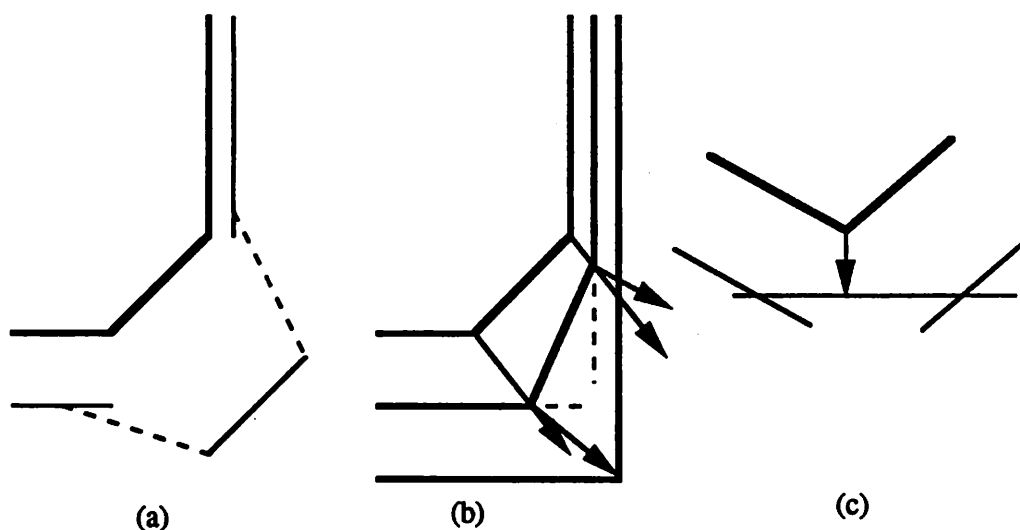




**Fig. 4.20:** Fastest direction inside facet-motion solid angle for concave point.



**Fig. 4.21:** Convex point in general directional etching (a) Fastest direction may be farther away than intersection points. (b) New facet formation.



**Fig 4.22:** Facets at concave corners. (a) Dotted line shows incorrect surface (b) Intersection of fastest direction with slowest plane to give correct result. (c) Using slowest direction in facet-motion solid angle.

This algorithm works for isotropic and simple directional etching with uniform or inhomogeneous etch rates.

#### 4.5.6. Facet Motion Algorithm for Strongly Direction Dependent Processes

The above algorithm relied on the cosine dependence of the etch rates along different directions. However, many etch models vary greatly with surface orientation. Ion milling and crystal etching are two highly anisotropic processes that cannot be simulated with the above algorithm. One important consideration is the formation of facets and edges from initially smooth surfaces.<sup>5, 14</sup> For convex points, it is no longer always true that the distance to the intersection of the colliding facets is greater than the maximum possible distance for all directions in the facet-motion solid angle as shown in Fig. 4.21. A test for the maximum etch direction in the facet-motion solid angle modifies step 5 for this case.

5) If the center of the first box is outside of the second box, test to see if any of the directions in the facet-solid angle result in a longer vector than the vector from the original point to the center of the first box. The new point location is the end of the longer of the two vectors. Stop.

The change correctly accounts for the formation of facets at convex surface points.

At concave surfaces, facets may collapse into edges as shown in Fig. 4.22a. The algorithm thus far gives the dotted line in Fig. 4.22a as the new surface. The result is incorrect since concave points would not become convex in the limit of an infinitesimally small time step. If the slowest moving direction is contained in the facet motion solid angle, that direction should be used. Otherwise, the intersection of the fastest moving direction with the plane containing the slowest moving facet, gives the correct result, as shown in Fig. 4.22b.

The complete algorithm, including a test for points on a flat surface, can be streamlined somewhat as follows:

- 1) If all facets attached to the point are immobile, do not move the point. Stop.
- 2) If all the normal vectors are within a small fraction of a steradian, move along the approximate normal vector of the surface. Stop.
- 3) For each triangle attached to the point of interest, locate the intersections of the two triangle edges at the point and all the other triangles. Also determine the fastest direction within the facet-motion solid angle.
- 4) If the point is convex, and the fastest direction is inside, move along the fastest direction. If the fastest direction is outside, move to the center of the box bounding all the line- triangle intersections. If no intersections exist, continue as if the point were concave.
- 5) Locate the slowest direction. If the slowest direction is inside the facet-motion solid angle, move along that direction. Otherwise, move to the intersection of the plane containing the slowest facet and the fastest direction in the facet-motion solid angle. If no intersection exist, move along the slowest direction. (If any facet is immobile, the point does not move.)

The above algorithm works for all types of etching and may be tailored for specific etch models to improve accuracy and performance. An explicit convexity test is made in step 5, since it is now possible for a convex node to possess no intersections due to the direction dependence.

#### **4.5.7. Border Points**

Points on the simulation border should remain on the simulation border. Their motion should be consistent with the evolution of points on 2D strings. This can be achieved by projecting the point motion according to the facets back onto the plane bounding the simulation region. The border point is then restricted to motion on the plane. For points that expand beyond the simulation region, it is also possible to determine the intersection of the surface

with the border plane, and take those points as the new border points.

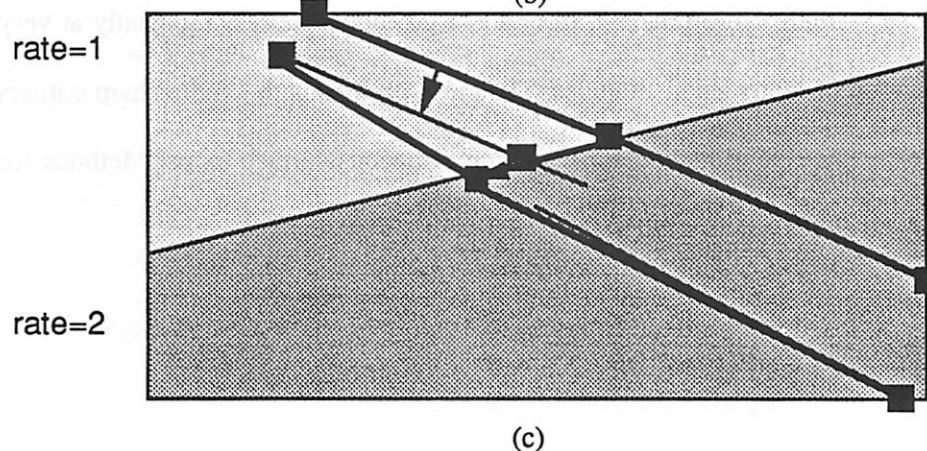
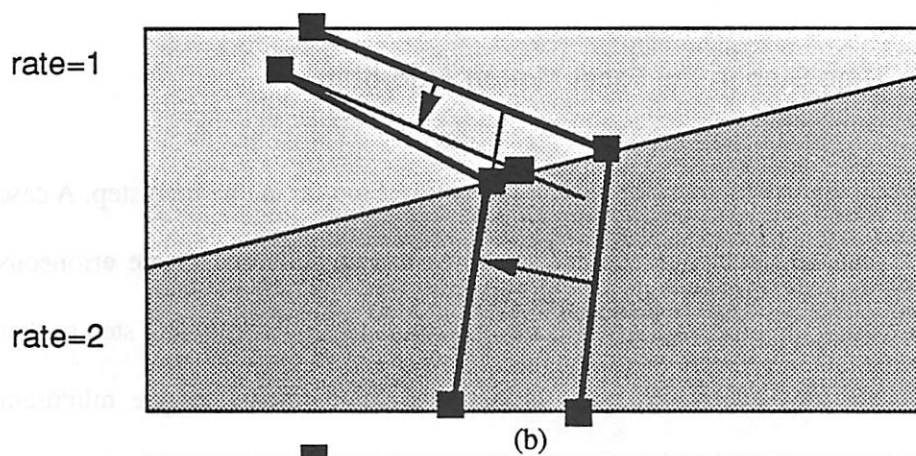
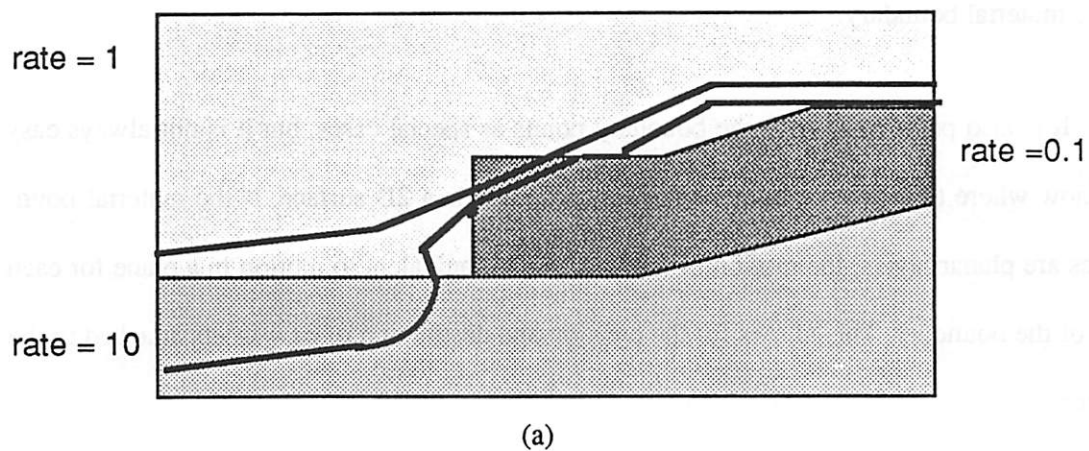
#### 4.5.8. Material Boundaries

The spatially inhomogeneous etch rates discussed thus far have not contained abrupt discontinuities. At material boundaries however, an abrupt change in etch rate is likely. The important question is how the surface evolves at the boundary.

As a first-order consideration, tests must be made to determine when a surface passes from one material to another. Since the etch rate may vary by a factor of 100 or more across the boundary, a calculation should be made to determine what fraction of the time step is spent before the border is reached. The distance traveled in the remaining part of the time step depends on the etch rate of the material past the boundary. This is shown schematically in Fig. 4.23a. In one instance, the etch rate is faster across the boundary, in the other the rate is slower.

Another problem is the motion of the surface at the material boundary. Thurgate has investigated this problem in 2D etch simulation.<sup>12</sup> One observation is that no segment may span a material boundary. Instead, a new node should be inserted at the material boundary. When the segment for either material collides with the boundary, the new position is the intersection of the segment and the boundary. This operation may be performed separately for each segment as shown in Fig. 4.23b. If the segment moves away from the boundary, as in Fig. 4.23c, The etch rate for each segment is projected onto the material boundary. The boundary node moves along the fastest etch vector.

The 3D facet-motion algorithm can handle multiple materials without any special con-



**Fig. 4.23:** Surface at material boundary. (a) Etch rate change at boundary. (b) Segment moving toward boundary. (c) segment moving away from boundary.

sideration, if nodes are added at material boundaries. The facets move at different rates just as in the case of direction dependent etching. At the end of the time step, new nodes are inserted at the material boundary.

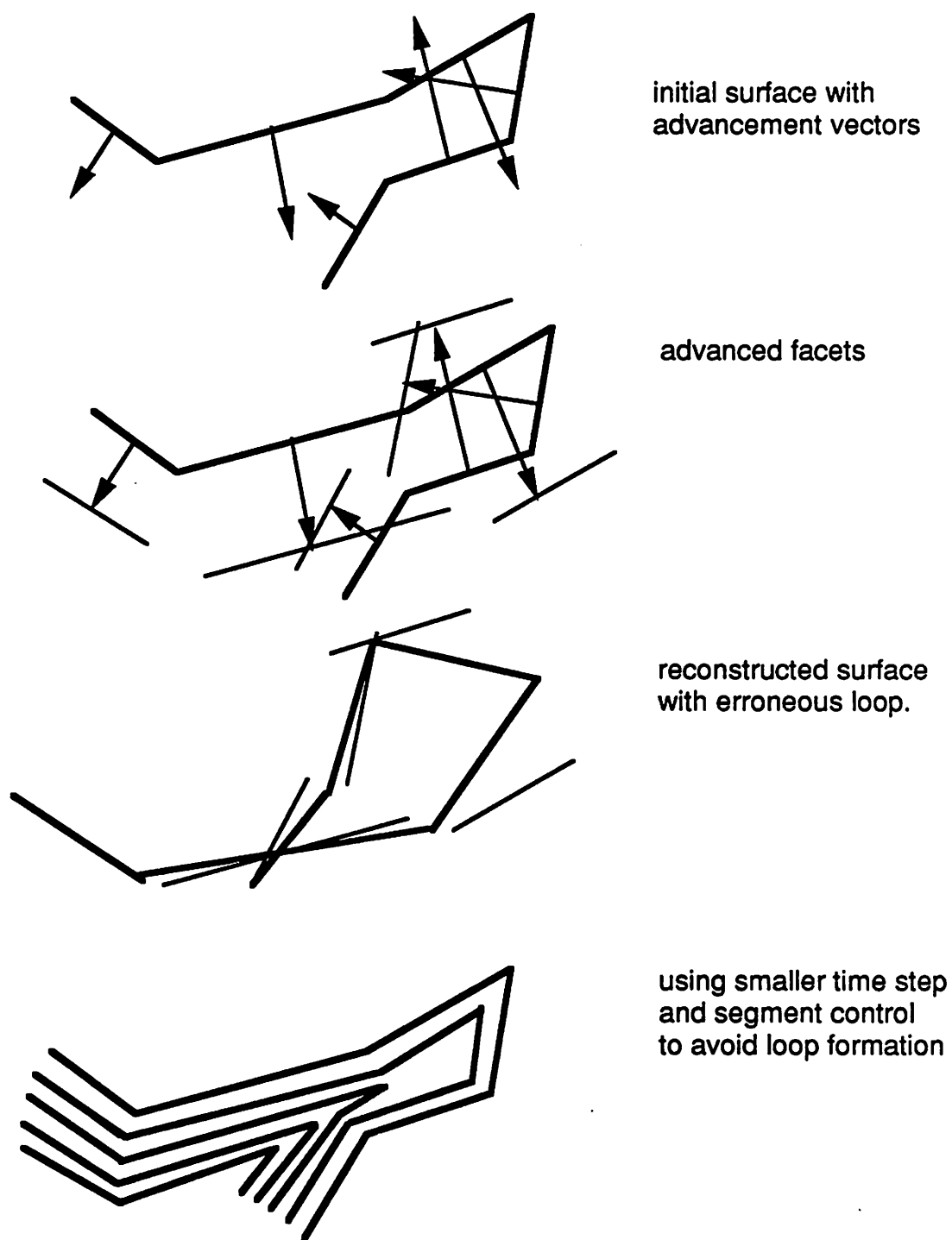
It is also possible to treat the boundary points as special cases, but it is not always easy to know where to move the points since the boundary is a 2D surface. If the material boundaries are planar layers, the motion of the interface is equivalent to motion in a plane for each side of the boundary. This allows for the creation and destruction of new facets attached to the border.

#### **4.5.9. Maximum Time Step In the Facet Motion Algorithm.**

The facet-motion algorithm does not work if facets travel too far in one time step. A case can result where facets overshoot one another as shown in Fig. 4.24 and create erroneous loops. One way to reduce the probability of this occurrence, is to restrict the time step so that the maximum distance traveled by any point is less than some fraction of the minimum allowed triangle edge length. Unfortunately, this cannot guarantee success especially at very sharp corners, made up of many small triangles. The only recourse is to handle sharp corners as special cases, or to provide a method for removing erroneous surface loops. Methods for loop detection and removal are presented in chapter 5.

### **4.6. ALGORITHM IMPLEMENTATION**

The facet-motion algorithm was implemented in the C programming language<sup>15</sup> All of the code examples in this section are written in C or C-like code.



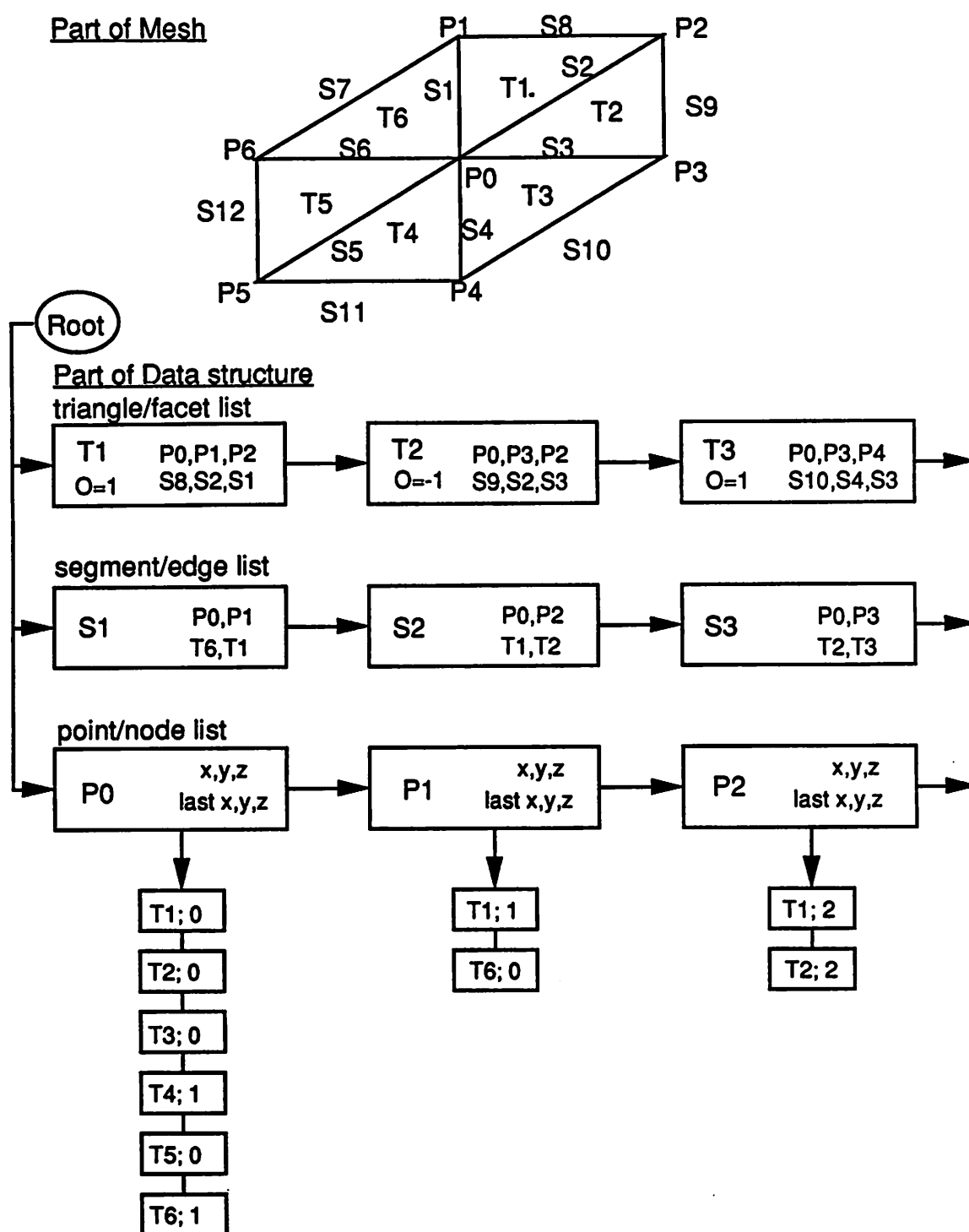
**Fig. 4.24:** Erroneous loop formation and avoidance in the facet motion algorithm.

#### 4.6.1. Data Structure

The data structure must contain information about surface points, and how they are interconnected to form triangular facets. Additional information about the connectivity of the points, facets and facet edges also must be maintained for efficient implementation of the facet-motion algorithm. It should also be possible to scan all the points, all the edges, or all the triangles with a minimum of effort.

Since the surface expands and contracts during simulation, a dynamically allocated data structure is a natural choice for storing the surface representation. The data structure is arranged as three linked lists, one of points, one of facet edges, and one of facets as depicted in Fig. 4.25. Each triangle maintains information about three edges, three points, the facet normal direction, and whether the points are listed in clockwise or counter-clockwise order when viewed along the orientation of the facet normal. Each facet edge (which is also known as a *segment*), maintains information about two points and two triangles attached to that edge, and whether the edge is on the border of the simulation region. Each point maintains its location, the location at the previous time step, and a variable length linked-list with pointers to all the triangles attached to the point. The linked-list of triangles also maintains information about which point on each triangle (0, 1, or 2) is the point for that list. Each data element also includes status flags to store pertinent information such as whether a point lies on the simulation border. It is the responsibility of the mesh refinement routines to maintain a self-consistent data structure. Thus, when a segment is split, the triangles on either side of the segment must be replaced with new triangles, which must in turn be reconnected with the other elements of the data structure.





**Fig. 4.25:** Multiple linked list data structure for surface mesh representation.

Several routines are required to maintain the data structure. These include routines for, point, segment, and triangle allocation, insertion, and deletion; routines for allocating deleting and inserting elements of the list of adjacent triangles at a point; routines for properly splitting and merging segments and their associated triangles and points; and routines for scanning the lists and checking for self-consistency.

#### **4.6.2. Moving the Facets**

Triangles maintain their vertex coordinates by pointing to the memory space containing point coordinates. One point may be shared by many triangles. However, when the facets are advanced, they can no longer share vertexes. An easy way to maintain this information is for each triangle to keep its new coordinates after advancement. The facet-motion solid angle at a point is found by scanning the list of triangles attached to that point and noting the positions of the respective advanced vertexes.

#### **4.6.3. Fast Determination of Line-Triangle Intersections**

There are many times when it is necessary to determine the intersection of a line segment (such as a facet edge) with a triangle. For the facet-motion algorithm, this operation must be performed  $2T^2$  times at each point, where  $T$  is the number of triangles attached to the point. A fast algorithm for making this determination proceeds by first finding the intersection of the line containing the line segment and the plane containing the triangle. If an intersection exists, and the line does not lie on the plane, then a test is made whether this intersection is contained in the line segment and in the triangle. Since the triangle and the intersection are coplanar, the problem can be projected onto a plane and solved in two dimensions.

The following C code determines the intersection of the line and the plane. The input is the endpoints of the line segment, the three triangle vertexes, and the triangle normal. The routine returns 0 for no intersection, 1 if there is an intersection, and 2 if the line and triangle are coplanar. The routine also returns the coordinates of the intersection R. The data type COORD is simply a structure holding three floating point values for the x,y, and z coordinates.

```

line_triangle(p1,p2,tp1,tp2,tp3,tv,R)
COORD *p1,*p2,*tp1,*tp2,*tp3,*tv,*R;
{
    double a1,a2,a3;
    double d,t,tvdotp,h1;
    double sqrlength();
    double fabs();
    double EPS = 1.0e-10;
    if(tri_or_line_degenerate(p1,p2,tp1,tp2,tp3)==TRUE)
        return(0);

    a1 = p2->x - p1->x;
    a2 = p2->y - p1->y;
    a3 = p2->z - p1->z;
    d = (tv->x)*(tp1->x)+(tv->y)*(tp1->y)+(tv->z)*(tp1->z);
    tvdotp = (tv->x)*a1+(tv->y)*a2+(tv->z)*a3;
    h1 = (tv->x)*(p1->x)+(tv->y)*(p1->y)+(tv->z)*(p1->z);
    if(tvdotp == 0.0) { return(2);}

    /*case when line and tri are coplanar */
    /*no determination is made whether line segment*/
    /*intersects the triangle in the plane */
    t = (d - h1)/tvdotp;

    /*find the intersection and store it in R */
    R->x = p1->x + a1*t;
    R->y = p1->y + a2*t;
    R->z = p1->z + a3*t;

    /*return 0 if intersection not on line segment */
    if( t < 0.0 || t > 1.0 ) return(0);
    /*check if intersection point is inside triangle*/
    if(point_on_tri(R,tp1,tp2,tp3,tv)>=1) return(1);{
        return(0);
    }
}

```

The above routine first checks that the triangle edges and line segment are non-degenerate, *i.e.* they have non-zero lengths. If the intersection lies on the line segment, a routine is called to test whether the intersection is inside the triangle.

This routine projects the triangle and point onto the  $z$  plane so long as the plane orientation vector has a non-zero  $z$ -projection. If that is not true, the projection is made onto the  $x$  or  $y$  plane. The inside/outside test follows a simplified version of the method given by Preparata and Shamos.<sup>16</sup> for a point of interest  $z$  and a triangle  $p_1p_2p_3$ , the algorithm determines whether angles  $p_3p_1z$ ,  $zp_1p_2$ , and  $p_2p_3z$  are right or left turns by evaluating a  $3 \times 3$  matrix for the three points making up the angle:

$$\det \begin{bmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{bmatrix} \quad (4.6)$$

If the sign of the determinant is +, the angle is a left turn. The point  $z$  is inside the triangle if and only if the three angles are all left turns or all right turns. The algorithm requires at most three  $3 \times 3$  determinants with 1's in the third column yielding 9 multiplications and 15 additions worst case. If the point  $R$  is inside the triangle, or on any of its edges or vertexes, the function below returns 1, else 0 is returned:

```
fast_point_on_tri_new(R,tp1,tp2,tp3,tv)
COORD *p1,*tp1,*tp2,*tp3,*tv;
{
    double zqp,zqp1,pplz;
    double dotp,fabs();

    if(point_on_vertex(R,tp1,tp2,tp3)==TRUE) return(1);
    if(fabs(tv->z) > 0.01){
/*project to z-plane*/
        zqp1=-((R->x)*(tp1->y-tp3->y)-(tp1->x)*(R->y-tp3->y)
                +(tp3->x)*(R->y-tp1->y));
        zqp=(R->x)*(tp1->y-tp2->y)-(tp1->x)*(R->y-tp2->y)
                +(tp2->x)*(R->y-tp1->y);
```

```

        if (! ((zqp1>=0.0&&zqp>=0.0) || (zqp1<=0.0&&zqp<=0.0))) {
            pplz = (tp2->x)*(tp3->y-R->y)
                  - (tp3->x)*(tp2->y-R->y)
                  + (R->x)*(tp2->y-tp3->y);

            if ((zqp1>=0.0&&zqp>=0.0&&pplz>=0.0) ||
                (zqp1<=0.0&&zqp<=0.0&&pplz<=0.0)) {
                return(1);
            } else {return(0);}
        } else { return(0);}
    etc.
}

```

The combination of degeneracy checking and projection to 2D yields a very fast algorithm for 3D line-triangle intersections.

#### 4.6.4. Finding the Fastest or Slowest Direction in a Solid Angle

In topography simulation of direction dependent processes, the fastest or slowest possible etch directions are known in advance, following a form similar to that in Fig. 4.26 where the axes are the  $\phi$  and  $\theta$  angles in spherical coordinates. Finding the extrema directions in a solid angle is a problem of determining whether any of the maximal rate directions are contained in the solid angle and whether that rate extrema are larger or smaller than any direction on the solid angle boundary. This can be reduced to a 2D polygon inclusion problem. All of the vectors bordering the facet-motion solid angles intersect a plane perpendicular to the axis of the solid angle as shown in Fig. 4.27. The intersection points form a simple polygon. The question becomes: Does the rate extremum direction intersect the plane and does the intersection lie within the solid-angle polygon? The facet-motion vectors are not listed in order, but the polygon edges can be extracted from the connectivity information of the original facets. This can be done in  $O(T)$  time where  $T$  is the number of triangles attached to the point. Once the polygon edges are known, the point inclusion question to be answered in  $O(\log(T))$  time,

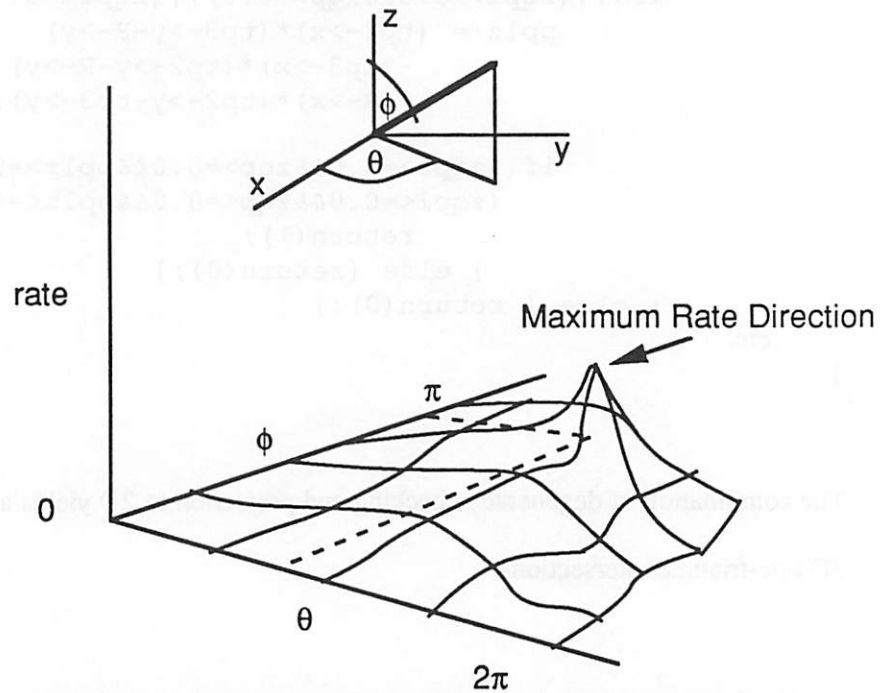


Fig. 4.26: Typical rate versus angle plot, showing local maximum at a given  $\phi, \theta$ .

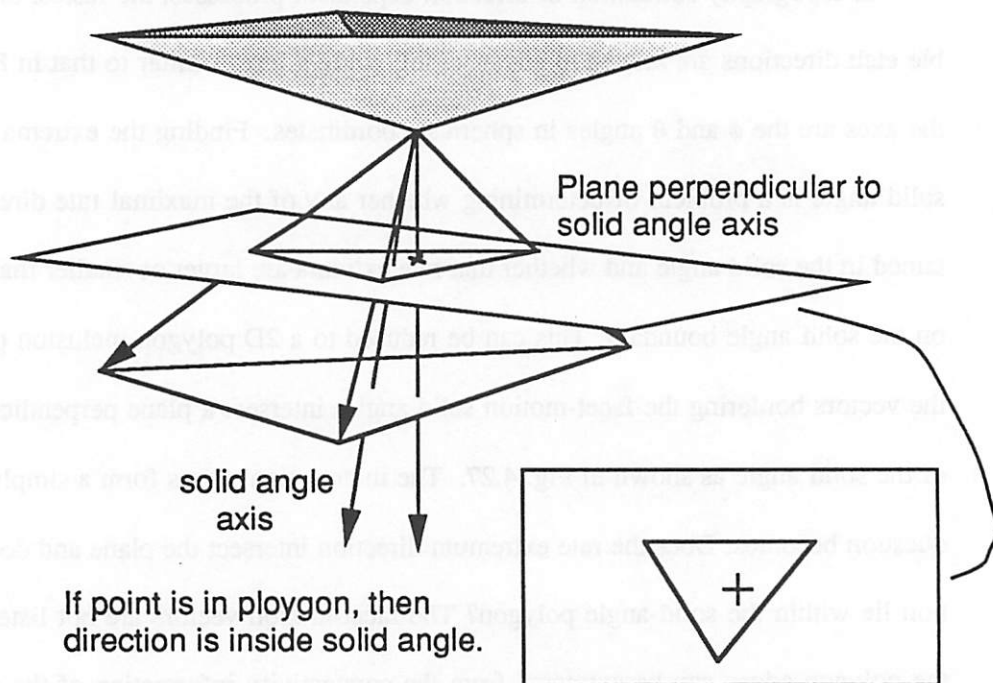


Fig. 4.27: Method for determining whether a direction is within facet-motion solid angle.

since the polygon is generally convex and at worst star-shaped. The algorithm for this determination along with the proof of  $O(\log(T))$  time is described by Preparata and Shamos:<sup>16</sup>

For a query point  $z$ , determine by binary search the wedge of the polygon in which it lies. Given a point  $q$  inside the polygon, the point  $z$  lies between the rays defined by polygon vertexes  $p_i$  and  $p_{i+1}$  if and only if angle  $zqp_{i+1}$  is a right turn and angle  $zqp_i$  is a left turn. Once  $p_i$  and  $p_{i+1}$  are found, then  $z$  is internal if and only if  $p_i p_{i+1} z$  is a left turn.

The extreme rate direction may not be unique. For example, the rate maximum may depend only on  $\phi$ . In this case, the rate maximum is inside the facet-motion solid angle if at least one facet moves along an angle greater than  $\phi_{\max}$  and at least one facet moves along an angle less than  $\phi_{\max}$ , or the facet-motion solid angle completely surrounds all possible  $\phi_{\max}$  directions. Since the maximum direction is non-unique, the original surface node moves along the fastest direction vector that is closest to the centroid of the facet-motion solid angle.

#### 4.6.5. Rate Dependent Adaptive Time Step

During the scan through the list of points to set the rates, it is possible to determine the maximum possible rate. An acceptable time step is one that guarantees the maximum distance traveled by any facet is less than some fraction of the minimum allowed segment length. This helps avoid the formation of erroneous loops as well as improves the simulation run time by allowing a longer time step when rates are slow.

#### 4.6.6. A Correction Factor for Inhomogeneous Rate Fields

The facet-motion algorithm is correct for uniform rates, but for completeness the issue of motion through a varying velocity field must be considered in more detail. The distance a point travels in a one-dimensional scalar rate field is:

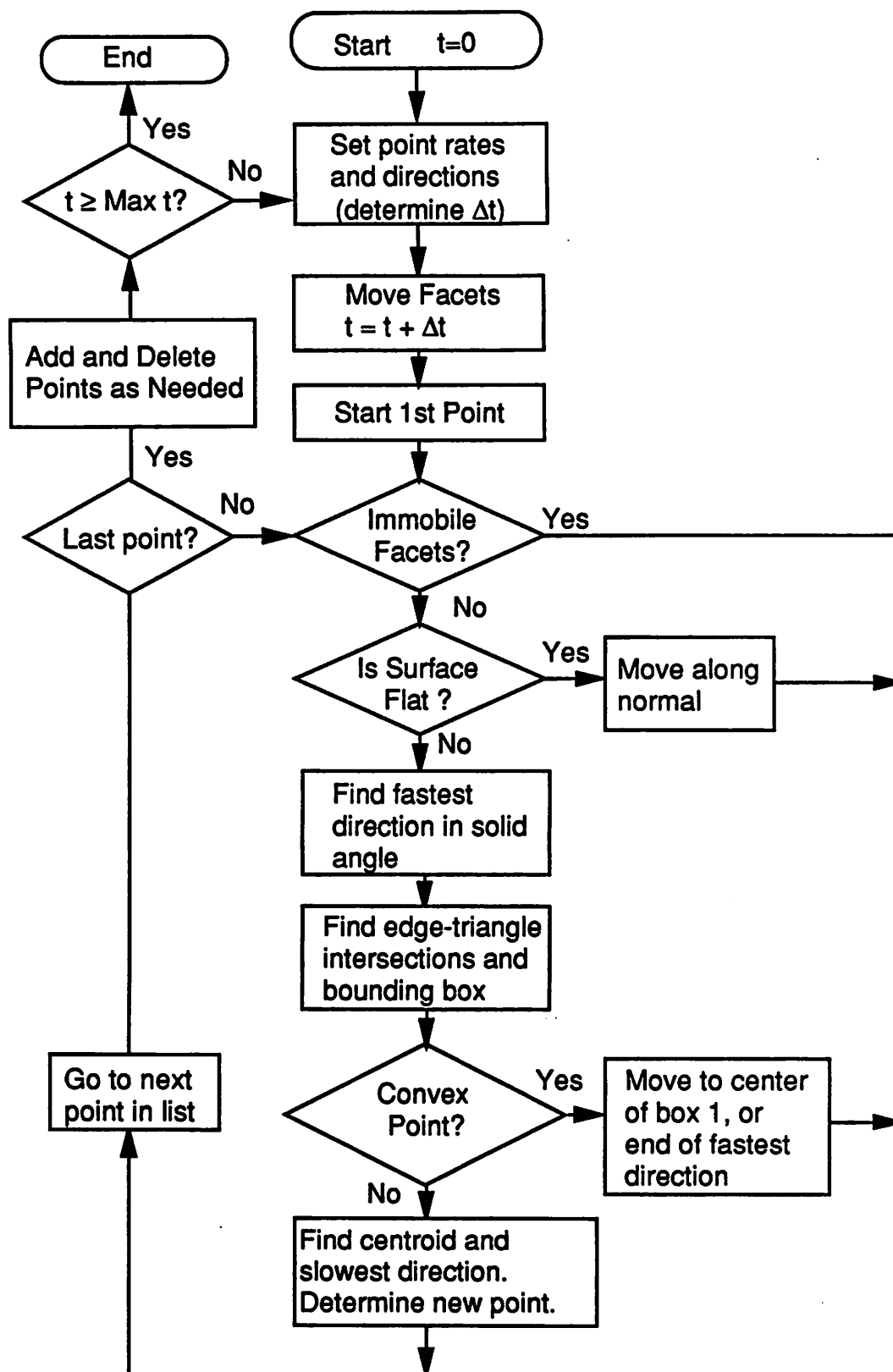


Fig. 4.28: Flow chart for the combined facet motion algorithm for surface advancement.



$$x(t) - x_0 = \int_0^t r(x(t)) dt \quad (4.7)$$

Discretizing the time will result in some error. If the rate  $r(x(t))$  increases with position, the time discretization will underestimate the distance traveled. If  $r(x(t))$  decreases, time discretization will overestimate the distance traveled. If the rate versus position function is known, then it may be possible to solve equation 4.7. For example, if the rate varies linearly with position  $r = Ax + B$ , then the position of the particle at time  $t$  is

$$x(t) = \frac{r(x_0)}{A} (e^{At} - 1) + x_0 \quad (4.8)$$

Unfortunately, in process simulation the function relating the rate with position is usually quite complex. For lithography development simulation, the ray-trace method solves a 3D version of equation 4.7 for an inhomogeneous rate field, resulting in an accurate final profile. The facet-motion algorithm does not solve this equation, but relies on the change in the shape of the surface over several small time steps to achieve a close approximation to the exact result. It is possible to derive a correction factor based on the rate at the initial and final positions of the point over one time step, following the well known predictor-corrector approach.

To estimate a correction factor, a point is first moved along its direction  $\mathbf{v}$ , a distance based on the rate  $r_0$  at the initial position  $\mathbf{x}_0$ :

$$\mathbf{x}_1 = r_0 \Delta t \mathbf{v}_0 + \mathbf{x}_0 \quad (4.9)$$

The etch rate at the new position is then compared with the original etch rate to estimate a linear rate vs. position function:

$$r(\mathbf{x} - \mathbf{x}_0) = A |\mathbf{x} - \mathbf{x}_0| + r(\mathbf{x}_0) \quad (4.10)$$

where

$$A = \frac{r_1 - r_0}{|\mathbf{x}_1 - \mathbf{x}_0|} \quad (4.11)$$

Then by applying equation 4.8 and dividing by  $\Delta t$ , an effective rate is found:

$$r_{eff} = \frac{|\mathbf{x} - \mathbf{x}_0|}{\Delta t} = \frac{r_0}{A \Delta t} (e^{A \Delta t} - 1) \quad (4.12)$$

The  $r_0$  term in equation 4.9 may now be replaced by  $r_{eff}$  to estimate the actual distance traveled by the particle along  $\mathbf{v}$  in the rate field. If the rate function is linear over the full distance traveled, there will be no error for a discrete time step. Typically the rate function is non-linear, however it is very close to linear over time steps small enough to maintain a stable surface. Also, the etch rate matrix for most lithography simulators supplies the actual rates only at certain points. The rate for an arbitrary position is determined by interpolation. In the case that a simple linear interpolation is sufficient, the effective rate given in equation 4.12 adds no error to that already inherent in the interpolation, and solves equation 4.7 exactly for the local velocity field. Interestingly, an expression similar to equation 4.12 was derived independently by Ishizuka for use in a 3D development simulator that traces the time evolution of the surface along fixed directions.<sup>17</sup> In practice, it turns out that a time step such that no point travels more than 10-20% of the minimum allowed segment results in an effective etch rate that is very close to the uncorrected rate. This is due to the small change in the rate field over the maximum displacement allowed to maintain an accurate surface representation. Also, as the surface evolves, the orientations of the facets change. The path traced out by a single surface point is very close to the path given by solving the ray-trace equation due to this reorientation. This result is consistent with previous work by K.K.H. Toh in ray-trace development simulation,<sup>3</sup> further demonstrating the equivalence of the facet-motion and ray-trace algorithms.

There are also some problems with the correction factor method. The computation needed to calculate the  $A$  factor the effective rate results in extra CPU time. This would not be a problem if it were possible to reduce the total number of time steps. But the criterion limiting the maximum distance a facet may travel must be preserved, so the number of time

steps may not be reduced much. Also, the correction scheme is not effective if the point passes through a local etch rate extremum, in which case only a smaller time step will yield the right final position. This is actually an important result which strengthens the general facet-motion algorithm. Since etch rate variation with position is not known in advance for surface-orientation dependent processes, the above result indicates that a small time step should be adequate for general surface advancement simulation.

#### 4.6.7. Putting It All Together

Fig. 4.28 shows a flow chart of the entire algorithm. Certain operations may be skipped if they are not needed in a given etch or deposition model. The time to calculate the new position of each point varies with the number of facets attached to the point, however, the total run time is related to the average number of triangles,  $\bar{T}$  at each point. The average time needed for one node is  $O(\bar{T}^2) + O(\bar{T}) + O(\log(\bar{T}))$ . This is itself a constant, therefore the total run time for  $N$  nodes is  $O(N)$ .

### 4.7. ALGORITHM EVALUATION

#### 4.7.1. Uniform Etch Rate

If the etch rate is the same everywhere in a material, an initial starting point expands into a sphere. Fig. 4.29a shows etching starting from a surface corner, with the rest of the surface effectively masked. Etching proceeds uniformly in all directions resulting in the spherical surface shown. The initial surface had 1586 triangles and 836 points. This evolved into 2328 triangles and 1215 points after etching out to a radius of 0.75. An adaptive time step was used, restricting point advancement to no more than 20% of the minimum allowed segment length.

Points were inserted on mesh segments according to the triangle insertion method of section 4.4, and Fig. 4.3. The total run time was 132 seconds on an IBM PowerStation 530 †

To better see the results, cross-sections of the spherical etch front are shown in Fig. 4.29b. Here, etch fronts at radius 0.25, 0.5, and 0.75 units are shown for two different methods of point insertion. The inner of each pair of curves resulted when points were added simply to the bisector of mesh triangle edges, whenever edges were split. The outer of each pair of curves resulted by putting the new points on the arc as described in Fig. 4.4. By comparing with the radius of a circle, the arc interpolation scheme gives best performance for this case, however the CPU time required was approximately double that for the other set of curves. Increasing the initial density of points decreases the discrepancy between the two curves, but increases the run time dramatically. The point density must be doubled (resulting in over 3000 initial points) to get nearly perfect circular etch fronts, but this quadruples the run time.

#### 4.7.2. Analytic Etch Rate: Gaussian

The performance of the algorithm for spatially varying etch rates can be easily tested by defining an analytic etch rate function. Fig. 4.30a shows the result after 1.0 seconds etch for the rate function:

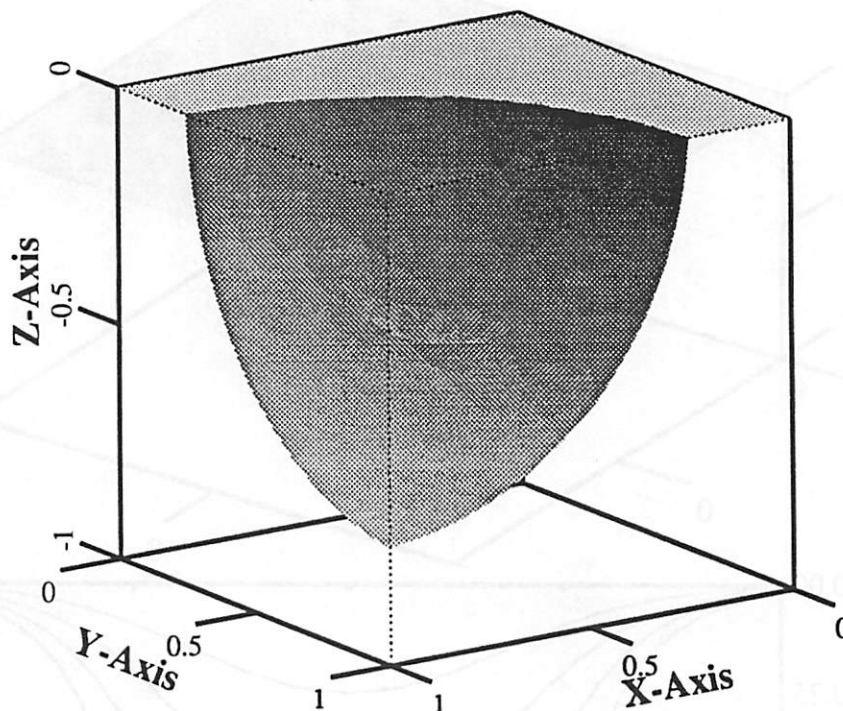
$$rate = \exp[-4(x^2 + y^2)] \quad (4.13)$$

A very smooth surface results for a time step of 0.0042 seconds and an initial mesh of 1444 triangles and 761 points. The final mesh has 2164 triangles and 1126 points, and required 167 seconds of CPU time.

---

† All CPU times in this chapter are quoted for an IBM PowerStation 530, running at 25 MHz and capable of approximately 12 MFlops average performance.

(a)



(b)

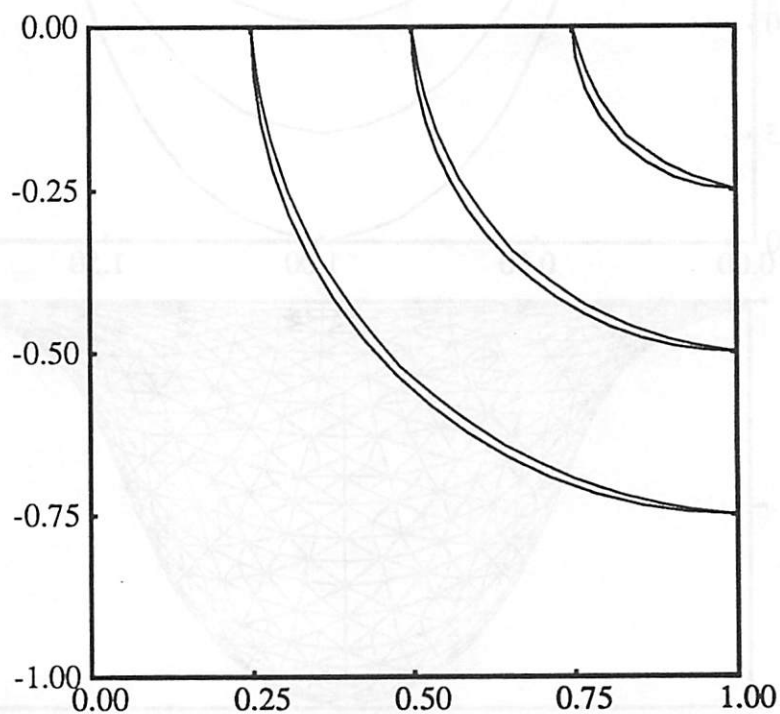


Fig. 4.29: Uniform spherical etching starting from one point. (a) Etching out to radius of 0.75 units. (b) Cross-sections along plane  $y=1$  for 0.25, 0.5, 0.75 units of etching. Two different mesh refinement techniques were used to generate two sets of curves.

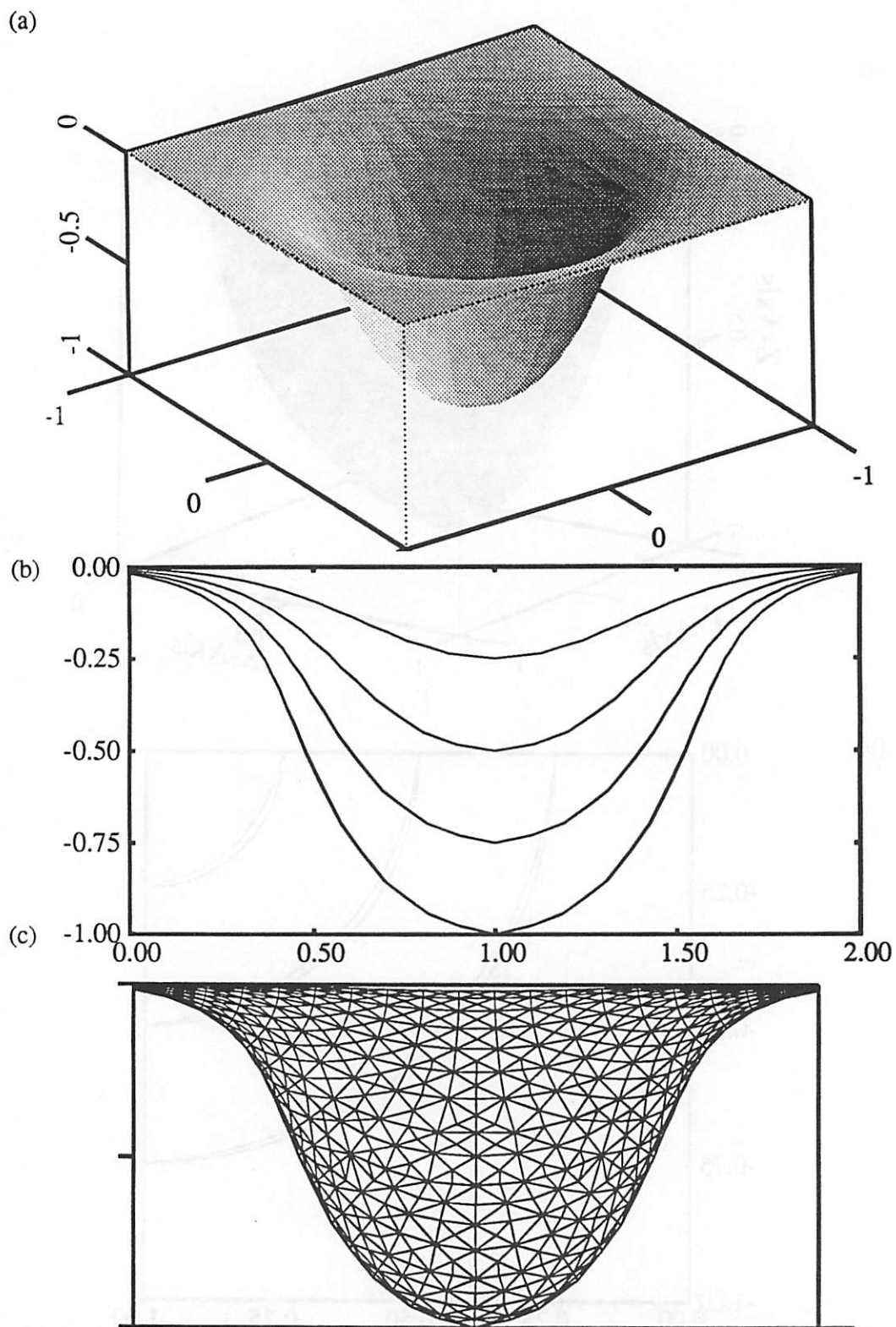


Fig. 4.30: Gaussian etch rate functions  $r=\exp[-4(x^2+y^2)]$ . (a) Result after 1 second etch. (b) cross-section views of etching after 0.25, 0.5, 0.75, and 1.0 seconds. Results are superimposed on ray-trace simulation results. (c) Side view of transparent 3D mesh.

Fig. 4.30b shows a cross-section along the  $yz$ -plane for time intervals of 0.25, 0.5, 0.75, and 1.00 seconds (the  $x$ -axis scale is shifted +1.0). The results for the facet-motion algorithm are superimposed on a plot of results for the same simulation using a ray-trace algorithm. No difference can be seen. This verifies the equivalence of the two algorithms. Fig. 4.30c is a side view of the simulation. In this view the mesh is transparent, showing that basic symmetry of the rate function is preserved by the simulation.

#### 4.7.3. Analytic Etch Rate: Triangular

The previous rate function resulted in a largely concave feature. To better test how the algorithm handles sharp corners, triangular and conical rate functions are employed. The triangular function is:

$$rate = 2|x| + 0.2 \quad (4.14)$$

The conical rate function is:

$$rate = 2\sqrt{x^2 + y^2} + 0.2 \quad (4.15)$$

These functions result in a sharp peak in the center of the simulation region.

Fig. 4.31 shows the results after 1 seconds etching. Fig. 4.31a is for the triangular rate function using the facet-motion algorithm. Fig. 4.31b is the result using a ray-trace algorithm. Similarly, Fig. 4.31c and 4.31d are for the conical etch function for facet-motion and ray-trace algorithms. In both cases, the ray-trace algorithm resulted in a loop in the surface where the rays crossed. By choosing a small time step such that no facet point travels more than 10% of the minimum allowed segment length, the facet-motion algorithm was able to avoid loop formation.

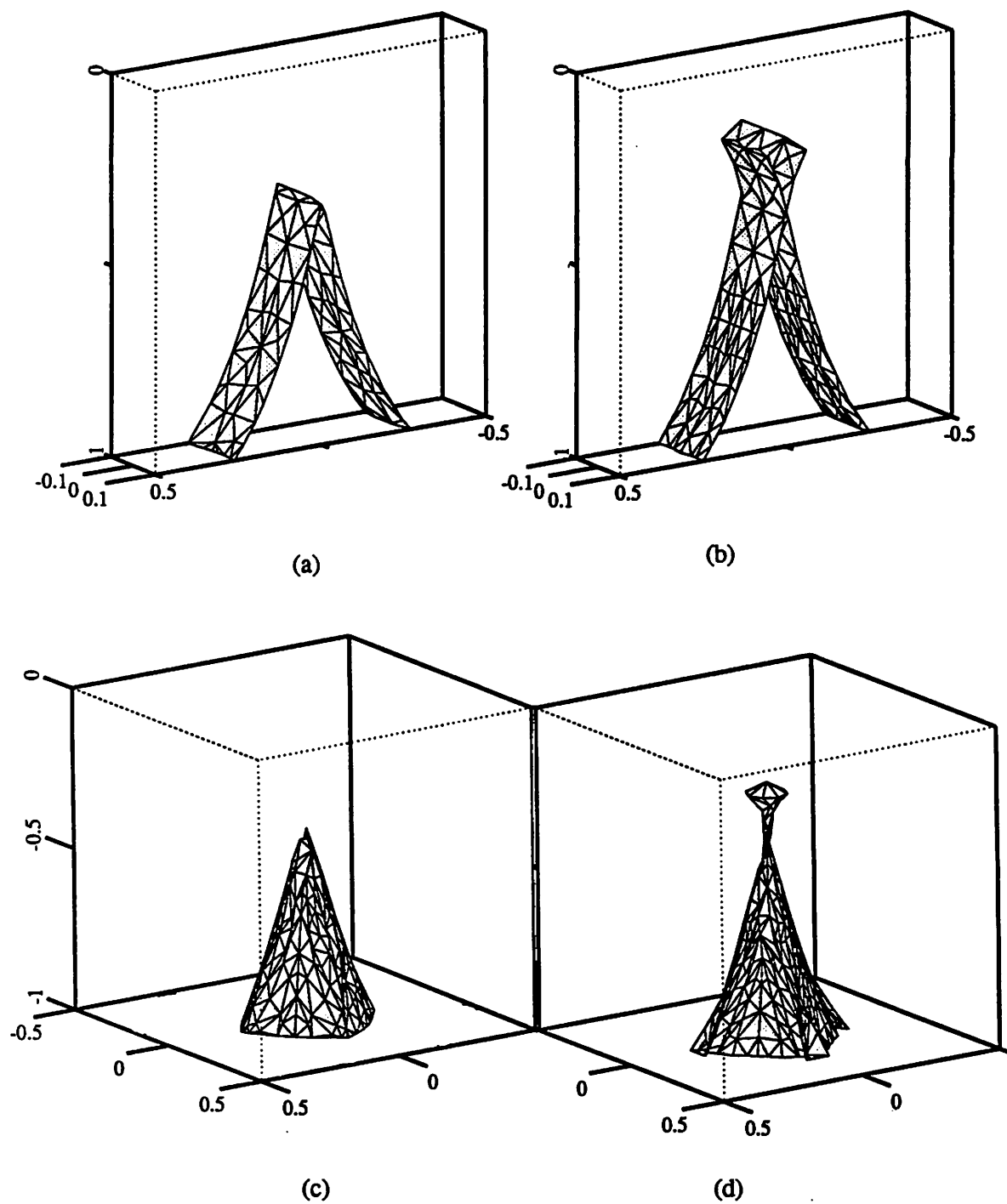


Fig. 4.31: Triangular ( $r = 2|x| + 0.2$ ) and conical ( $r = 2(x^2 + y^2)^{1/2} + 0.2$ ) etch rate functions.  
 (a) Facet-motion algorithm for triangular. (b) Ray-trace for triangular  
 (c) Facet-motion algorithm for conical (d) Ray-trace for conical. Time = 1.0s



Fig. 4.32 shows cross-sectional views of triangular etching at time intervals of 0.25, 0.5, 0.75, and 1.00 seconds (the x-axis scale is shifted +0.5) for the facet-motion (4.32a) and ray-trace (4.32b) algorithms. The only difference in the two sets of curves is the loop avoidance in the facet-motion algorithm.

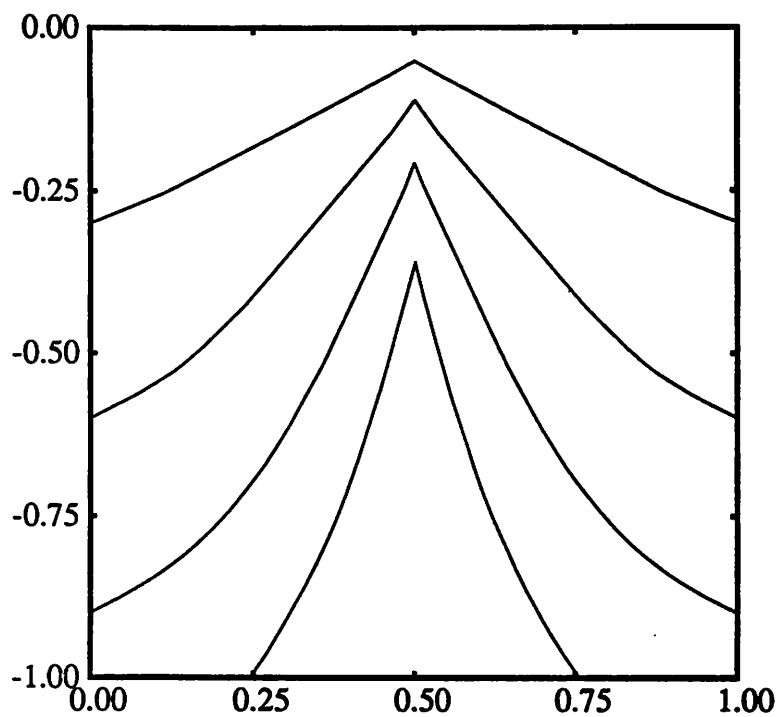
#### 4.7.4. Inhomogeneous Rates in Lithography

A dissolution rate matrix for a photoresist with strong standing wave effects is a particular challenge for 3D topography simulation. Comparing cross-sections of 3D line edge development with the 2D results from SAMPLE 1.8a provides additional insight into the facet-motion algorithm. Fig. 4.33a shows the SAMPLE results for 10 and 20 seconds development under the following conditions:  $\lambda = 500\text{nm}$ ,  $\text{N.A.} = 0.5$ ,  $\sigma = 0.7$ , defocus of  $1.5\mu\text{m}$ , dose =  $150\text{ mJ/cm}^2$ , for KTI 820 resist. † Fig. 4.33b shows a cross-section of the 3D result using the ray-trace algorithm after 10 seconds development evidencing loops at the standing waves. Fig. 4.33c shows the facet-motion algorithm result with an initial mesh density of 15 points along the x-axis. Clearly, more points are needed. Fig. 4.33d shows the result when 30 initial points are used. The result is better, and loops are eliminated if the facet advancement in one time step is limited to 5% of the minimum allowed segment length. There is some inaccuracy at the regions of highest curvature. A mesh refinement method whereby the mesh density is increased in regions where the etch rate changes rapidly was employed to generate Fig. 4.33e, yielding the best results thus far.

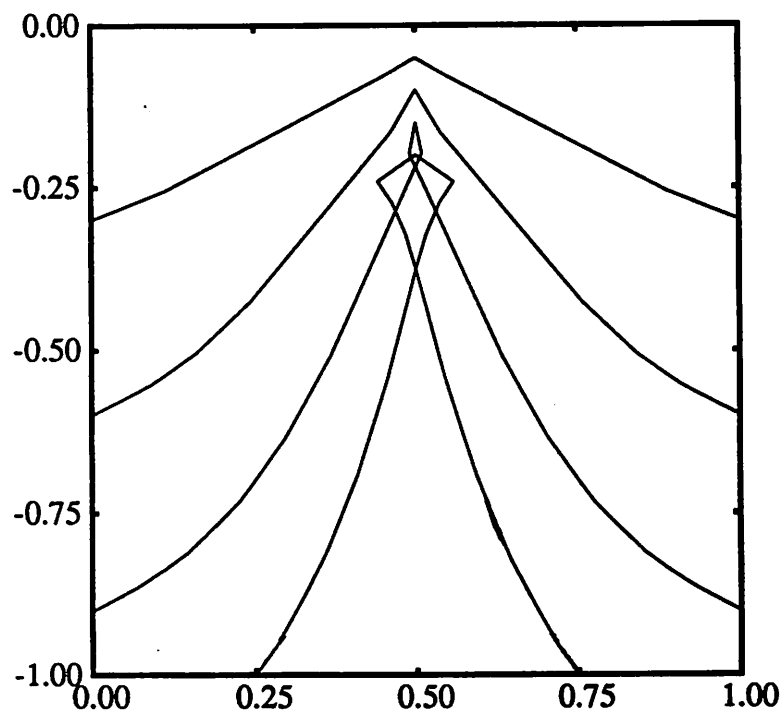
The last figure, Fig. 4.33f, shows what happens when loops occur in the facet-motion algorithm. Unlike the ray-trace method, if a loop forms, it rapidly gets out of control. This is

---

† These are the same conditions used in chapter 3 to generate Fig. 3.3.

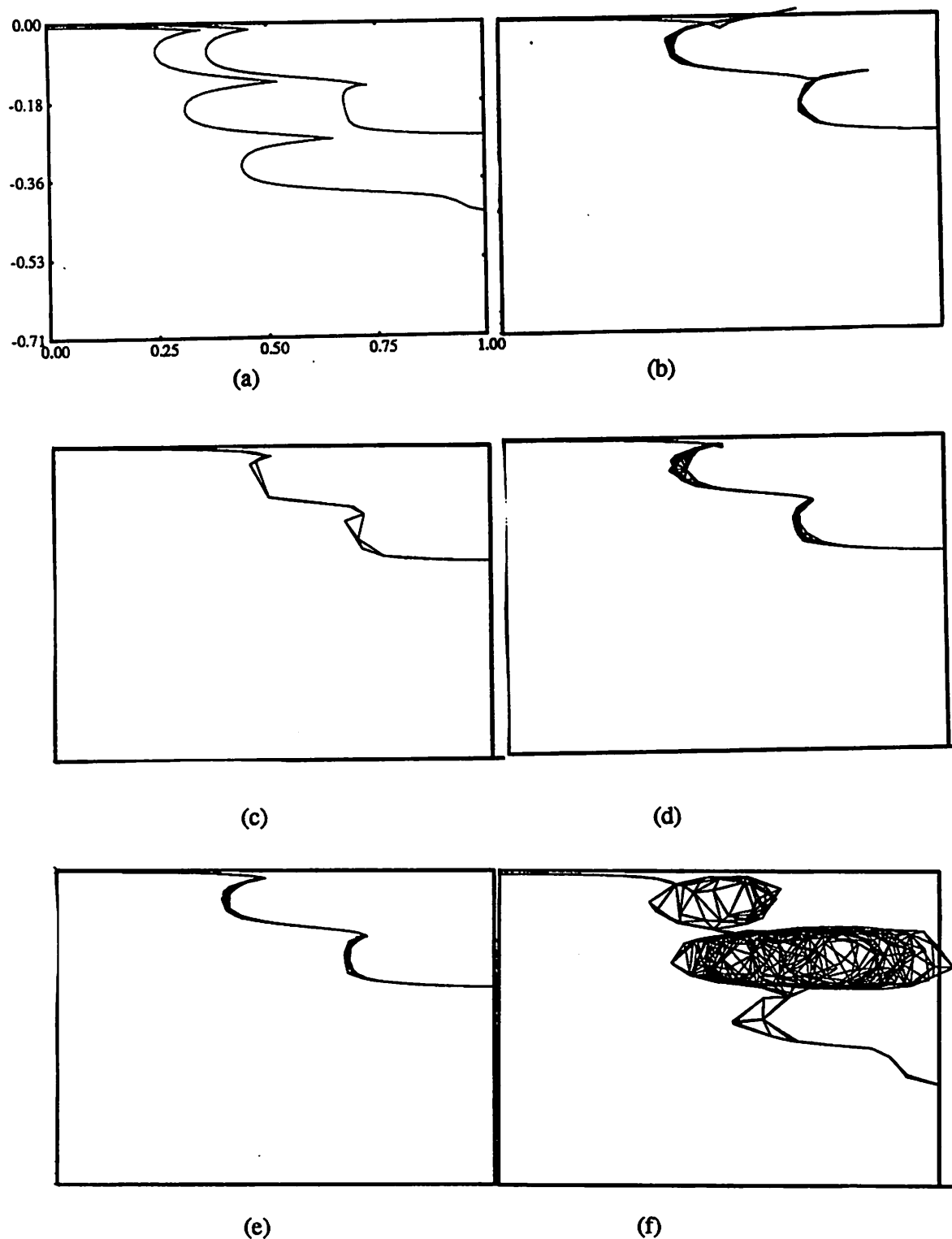


(a)



(b)

**Fig. 4.32:** Cross-sections of triangular etch rate function for 0.25, 0.5, 0.75, and 1.0s.  
 (a) Loop avoidance with facet-motion algorithm. (b) Loop formation with ray-trace algorithm.



**Fig. 4.33:** Cross-section views of line edge lithography development. (a) SAMPLE 2D result for 10s & 20s. (b) Ray-trace for 10s. (c) Facet-motion algorithm for 15 initial points on x-axis. (d) 30 initial points. (e) Adaptive mesh refinement based on change in local etch rate. (f) Loop formation in the facet-motion algorithm.

a result of the inherent surface dependence of the facet-motion algorithm. If the surface is bad, the result is likely to be worse! In theory, reducing the time step should reduce the potential for loop formation, but even reducing the time step 10 times cannot guarantee that no loops form throughout the resist development. A method for loop detection and removal is discussed in chapter 5.

#### 4.7.5. Isotropic Etching with a Mask

Pattern transfer means that a pattern in one layer is transferred to the next layer. The top layer is a mask that etches slowly with respect to the layer underneath. Ideally, the mask layer does not etch at all. Often it is convenient in simulation to think of the mask as a flat 2D object that prevents etchant from attacking any material covered by the mask. In isotropic etching, etchant attacks any material not covered by the mask, and removes material underneath the mask edge as depicted in Fig. 4.34. From the top as in Fig. 4.35, the material surface directly beneath the mask dissolves isotropically away from the mask edge.

For the three-dimensional facet-motion algorithm, it is convenient to consider the mask edge as a 2D string of points, much like the surface intersections with the vertical planes bounding the simulation region. An arbitrary mask can be described as a collection of rectangles and triangles as shown in Fig. 4.36a. The surface mesh is then modified by splitting all the segments that intersect with the mask edge and defining the new points as mask border points as shown in Fig. 4.36b. The mask border points then evolve away from the mask edge according to the local etch rate and the orientation of the mask border string, as depicted in Fig. 4.36c.

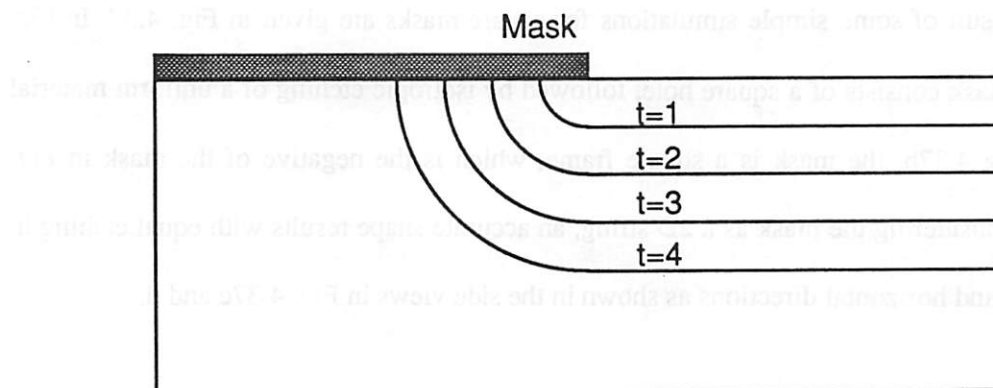


Fig. 4.34: Side view of isotropic etching under a mask edge

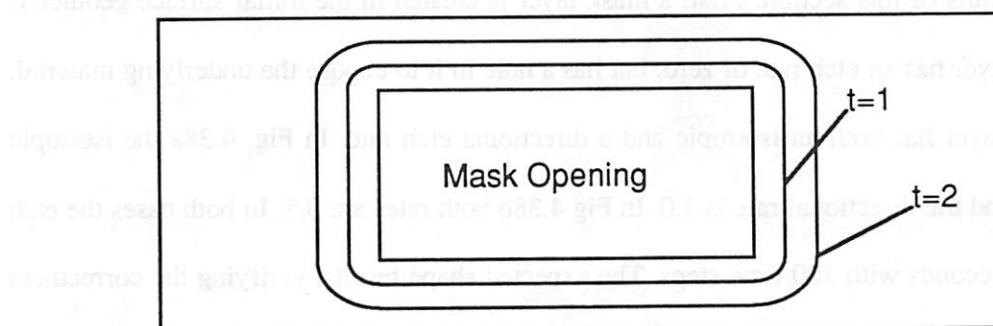


Fig. 4.35: Top view of isotropic etching away from a mask

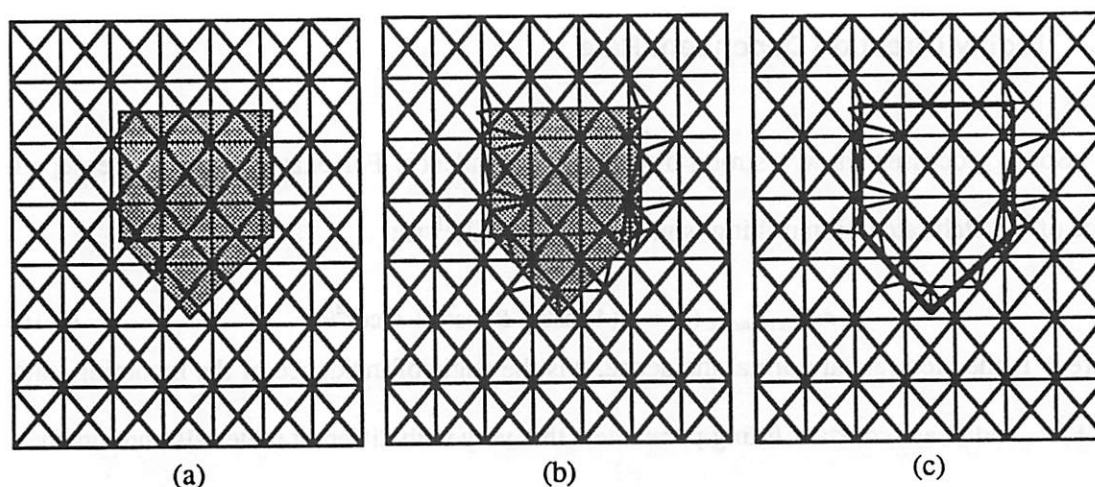


Fig. 4.36: Defining a mask for simulation. (a) Mask as a collection of rectangles and triangles (b) Inserting new mesh points at mask edge. (c) Mask border string

The result of some simple simulations for square masks are given in Fig. 4.37. In Fig. 4.37a, the mask consists of a square hole, followed by isotropic etching of a uniform material layer. In Fig 4.37b, the mask is a square frame, which is the negative of the mask in Fig. 4.37a. By considering the mask as a 2D string, an accurate shape results with equal etching in the vertical and horizontal directions as shown in the side views in Fig. 4.37c and d.

#### 4.7.6. Simple Directional Rate

The simple cosine model for directional etching, given by equation 4.3, is used to generate the results of this section. First, a mask layer is created in the initial surface geometry. The mask layer has an etch rate of zero, but has a hole in it to expose the underlying material. The lower layer has both an isotropic and a directional etch rate. In Fig. 4.38a the isotropic rate is 0.0 and the directional rate is 1.0. In Fig 4.38b both rates are 0.5. In both cases the etch time is 0.5 seconds with 100 time steps. The expected shape results, verifying the correctness of the facet-motion algorithm for this etch model.

#### 4.7.7. Highly Direction Dependent Rate

Some processes show a strong directional dependence. For example, the angle dependence of the etch rate in ion milling is given in SAMPLE<sup>18</sup> as:

$$rate_{directional}(\theta) = R(A \cos\theta + B \cos^2\theta + C \cos^4\theta) \quad (4.16)$$

where  $R$  is the etch rate at normal incidence,  $\theta$  is the angle of incidence of the incoming ions, and  $A$ ,  $B$ , and  $C$  are constant fitting parameters that vary with different materials and ion energies. This etch rate rises as the angle of incidence increases, until a peak is reached, after which the rate falls rapidly. The ratio between the peak etch rate and the rate at normal incidence may be 5:1 or greater.

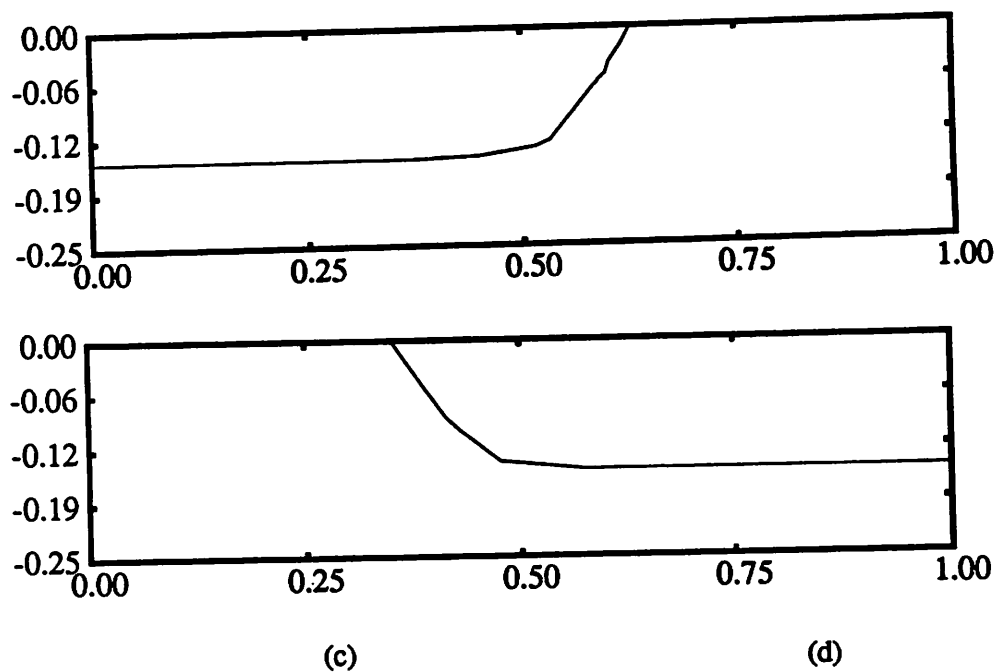
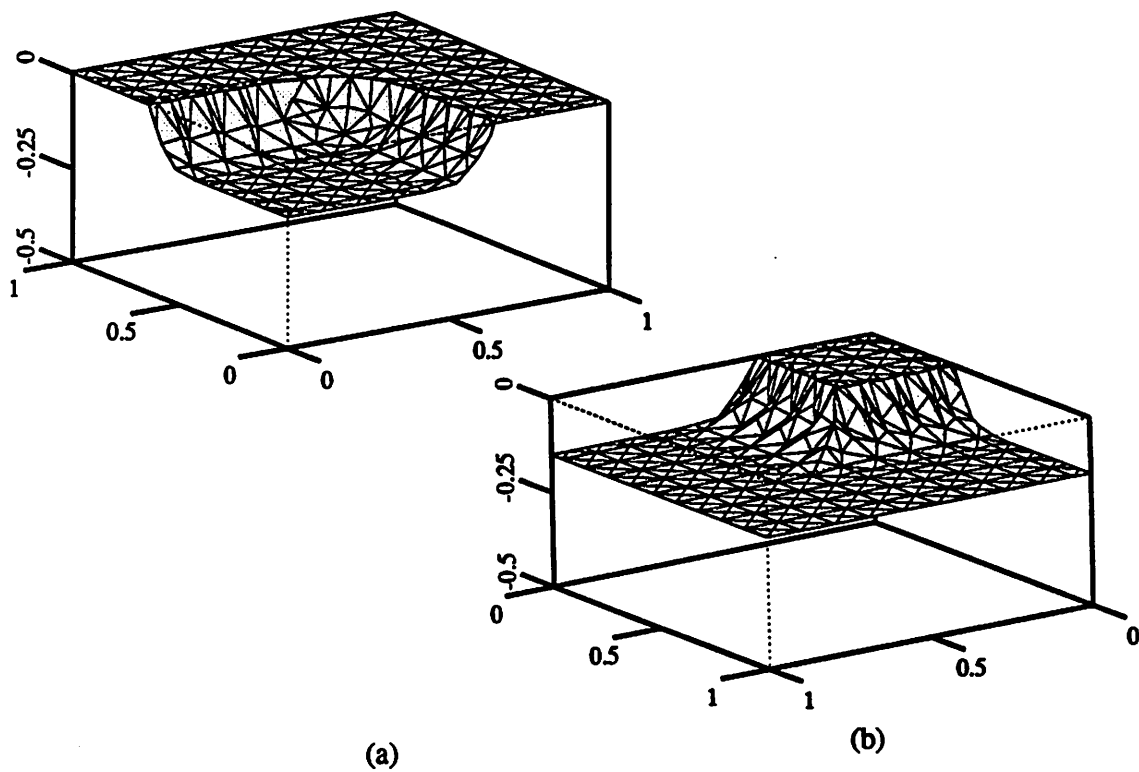
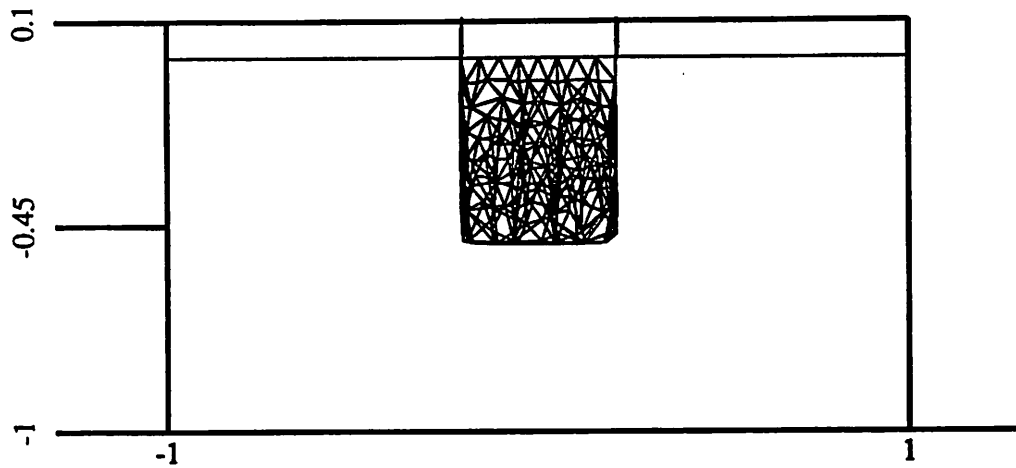
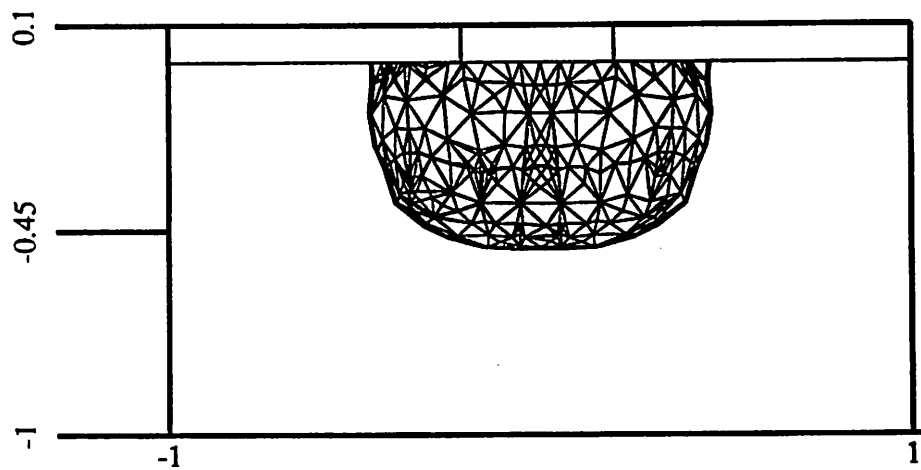


Fig. 4.37: Isotropic etch simulation with masks. (a) Square hole. (b) Square mask. (c) side view of (a). (d) side view of (b).



(a)



(b)

Fig. 4.38: Simple directional etching, side views after 0.5 seconds:  
 (a) directional rate = 1.0, isotropic rate = 0.0. (b) directional rate = 0.5  
 isotropic rate = 0.5.

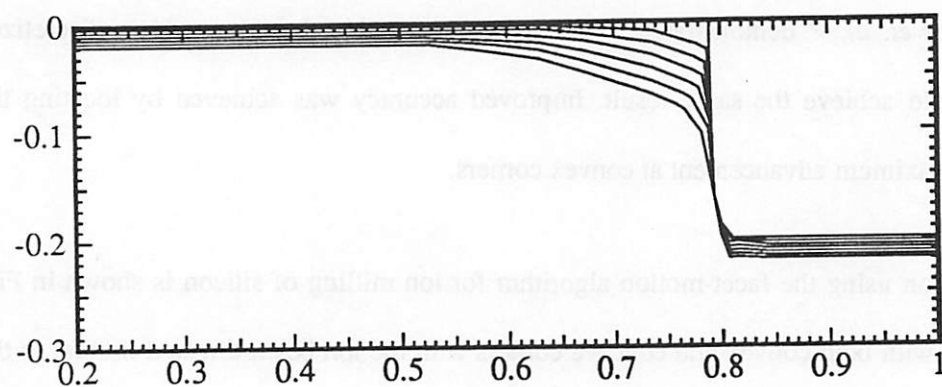


An important effect of the angular rate dependence in equation 4.16 is the appearance of facets at convex corners which contain the fastest etch rate direction. An analytic 2D solution for the ion milling of Si was derived by Ducommun *et. al.*<sup>5</sup> Simulations of the same situation by Neureuther *et. al.*,<sup>19</sup> demonstrated that a string-based etch algorithm with a discretized time step could achieve the same result. Improved accuracy was achieved by locating the direction of maximum advancement at convex corners.

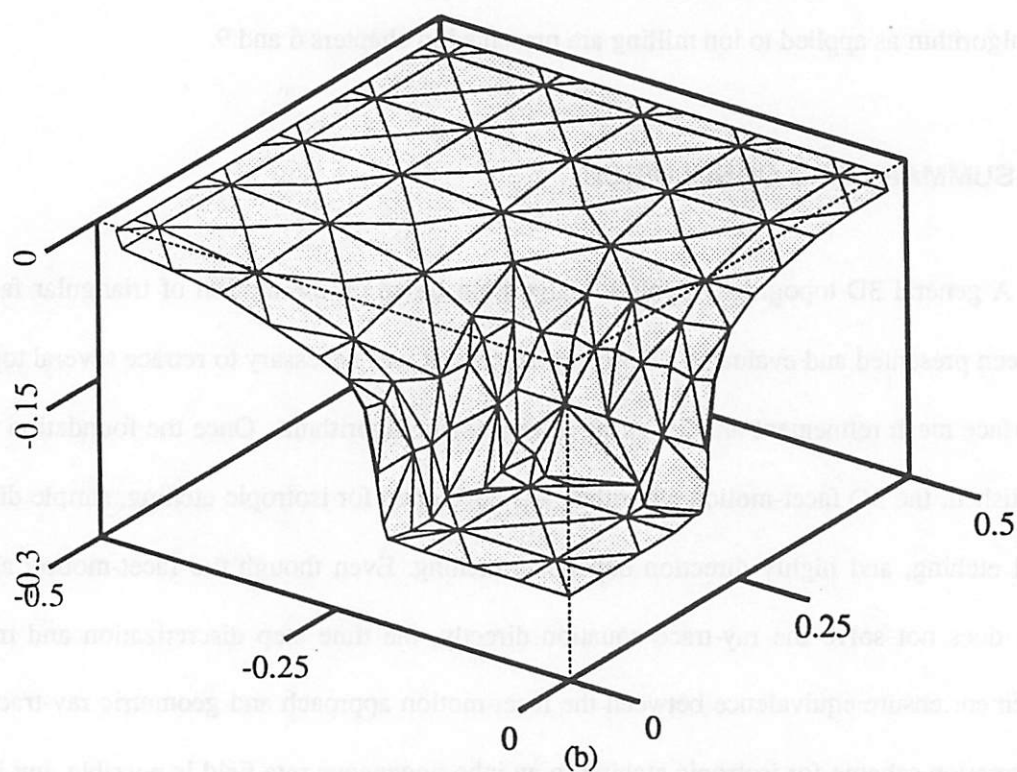
Simulation using the facet-motion algorithm for ion milling of silicon is shown in Fig. 4.39, starting with both convex and concave corners with the ion beam directed normal to the wafer plane. The simulation used equation 4.16, with  $A=3.2696$ ,  $B=13.1059$ , and  $C=-15.3755$ , giving a maximum etch rate along  $\phi=38^\circ$ . Fig. 4.39a shows a cutaway of a 3D simulation. Fig. 4.39b gives the cross sections at several equal time intervals. Further examples of the etch algorithm as applied to ion milling are presented in chapters 6 and 9.

#### 4.8. SUMMARY AND CONCLUSION

A general 3D topography evolution algorithm based on the motion of triangular facets has been presented and evaluated. For completeness, it was necessary to retrace several topics in surface mesh refinement and 2D string advancement algorithms. Once the foundation was established, the 3D facet-motion algorithm was developed for isotropic etching, simple directional etching, and highly direction dependent etching. Even though the facet-motion algorithm does not solve the ray-trace equation directly, the time step discretization and mesh refinement ensure equivalence between the facet-motion approach and geometric ray-tracing. A correction scheme for isotropic etching in an inhomogeneous rate field is possible, but is of limited benefit since a time step small enough to ensure a valid mesh already yields sufficient accuracy. Simulation examples for several cases support the validity of the algorithm.



(a)



(b)

Fig. 4.39: Highly directional rate like that in ion milling. (a) Cross-section of ion-milling of masked Si. (b) 3D view of ion milling

Deposition was not considered directly, but the algorithms are equally valid for negative etch rates to model film growth.

For different physical models, different levels of generality may be employed. The least general simple and isotropic etching cases are computationally fastest, and are adequate for development simulation, or models which give the node advancement vectors independent of the surface orientation. The most general case is needed for processes with a strong dependence on surface orientation, such as ion milling.

Limitations of the algorithm also exist. A fairly small time step (such that no node travels a distance more than 5-20% of the minimum segment length) is needed to preserve accuracy. In the case of sharp corners, the intersection checking feature of the algorithm tends to eliminate loops, but only if the time step is small and the corners are not too sharp. This points to the need for an effective deloop technique. Also, the algorithm by itself does not address intra-surface geometric effects such as shadowing and visibility. Algorithms for deloop and visibility operations are discussed in the next chapter.

## REFERENCES

1. G. Farin, *Curves and Surfaces for Computer Aided Geometric Design*, Academic Press, Boston, 1990.
2. A. Moniwa, T. Matsuzawa, T. Ito, and H. Sunami, "A Three-Dimensional Photoresist Imaging Process Simulator for Strong Standing-Wave Effect Environment," *IEEE Transactions on Computer Aided Design of Integrated Circuits*, vol. CAD-6, no. 3, pp. 431-437, May 1987.
3. K.K.H. Toh, "Algorithms for Three-Dimensional Simulation of Photoresist Development," Memo. No. UCB/ERL M90/123, Ph.D. Dissertation, University of California, Berkeley, December 14, 1990.
4. I.A. Blech, "Evaporated Film Profiles Over Steps in Substrates," *Thin Solid Films*, vol. 6, pp. 113-118, Elsevier Sequoia S.A., 1970.
5. J.P. Ducommun, M. Cantagrel, and M. Moulin, "Evolution of well-defined surface contour submitted to ion bombardment: computer simulation and experimental investigation," *Journal of Materials Science*, vol. 10, pp. 52-62, 1975.
6. A.R. Neureuther, C.H. Ting, and C.-Y. Liu, "Application of Line-Edge Profile Simulation to Thin-Film Deposition Processes," *IEEE Transactions on Electron Devices*, vol. ED-27, no. 8, pp. 1449-1455, August 1980.
7. J.L. Reynolds, *Characterization of Plasma Etched Structures in IC Processing*, Ph.D. Dissertation, University of California, Berkeley, 1983.
8. C.W. Jurgensen and E.S.G. Shaqfeh, "Kinetic Theory of Bombardment Induced Interface Evolution," *Journal of Vacuum Science Technology B*, vol. 7, no. 6, pp. 1488-1492, Nov/Dec 1989.

9. J. McVittie, J. Rey, L.-Y. Cheng, A. Bariya, S. Ravi, and K. Saraswat, "SPEEDIE: A Profile Simulator for Etching and Deposition," *TECHCON '90, Extended Abstract Volume*, pp. 16-19, Semiconductor Research Corporation, San Jose, California, October 16-18, 1990.
10. S. Tazawa, S. Matsuo, and K. Saito, "A General Characterization and Simulation Method for Deposition and etching Technology," *to appear: IEEE Transactions on Semiconductor Manufacturing*.
11. A.G. Dirks and H.J. Leamy, "Columnar Microstructure in Vapor-Deposited Thin Films," *Thin Solid Films*, vol. 47, pp. 219-233, 1977.
12. T. Thurgate, "Segment Based Etch Algorithm and Modeling," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. CAD-10, no. 9, pp. 1101-1109, September 1991.
13. S. Wolfram, *Mathematica, A System for Doing Mathematics by Computer*, Addison-Wesley, Redwood City, California, 1988.
14. R. Smith, G. Carter, and M.J. Nobes, "The theory of surface erosion by ion bombardment," *Proceedings of the Royal Society of London A*, vol. 407, pp. 405-433, 1986.
15. B.W. Kernighan and D.M. Ritchie, *The C Programming Language*, Prentice-Hall, Englewood Cliffs, New Jersey, 1978.
16. F.P. Preparata and M.I. Shamos, *Computational Geometry: An Introduction*, p. 43, Springer-Verlag, New York, 1985.
17. T. Ishizuka, "Three-Dimensional Simulation in Photoresist Development," *IEICE (In Japanese)*, vol. J73-C-II, no. 11, pp. 775-785, November 1990.
18. *SAMPLE 1.8 User's Guide*, Electronics Research Laboratory, University of California, Berkeley, 1991.

19. A.R. Neureuther, C.Y. Liu, and C.H. Ting, "Modeling Ion Milling," *Journal of Vacuum Science Technology*, vol. 16, no. 6, pp. 1767-1771, November/December 1979.

## CHAPTER 5

### ALGORITHMS FOR 3D GEOMETRIC OPERATIONS

*Entia non sunt multiplicanda praeter necessitatem.*  
(Entities should not be multiplied unnecessarily.)  
William of Okham, *Quodlibeta*

#### 5.1. GEOMETRIC OPERATIONS IN PROCESS SIMULATION

Many processes are strongly affected by the shape of the surface. For some processes it is sufficient to consider the local shape, for example the surface orientation. Many other processes require information about the shape of the entire surface. The most common requirement is for visibility information. Frequently, one part of the surface will shadow other parts. Additionally, some surface regions will have a restricted view of the "sky" above the wafer. There may also be cases when it is necessary to determine when particles hit one part of the surface but are reflected onto other parts. These effects all vary the rate at which the surface evolves.

Process simulation algorithms can also introduce intra-surface effects. The formation of erroneous loops discussed in the previous chapter is one such example. In order to remove a loop it is necessary to perform a series of geometric operations to determine which part of the surface is in the loop, and which part is not.

The surface mesh representation presented in the previous chapter contains all the information necessary for locating shadows, determining visibility, and identifying loops. However, the data representation does not lend itself to efficient implementation of these operations. In the case of shadow detection, unless a facet shadows itself, it is necessary to determine whether a line from the facet to a source point intersects any other facet. In the worst

case this requires as many 3D triangle-line segment intersection tests as there are triangles describing the surface. It is possible to implement a variety of schemes for making the intersection search more efficient, but for  $M$  surface facets, the shadow test algorithm will require  $O(M)$  time for every surface point  $N$  of interest. Thus the total time to test the entire surface for shadows is  $O(M \cdot N)$ .

If 3D process simulation is to be practical, an efficient mechanism for performing intra-surface geometric operations is necessary. This chapter first discusses visible surface determination techniques used in 3D computer graphics, to provide additional insight into similar problems in process simulation. A new data representation that combines surface and cell based topography representations is then presented. Algorithms for several geometric operations are described which use the cell data representation to achieve computational efficiency without serious accuracy degradation.

## 5.2. VISIBILITY PROBLEMS IN COMPUTER GRAPHICS

Visible surface determination problems have been studied extensively in computer graphics research. The problem can be stated as one of finding all the parts of a surface that are visible given a particular viewpoint. The method described in the previous section is essentially a brute-force ray tracing technique similar to many others used in computer graphics.<sup>1</sup> Most improvements to the ray trace technique rely on speeding up the ray-polygon intersection tests or avoiding them altogether. Still the algorithm remains essentially  $O(M \cdot N)$  for the type of surface considered here.

Another approach commonly used in computer graphics is simply to sort all the surface polygons from back to front, and draw them in that order.<sup>2</sup> Only the objects with nothing on



top of them will be visible. The sort can be performed using a Quicksort algorithm<sup>3</sup> requiring  $O(M \log M)$  time on average for  $M$  triangles. It is easy to imagine sorting all the surface triangles according to their distance from an illumination source, projecting the result onto a 2D plane, and then determining which surface points are covered and which are not. Unfortunately, the last two steps require additional computation time. The 2D transformation step requires time  $O(N)$  proportional to the number of surface points  $N$ . Then each point must be tested against the transformed polygons to see if it is covered. Computer graphics gets around this problem by simply drawing over pixels with an exclusive-OR operation. Full shadow determination however requires a total of  $M \cdot N$  point inclusion tests, which is no better than the brute force ray trace test. Techniques for improving the back-front ordering speed, such as those using octrees,<sup>4</sup> are also of limited use, since they do not reduce the number of 2D intersections that must be tested.

An attractive way to reduce the time spent on intersection tests is to use a bounding volume whose intersection test is less expensive. Rectangular solids are particularly useful.<sup>5</sup> The regular order of rectangular elements, and the ease with which a point can be compared with the minimum and maximum  $x, y$ , and  $z$  of a rectangular prismatic element lead to this efficiency. This observation raises the question: is it possible to represent the surface mesh with rectangular cells to achieve efficient intra-geometric operations?

### 5.3. THE SURFACE-CELL HYBRID DATA REPRESENTATION

Surface based topography simulation algorithms have superior accuracy and efficiency whereas cell based volume removal algorithms offer superior stability. It is also easy to test whether a point is inside a rectangular prismatic elements. It is natural to conclude that a proper combination of these representations might yield accurate and efficient process

simulation. This is the philosophy behind the surface-cell hybrid.

Fig. 5.1 shows a surface advancing through a 3D grid of rectangular prismatic elements organized in a regular array. As the surface advances, any grid elements behind the surface are updated to reflect the fact that the surface has past. It is easy to find the cell containing any coordinate within the simulation region by simple arithmetic operations. Given a coordinate  $(x, y, z)$ , the corresponding cell has an array address as follows:

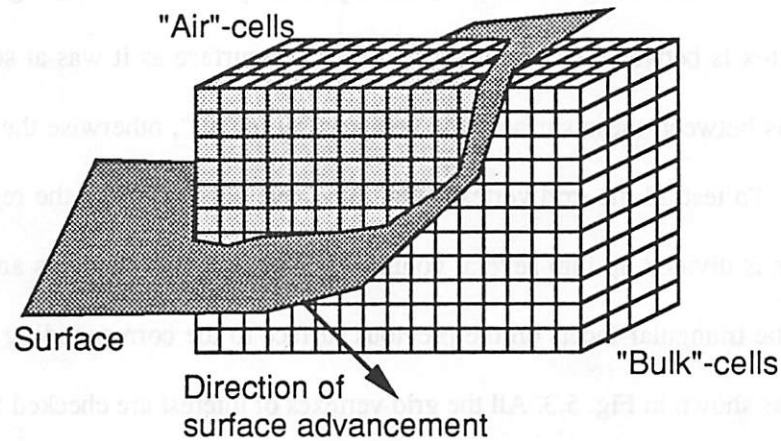
$$\begin{aligned} i &= \text{trunc} \left( \frac{x - x_{\min}}{x_{\max} - x_{\min}} \cdot n_i \right) \\ j &= \text{trunc} \left( \frac{y - y_{\min}}{y_{\max} - y_{\min}} \cdot n_j \right) \\ k &= \text{trunc} \left( \frac{z - z_{\min}}{z_{\max} - z_{\min}} \cdot n_k \right) \end{aligned} \quad (5.1)$$

where  $n_i$ ,  $n_j$ , and  $n_k$  denote the number of elements of  $i$ ,  $j$ , and  $k$  in the simulation region, and the function *trunc* means truncate to eliminate the fractional part. Once the cell is located, it is possible to determine whether the coordinate of interest is in front of or behind the advancing surface.

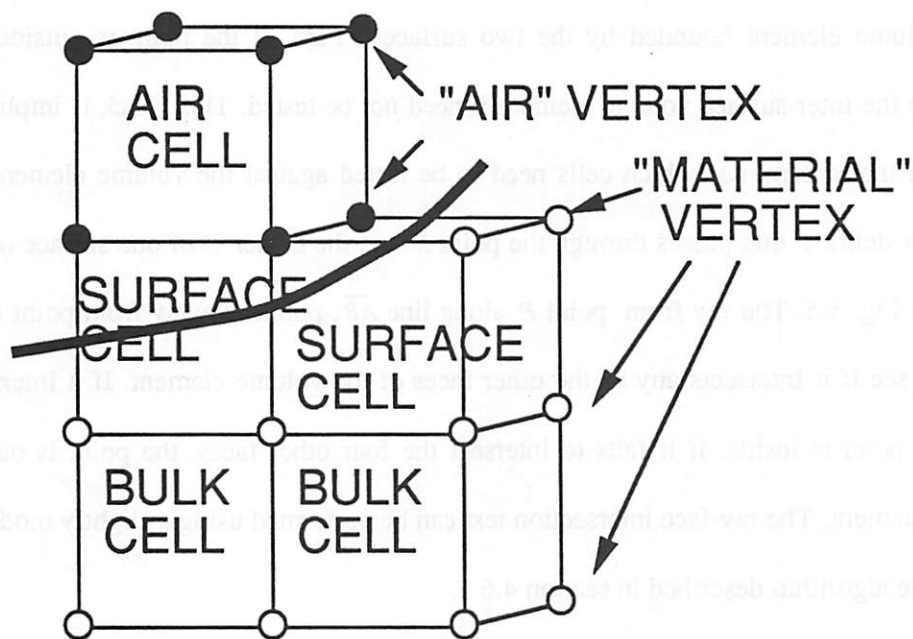
The specific data representation is illustrated in Fig. 5.2. As the surface advances, the vertexes between the cells are updated as "material" or "air". To determine the status of any one cell, it is necessary to consider all eight cell vertexes. If all the vertexes are "material" the cell is in the bulk. If all the vertexes are "air" the cell is in air. If some of the vertexes are "material" and some are "air", the cell is a surface cell.

#### 5.4. ALGORITHM FOR UPDATING THE CELLS

It is essential that the cells be correctly identified as bulk, surface or air. The algorithm making this determination should also be efficient. Both of these goals are met by the cell



**Fig. 5.1:** Surface advancing through a grid of rectangular prismatic elements or "cells". The cells behind the surface are flagged as "air" cells.

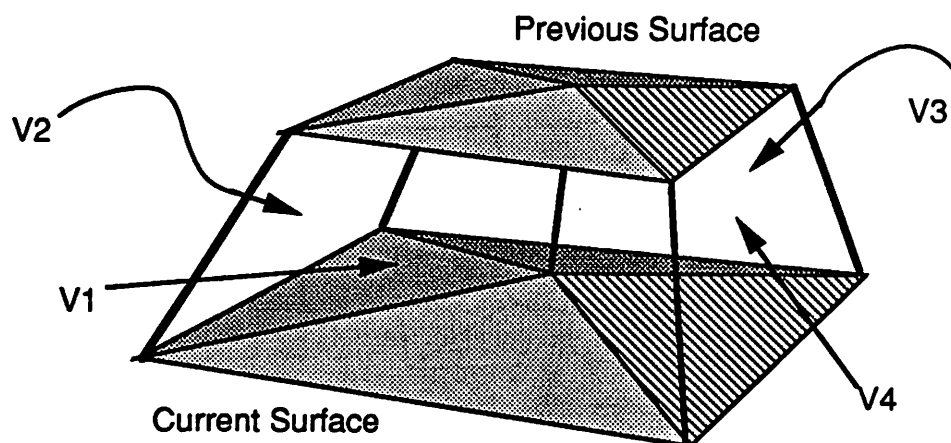


**Fig. 5.2:** The cell-based topography representation. Cell vertexes behind the surface are labeled "air", and those ahead are labeled "material." If a cell has both air, and material vertexes, it is a surface cell.

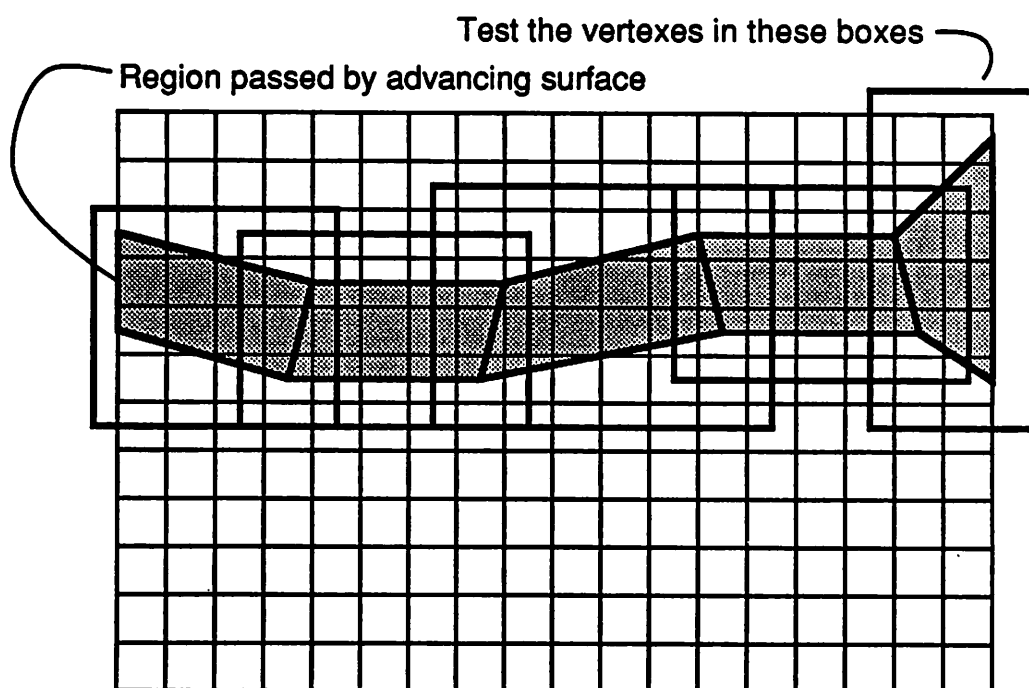
update algorithm described in this section. The concept behind the algorithm is straightforward. To determine whether a grid vertex has been passed by the surface the algorithm checks whether the vertex is between the current surface and the surface as it was at some previous time step. If it is between these surfaces the vertex is set to "air", otherwise the cell status is left unchanged. To test all the grid vertexes that may have been affected, the region between the two surfaces is divided up into several volume elements. These elements are constructed by connecting the triangular facets on the previous surface to the corresponding facets on the current surface as shown in Fig. 5.3. All the grid vertexes of interest are checked by testing the vertexes in the vicinity of each volume element. The procedure is illustrated schematically in 2D in Fig. 5.4.

The following algorithm is used to determine whether a point is contained in the five-sided volume element bounded by the two surfaces. First, if the point is outside the box bounding the inter-surface volume element it need not be tested. This check is implicit in the operation that determines which cells need to be tested against the volume element. Next a line  $\overline{AB}$  is defined that passes through the point  $P$  and the center  $C$  of one surface triangle as shown in Fig. 5.5. The ray from point  $P$  along line  $\overline{AB}$ , pointing away from point  $C$  is then tested to see if it intersects any of the other faces of the volume element. If it intersects any face, the point is inside. If it fails to intersect the four other faces, the point is outside the volume element. The ray-face intersection test can be performed using a slightly modified version of the algorithm described in section 4.6.3.

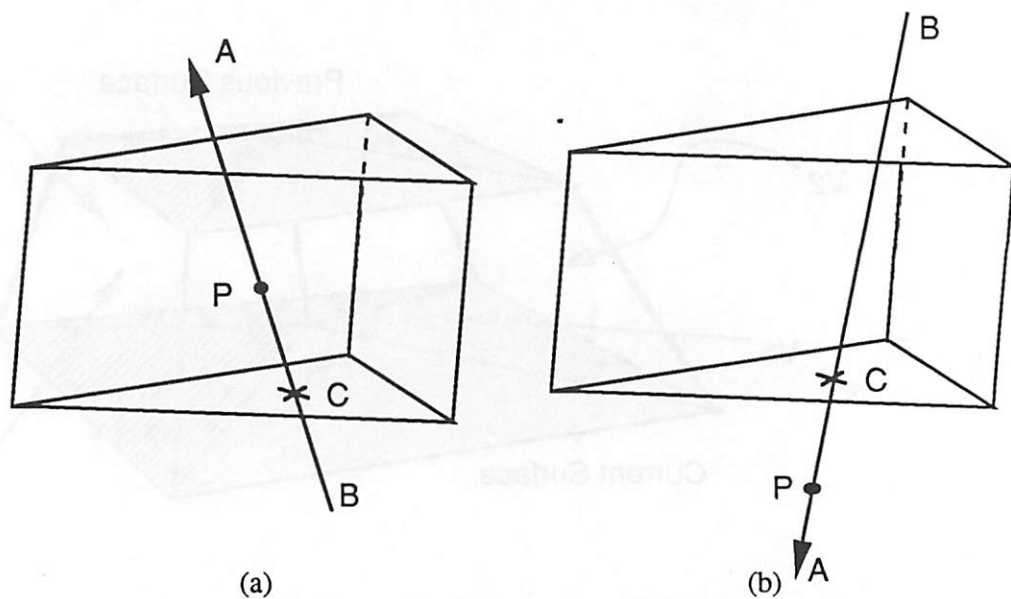
The surface cells should be updated whenever any facet travels a distance slightly greater than the length of a cell edge. It may not be necessary to test the cells every time step if the distance traveled is fairly small. However it is better to test the cells somewhat more frequently than may appear to be necessary in order to avoid the case shown in Fig. 5.6. In that



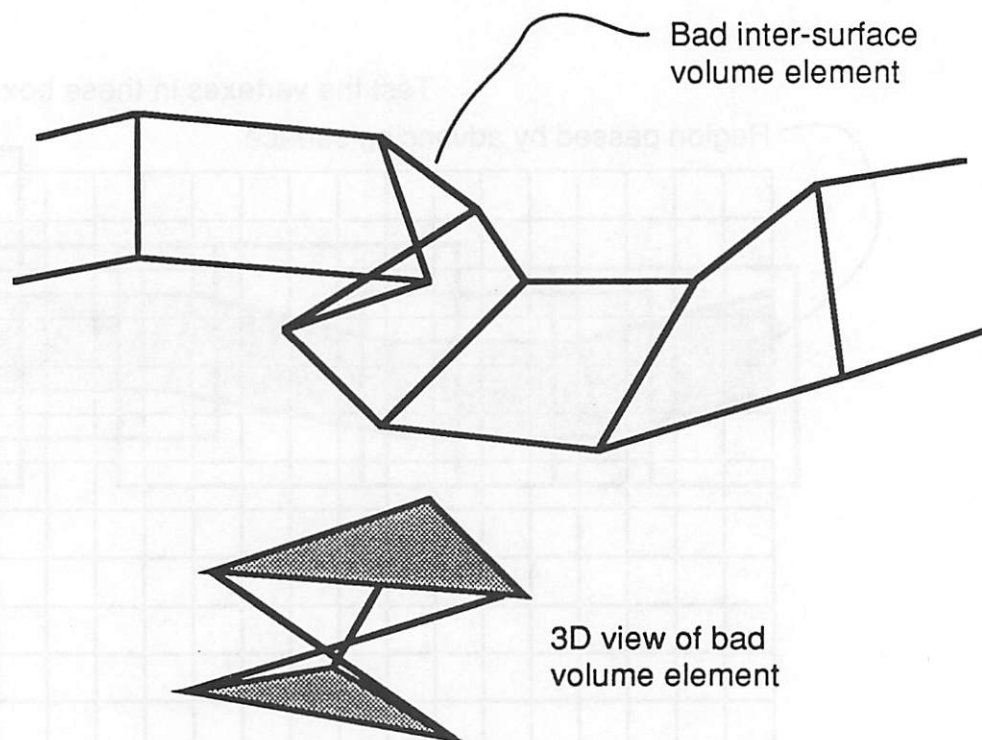
**Fig. 5.3:** Volume elements used to determine whether a grid vertex has been passed by the advancing surface. They are constructed by connecting the corresponding surface nodes.



**Fig. 5.4:** 2D view of the cell vertexes that need to be tested. The rectangular box bounding each volume element contains the cell vertexes that must be tested against that volume element.



**Fig. 5.5:** Geometry for determining whether a point is contained in the inter-surface volume element. (a) Point inside, ray PA intersects a face. (b) Point outside, ray PA does not intersect any faces.



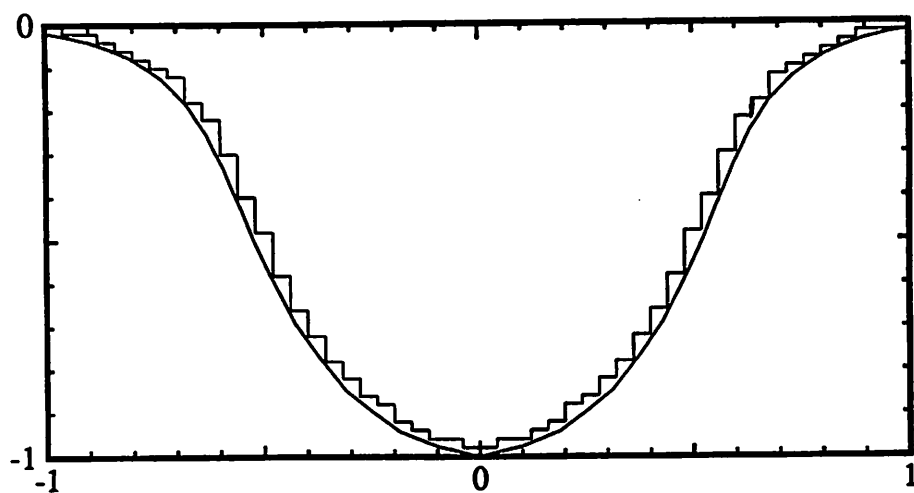
**Fig. 5.6:** Problem that may occur if grid vertexes are not updated frequently enough. Some volume elements may not be simple convex polyhedra.

figure, the surface has curved so rapidly that the volume element for the cell update test is not a simple convex polyhedron. The point inclusion test described above will not work for that case. It is also important to be sure that no point is converted to air and then back to material. Once it is determined that a point was passed by the advancing surface, that point becomes an air point regardless of its inclusion in other volume elements. This restriction also speeds up the algorithm since it is not necessary to retest a grid point once its status has been set to air.

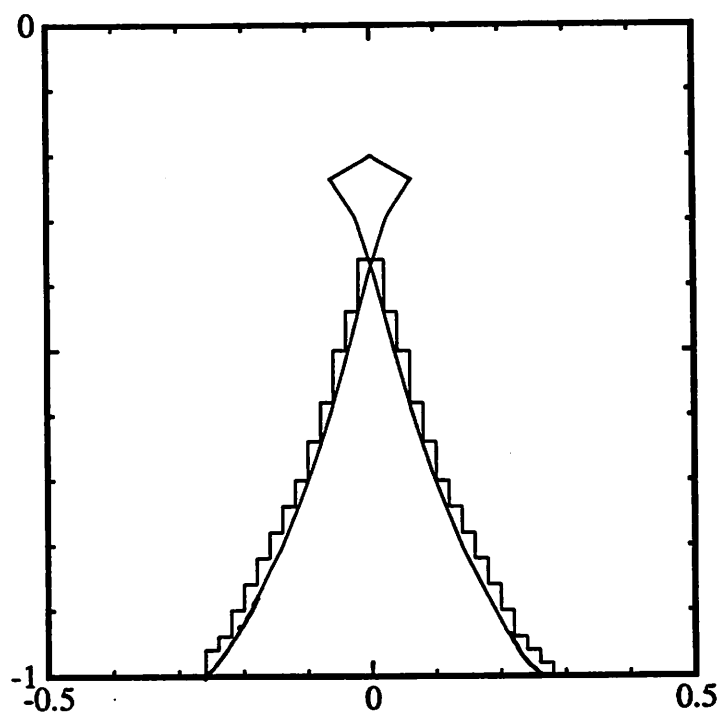
The cell update described above is designed for etching simulation. It can be converted for use in deposition simulation by reversing the roles of material and air grid points. A process that includes both deposition and etching simultaneously can be modeled by breaking it up into an etching step followed by a deposition step for each time step and handling the cell update differently for each case.

The CPU time for this algorithm is proportional to the number of grid point inclusion tests that must be made. For a fixed grid, this is roughly proportional to the number of surface facets at a time step. Therefore, the algorithm runs in  $O(M)$  time at each time step it is called, where  $M$  is the number of surface triangles.

The cell update algorithm is easy to test, by simply comparing the boundary of the air cells with the material surface. Some 2D cross-sections are given in Fig. 5.7. In Fig. 5.7a, the cells are compared against a Gaussian etch front like that in section 4.7.2. 100 time steps were used, and the 50x50x50 cell array was updated 50 times. The total CPU time was 126 seconds with 81% of that time spent performing the cell update. Doubling the initial number of surface triangles resulted in a total CPU time of 174 seconds with 72% of the time spent in the update tests. Fig. 5.7b compares the cells with a triangular rate function like that of section 4.7.3. In this figure, the ray-trace algorithm is used showing loop formation. The cell surface



(a)



(b)

**Fig. 5.7:** Evaluation of cell update algorithm by comparing border of air-cells with surface cross-section. (a) Comparison against gaussian etch rate function. (b) Comparison against triangular etch rate function with loop formation.



does not contain the loop, since those cells were already converted to air before the loop began forming. Here, there were 100 time steps and 50 cell array updates. The total CPU time was 97 seconds with 82% of the time spent performing the update tests.

The main drawback of the cells is that they require additional memory allocation. Fortunately, the amount of memory required is only on the order of a few bytes per cell, depending on how many variables are maintained in each cell. A 100x100x100 array can be allocated on most engineering workstations without difficulty. This size is sufficient for most simulations, since the cell linear density need not be much more than twice the surface mesh linear density (*i.e.*, for a 20x20 initial mesh, 50x50x50 cells is sufficient.) The cell update also introduces a noticeable amount of computational overhead. The important fact however is that this overhead increases no more than linearly with the surface area. If the surface mesh is very dense, the cells must also be quite fine, however as the surface area expands, the cell size may remain constant. Above all, the updated cells allow very efficient visibility calculation thus the computational overhead is easily justified.

## 5.5. CELL BASED GEOMETRIC OPERATIONS

Once the cells are in place, several important geometric operations can be performed with a minimum of computation time. All of these operations derive their efficiency from the ease with which an  $(x, y, z)$ -coordinate can be translated to an array address, which then allows the program to access information about the cell status.

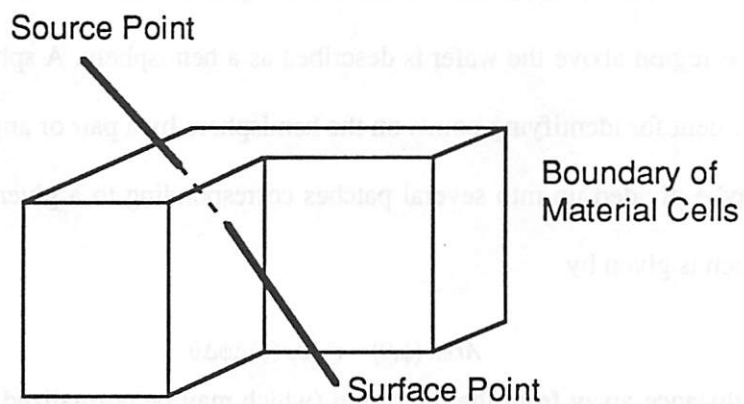
### 5.5.1. Efficient Shadow Test

First, points on facets which are oriented away from the illumination source are self-shadowed. Also, any point on the highest part of the surface cannot be shadowed. To see whether any other surface point is shadowed requires nothing more than testing the cells along a line from the point to a source point as shown in Fig. 5.8. If any of the cells contains material, the point is shadowed. The longest test occurs when no shadow exists, and every cell along the line is checked. The very worst case occurs when the surface point is in one corner of the simulation region and the source point is in the opposite corner. For a  $K \times K \times K$  cell array, at most a constant  $\sqrt{3}K$  point-in-cell tests are necessary. Thus, for a surface with  $N$  points, all the shadowed points can be found in  $O(N)$  time. Points on the top of the surface, or on nearly flat surfaces may avoid the shadow test entirely, for additional computational efficiency.

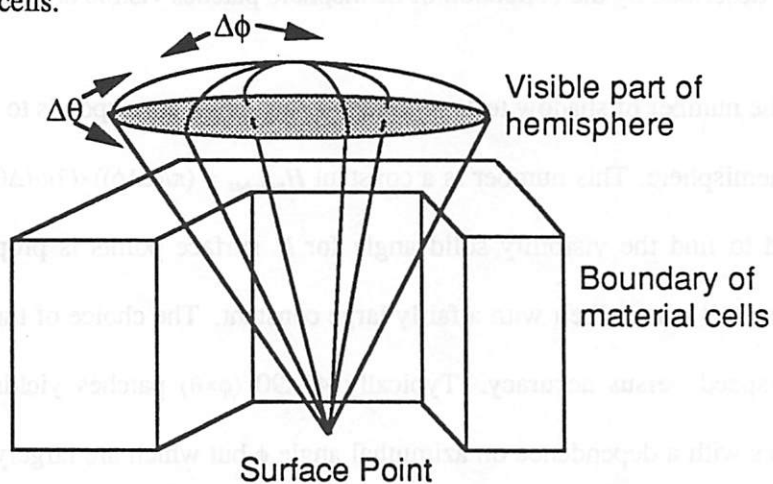
It might be argued that the shadow found by comparing against cells is not as accurate as one found by tracing rays through the surface mesh triangles. This point is no doubt true, however, the error introduced by the cells is small when compared with the discretization error introduced by having a finite number of points to describe the surface. Only a relatively small number of points are close to the shadow border. There may be some inaccuracy in the shadow edge, but the number of surface points which are incorrectly determined to be shadowed is very small, and in no case are those points very far from the actual shadow edge.

### 5.5.2. Efficient Solid Angle Visibility

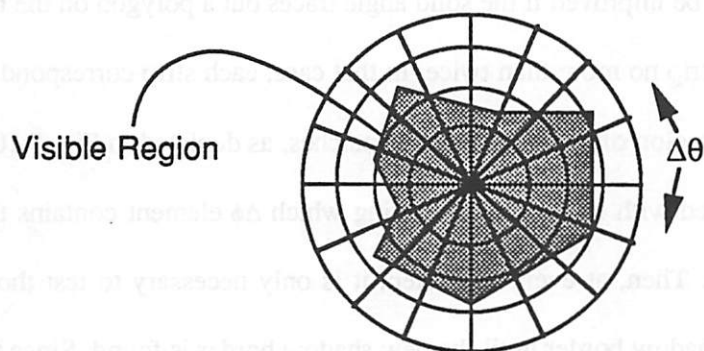
The problem of whether a certain patch of sky is visible from a viewing point on the ground can be reduced to a series of shadow tests. The sky is divided up into several patches



**Fig. 5.8:** Testing whether a point is shadowed by tracing a line from the point to an illumination source and determining whether that line intersects any material cells.



**Fig. 5.9:** Solid angle visibility test: determine how many patches on the hemisphere above the wafer are visible at a given point by applying multiple shadow tests.



**Fig. 5.10:** A top view of the hemisphere showing how the number of shadow tests can be reduced if the solid angle traces out a simple convex polygon on the hemisphere. Only the border at each  $\Delta\theta$  strip need be tested.

and a shadow test is made to determine which of those patches are visible. Fig. 5.9 illustrates the concept. The region above the wafer is described as a hemisphere. A spherical coordinate system is convenient for identifying points on the hemisphere by a pair of angles  $\phi$  and  $\theta$ . The hemisphere may be divided up into several patches corresponding to a given  $\Delta\phi$  and  $\Delta\theta$ . The area of each patch is given by

$$Area(\phi, \theta) = r \cdot \cos(\phi) \Delta\phi \Delta\theta \quad (5.2)$$

where  $r$  is the distance away from the the origin (which may be normalized to 1.) The origin of the hemisphere is simply the observation point on the surface. The solid angle visible at any point is described by the collection of hemisphere patches visible at that point.

The number of shadow tests required at each point corresponds to the number of patches in the hemisphere. This number is a constant  $H_{patches} = (\pi/(2\Delta\phi)) \times (2\pi/(\Delta\theta))$ . Therefore, the time required to find the visibility solid angle for  $N$  surface points is proportional to  $H_{patches} \cdot N$ , which is  $O(N)$  time albeit with a fairly large constant. The choice of the number of patches is one of speed versus accuracy. Typically  $45 \times 90$  ( $\phi \times \theta$ ) patches yields high accuracy. For processes with a dependence on azimuthal angle  $\phi$  but which are largely symetric for various  $\theta$  directions,  $45 \times 10$  is sufficient, yielding a factor of 9 run time improvement.

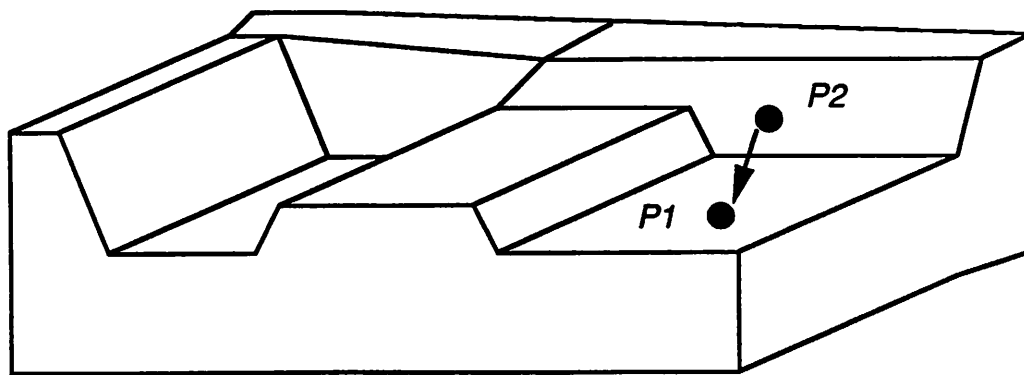
The CPU time can be improved if the solid angle traces out a polygon on the hemisphere which crosses each  $\Delta\theta$  strip no more than twice. In that case, each strip corresponding to a  $\Delta\theta$  increment has only one region of connected visible patches, as depicted in Fig. 5.10. An additional array may be stored with each point indicating which  $\Delta\phi$  element contains the shadow border for each  $\Delta\theta$  strip. Then, at every time step, it is only necessary to test those patches closest to the previous shadow border until the new shadow border is found. Since the surface shape changes incrementally, the shadow border only changes a small amount with each time step. Thus, it is likely that the shadow border will move no more than a few  $\Delta\phi$  at a time. The

total number of tests to determine the solid angle at a point is now close to  $2\pi/\Delta\theta$ . For a hemisphere with  $180 \times 90$  patches, the CPU time is improved nearly a hundredfold. This short-cut may not be taken if the solid angle border crosses over a  $\Delta\theta$  strip three or more times, but it may be used for the vast majority of structures common to process simulation such as contact holes, and trenches.

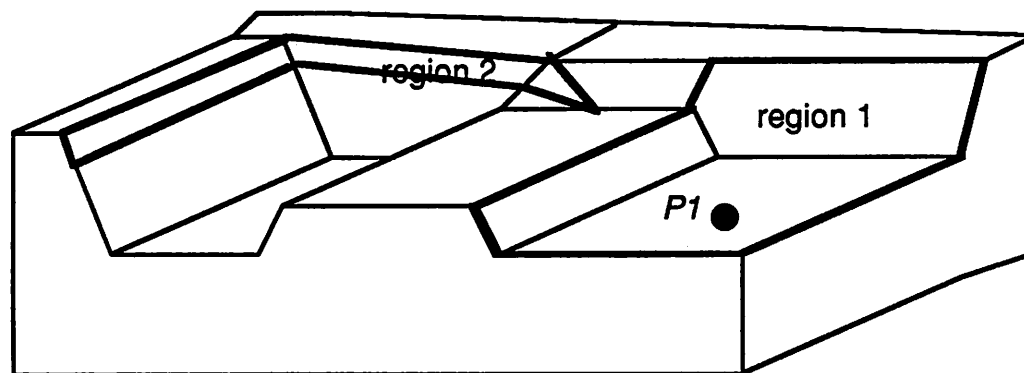
Once the solid angle is known, it can easily be used as a region of integration. For example, the total flux of material arriving at some surface point, is simply the integral of the flux distribution over the visible solid angle. The average flux times the area of each visible patch can be summed. The smaller the patches, the more accurate the integration as in any Riemann sum. This operation will prove extremely useful in both etch and deposition modeling.

### 5.5.3. Surface Reflection

Sometimes it is necessary to consider the reflection of light or particles off of the surface. The reflected particles may land somewhere else on the surface, affecting the local etch or deposition rate. Fig. 5.11a shows how this effect may be calculated. First, the number of directly incident particles (or the light intensity) must be calculated for every surface point. Then starting from a point of interest  $P1$ , a neighbor point  $P2$  is tested to see if it has a line of sight to the first point. If  $P2$  is visible from  $P1$  then the amount of material reflected from  $P2$  back to  $P1$  may be calculated. The exact amount is determined based on the length of the line between the two points and the orientation of the line with respect to the surface at the two points. In computer graphics, hyper-cosine functions have been used to represent the amount of light emanating from a non-perfect reflector.<sup>6</sup> In process simulation, the function giving the amount of material ejected at different directions from a surface point depends on the process and materials. The amount of material ejected is also related to the amount of material



(a)



(b)

**Fig. 5.11:** Surface reflection calculations. (a)  $P2$  reflecting back to  $P1$ . (b) Two non-connected regions visible from point  $P1$ .

coming into that point via a sticking coefficient.

Only a certain number of surface points will be visible from a given surface point. Obviously, it is desirable to reduce the number of point-to-point visibility test. There is no simple way to do this in general. Even for a simply connected surface (*i.e.* one with no holes or tunnels) there may be several regions visible from a point which are not connected. This occurrence is illustrated in Fig. 5.11, in which two disconnected regions are visible from point *P1*. To locate all the visible regions, every surface point must be compared with every other point, requiring  $O(N^2)$  time.

The only way to improve the algorithm time is to allocate memory at each point to store lists of pointers to the borders of visible regions. Only the border need be updated at each time step resulting in  $O(N \cdot B)$  time to test the entire surface, where  $B$  is the number of border points. Unfortunately, this requires  $O(N \cdot B)$  memory. For  $N = 10,000$  points, where  $B$  is approximately  $\sqrt{N} = 100$ , this is 1,000,000 elements of storage, each requiring several (perhaps 10-20) bytes of memory. Also there is the problem of determining when disconnected visible regions merge, or when connected regions split up. In any event, the reflection problem cannot be reduced much beyond  $O(N^2)$  or  $O(N^{1.5})$  time (assuming  $B \approx \sqrt{N}$ ), since no matter how the visibility is determined, the contribution of reflection from all the visible surface points must still be calculated. One simplification of the problem is to express the amount of reflected material arriving at a point as dependent only on the non-visible part of the hemisphere above the surface. This assumption gives an estimate of the actual reflecting surface area, and is reasonable if the reflection completely randomizes the direction in which particles travel. An example of this simplification, which is the only reflection algorithm currently implemented in SAMPLE-3D, is given in chapter 6.

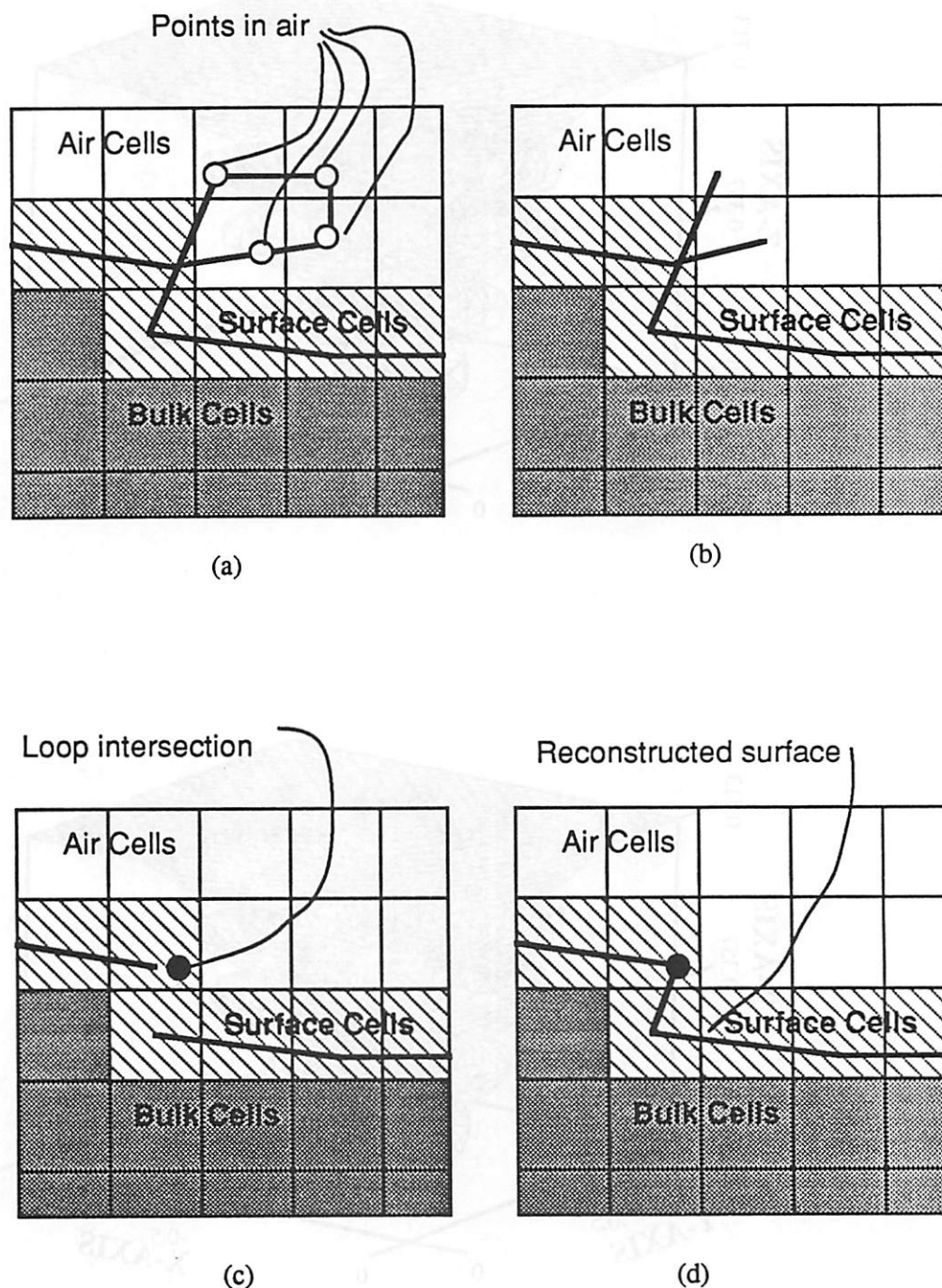
#### 5.5.4. Loop Identification

Loop identification was already hinted at in Fig. 5.7b. A loop may be defined as any region of points on the surface mesh that have traveled back into air cells. Care must be taken that the valid surface is contained within the air-cell/surface-cell boundary, but this was already done in the basic cell update algorithm. Loop detection is now simply a matter of scanning the surface to see which points are in air. The entire surface may be tested in  $O(N)$  time for  $N$  surface points.

Once the loop is found, like that in Fig. 5.12a, it must be removed. Loop removal is achieved in SAMPLE-3D by merging all the segments that are contained fully within the loop, until the loop is reduced to only those triangles that have at least one valid point as shown in Fig. 5.12b. Unfortunately, this leaves a ragged edge, and does not fully eliminate the loop. To completely restore the surface, the intersection at the loop edge must be determined. Then all the triangles that have any loop points should be removed as depicted in Fig. 5.12c. Finally, the surface mesh data structure must be reconstructed to close the gap in the surface. The intersection line which was once the loop boundary is now part of the new surface illustrated in Fig. 5.12d.

In theory loop removal is not difficult. In practice however, it is very challenging to find a way to reconstruct the surface mesh data structure correctly. Fig. 5.13. shows loop removal for ray-trace lithography simulation using the method of Fig. 5.12b, demonstrating that it can be done. Nevertheless, there is still room for more work in this important topic.





**Fig. 5.12:** Loop identification and removal. (a) Loop location with the cells. (b) Loop removal by deleting all triangles with three points in air. (c) Locating the loop intersections. (d) Full 3D loop removal.

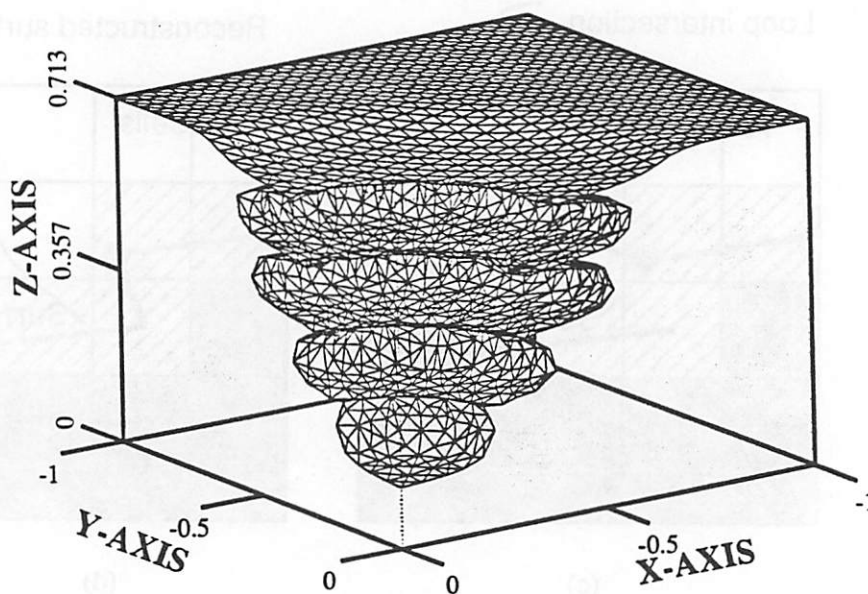
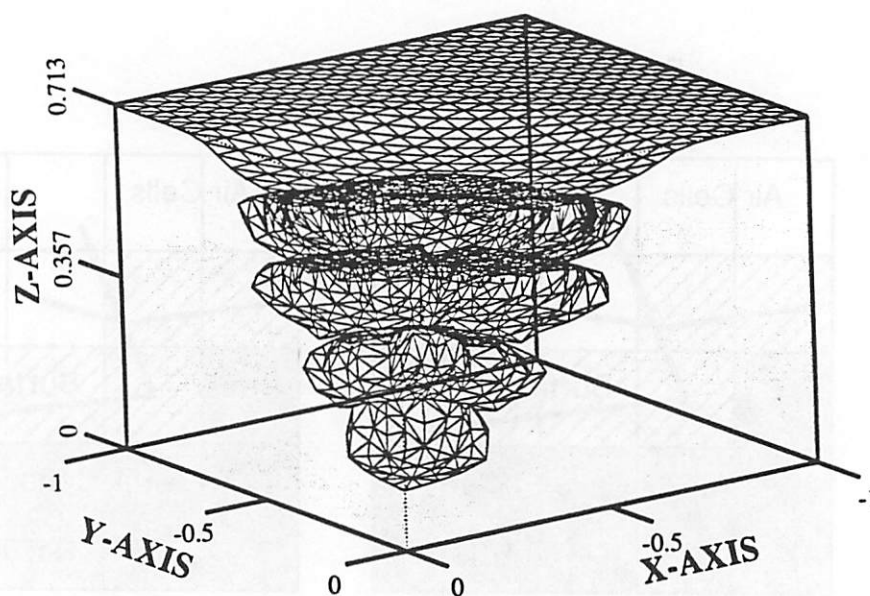


Fig. 5.13: Loop detection and removal in lithography simulation. The above plot is with no loop removal, below is with loop removal.

## 5.6. SHADOW AND SOLID ANGLE PERFORMANCE EVALUATION

Verification of the algorithms for shadow detection and solid angle visibility is straightforward. Fig. 5.14 shows a shadow cast in a  $1.0 \times 1.0 \times 0.5 \mu\text{m}^3$  contact hole. The light source is directed along  $\phi=45^\circ$  and  $\theta=45^\circ$ . The plot shows all triangles with three shadowed vertexes as dark, and all other triangles as light. The time required to calculate the shadow was less than a fraction of a second.

Fig 5.15 shows a solid angle calculation for a surface calculated using a Gaussian etch rate. The example assumes a flux distribution given by  $\cos\phi$ . The shading shows the relative total flux received at each facet, by integrating the flux distribution over the visible angle at each point, and then averaging over the three triangle vertexes. In the bottom of the feature, the visibility is most restricted and the shading is darkest.

Table 5.1: Solid Angle Calculation Times						
Cells per Micron	$N_\phi \times N_\theta$	# Pts	# Tris	# Shadow Tests	Solid Angle CPU Time(s)	Cell Time (s)
50	45x10	936	1810	118492	11.4	17.9
50	45x10	1536	2994	208089	19.8	22.9
50	45x10	3888	7658	559678	53.8	48.6
50	45x90	936	1810	1003227	104.0	17.9
50	45x90	1536	2994	1659490	170.2	22.9
50	45x90	3888	7658	4346601	430.4	48.6
25	45x10	936	1810	105042	7.0	9.5
25	45x10	1536	2994	185181	12.4	14.8
50	10x10	936	1810	36818	5.1	17.9
50	10x10	1536	2994	60420	8.2	22.5

The simulation run times for optimized C code running on an IBM RS6000/530 workstation are summarized in the above table, which gives CPU times to calculate integrals over visibility solid angles for square contact holes like that in Fig. 5.14. The solid angle time is that to integrate over the visible solid angle for all the surface points point at one time step. The

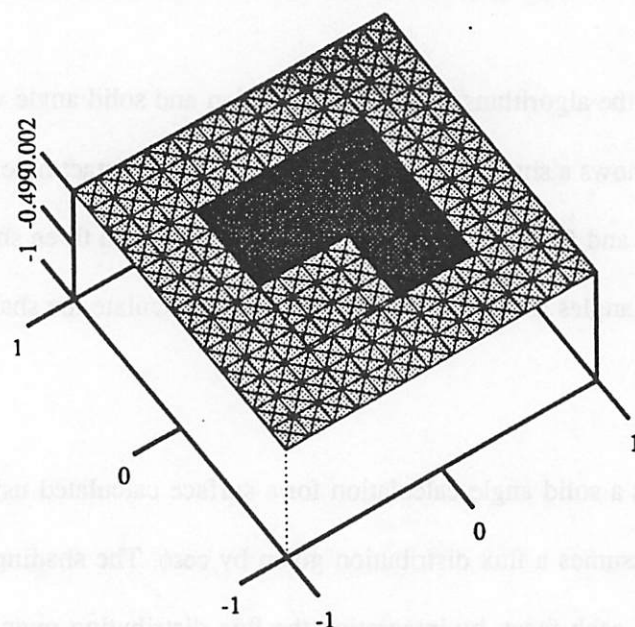


Fig. 5.14: Shadowing of a square contact hole. The light source is along  $\theta=45^\circ$ ,  $\phi=45^\circ$ . Triangles with 3 shadowed vertices are dark.

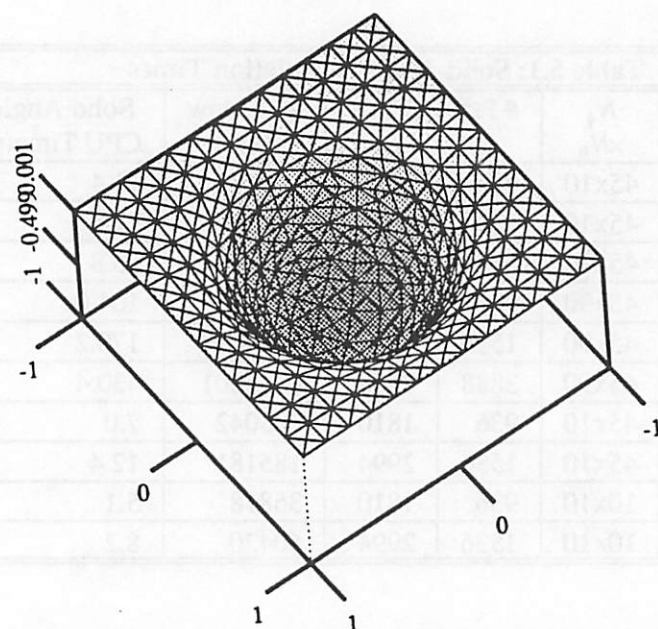


Fig. 5.15: Relative incoming flux for a surface generated using a gaussian etch rate. The lightest triangles see the widest visibility solid angle. The darkest triangles have the most restricted visibility.

time to set the cells is the total time to advance an initially flat surface to a  $1.0 \times 1.0 \times 0.5 \text{ m}^3$  hole and update all the cells. The time to calculate the solid angle is almost exactly  $0.013s \times (\# \text{ Points})$  for  $50 \times 50 \times 50$  cells, and  $45 \times 10$  hemisphere patches. The run time scales almost linearly with the number of  $\theta$  elements. The time to update the cell array while the hole was created increases more slowly than the rate at which the number of surface triangles increases, since the total number of cells remains constant. This results in fewer cells that need to be checked for each triangle. Reducing the number of cells per micron reduces the CPU time since each shadow test is faster. Fewer cells and fewer patches degrades the accuracy of the calculation, but roughly 50 cells per micron and  $45 \times 10$  hemisphere patches gives good results with efficient run times.

## 5.7. COMMENTARY ON THE SURFACE-CELL HYBRID METHOD

It is difficult to imagine a combination of data structures and algorithms which can perform the above operations in a time faster than  $O(N)$  where  $N$  represents the surface area (points, or triangles). There are also some hidden benefits. For example, the cells may be used to store material information such as local density, which might vary in deposition as the surface orientation changes. However there are a few areas that may yet be improved. The overall memory usage may be reduced by a clever allocation of the memory used to describe the surface cells. The greatest density of cells is only needed at the surface. Elsewhere, relatively coarse cells are sufficient. An octree-based approach comes to mind. With an octree, the overall storage may be reduced to about  $O(K^2)$  instead of  $O(K^3)$  where  $K$  is the number of cells along one axis of the simulation region.<sup>7</sup> A preliminary implementation in SAMPLE-3D by J. Helmsen supports this idea.<sup>8</sup> The time to perform the individual shadow and solid angle tests may also be improved by the octree surface representation since a block of air cells

would be replaced by one octree leaf, thus reducing the number of tests necessary to see if a point on the shadow line hits a material cell. Improvements here would benefit a variety of research efforts around the world, since it is not dependent on the cell or surface-based nature of the topography representation.<sup>9</sup>

It may also be desirable to increase the accuracy even further. The cell or octree data structure could be used to locate the exact triangles casting the shadow. This information could then be used to insert points in the surface mesh along the shadow boundary. It might also be possible to keep track of the shadow boundary as it varies with respect to a changing shadow casting edge.

It is also possible to keep track of the solid angle border seen by each surface point. this is an extension of the method used in section 5.5.2 to improve the solid angle evaluation time. By maintaining a list describing the actual solid angle border, the restriction that the border must not cross a  $\Delta\theta$  strip more than twice is removed. This method has the potential to preserve the efficiency achieved by the algorithm of section 5.5.2 but with added geometric generality.

The surface-cell hybrid data representation and algorithms presented in this chapter are an effective way to answer several important questions in 3D topography simulation. One way of looking at the surface-cell hybrid data representation combined with the facet-motion algorithm is to view the whole thing as a volume removal algorithm for topography simulation that accurately calculates the amount of material removed based on the correct surface orientation. In this way, the advantages of the cell-based approaches described in chapter 3 are preserved with the added benefit of accuracy and generality provided by the facet-motion algorithm of chapter 4.

## REFERENCES

1. E. Haines, "Essential Ray Tracing Algorithms," in *An Introduction to Ray Tracing*, ed. A.S. Glassner, pp. 33-77, Academic Press, London, 1989.
2. M.E. Newell, R.G. Newell, and T.L. Sancha, "A solution to the Hidden Surface Problem," *Proceedings of the ACM National Conference 1972*, pp. 443-450, 1972.
3. C.A.R. Hoare, "Quicksort," *Computer Journal*, vol. 5, pp. 10-15, 1962.
4. I. Garganti, T. Walsh, and O. Wu, "Viewing Transformations of Voxel-Based Objects via Linear Octrees," *IEEE Computer Graphics and Applications*, vol. 6, no. 10, pp. 12-21, October 1986.
5. S.M. Rubin and T. Whitted, "A 3-Dimensional Representation for Fast Rendering of Complex Scenes," *Proceedings of SIGGRAPH '80, in Computer Graphics*, vol. 14, no. 3, pp. 110-116, July 1980.
6. Bui-Tuong Phong, "Illumination for Computer Generated Pictures," *Communications of the ACM*, vol. 18, no. 6, pp. 311-317, June 1975.
7. G.M. Hunter, "Efficient Computation and Data Structures for Graphics," Ph.D. Thesis, Department of Electrical Engineering and Computer Science, Princeton University, Princeton, NJ, 1978.
8. J. Helmsen, *Personal Communication*, Berkeley, CA, July 1991.
9. M. Fujinaga, *Personal Communication*, June 1991.

## CHAPTER 6

### THREE-DIMENSIONAL ETCH MODELING AND SIMULATION

*The old order changeth,  
yielding place to the new.*  
Alfred Lord Tennyson

#### 6.1. GEOMETRIC ETCH MODELING FOR PROCESS SIMULATION

Chapters 4 and 5 present efficient surface evolution algorithms using the motion of triangular facets and efficient geometric operations using a surface-cell hybrid data structure. Until now, only cursory attention has been paid to the physical process models for moving the facets. This was done deliberately in order not to restrict the algorithms to a narrow class of physical models. There is still much debate as to the exact physical mechanisms responsible for observed phenomena. The algorithms and data structures developed thus far support a wide variety of possible surface evolution mechanisms. This chapter develops mathematical process models for etching simulation based on postulated and measured physical effects which have been reported in the IC fabrication process literature. The algorithms of the previous chapters for directionally dependent surface advancement, shadowing, solid angle visibility, and surface reflection support the coupling of process physics and surface topography and are used to simulate the time evolution of 3D surfaces with the etch process models.

This chapter discusses etch process simulation from the perspective of physical models and seeks to address several questions. How do surface geometries and material properties interact in etch process simulation? How do the surface advancement and geometric visibility algorithms developed thus far allow the simulation of a given process model? Are there any additional factors of importance which must be considered? The philosophy is to formulate geometric etch models in a general way. The models take the form of general mathematical



expressions, which are independent of the algorithms used to evaluate the expressions. Physical interactions enter into the models as parameters with a dependence on the surface materials and topography. The significant body of literature in etch modeling provides the basis for the general 3D models presented in this chapter.

Many previous models for surface-orientation dependent etching begin with basic principles and then proceed to develop mathematical expressions for the continuous time evolution of a 2D surface profile.<sup>1, 2, 3, 4, 5</sup> One model also considered 3D ion milling.<sup>6</sup> These formulations often depend on an analytic expression of the sputter yield vs. angle or etch rate vs. angle. A full analytic formulation of the profile evolution has the advantage that predictions about the formation of corners and certain geometries may be made; however, it has the disadvantage that it is difficult to introduce new rate vs. angle expressions. It is also difficult to develop an analytic 3D formulation for general visibility solid angles. Alternatively, simulation performs numerical integrations of arbitrary flux distributions over arbitrary visibility solid angles. A wide range of possible theoretical or empirical physical relationships may be incorporated into the simulator which can then be applied to evolving surface geometries. Predictions from fully analytic formulations for certain simple cases may be compared with simulation results, to verify the simulation validity.

This chapter uses a series of examples to illustrate the development of physical etch process models and their simulation using surface advancement and visibility algorithms. A general etch rate formulation based on sputter yield and particle flux distributions at the surface is presented to model plasma etch processes, including reactive sputter etching and ion milling. Extensions to the basic model, including consideration of surface kinetics based on the equilibrium between adsorption, desorption and surface diffusion in plasma etching are discussed. Directionally dependent crystal etching is also considered. Simulation results are presented to

demonstrate the applicability of the algorithms to a variety of etch models.

## **6.2. A BASIC MODEL FOR PLASMA ETCHING AND ION MILLING**

### **6.2.1. Factors of Importance In Glow Discharge Etching**

Many factors contribute to the final shape of etched surfaces in glow discharge pattern transfer processes. These factors have been summarized succinctly by Jurgensen and Shaqfeh:<sup>7</sup> (1) the energy and angular distributions of the energetic particles incident on the surface, (2) the energy and angle dependences of the sputter yield (which is the ratio of particles removed to incident particles), (3) deposition of polymers and redeposition of sputtered material, (4) sticking coefficients and reaction probabilities of radicals, (5) glancing angle reflection of energetic particles, (6) neutral to ion flux ratios, and (7) ion trajectory variation due to local electric fields in the vicinity of the feature. Work by McVittie *et. al.* has also reinforced the importance of remitted particles and redeposited material.<sup>8, 9</sup> Additionally, a model for surface diffusion effects in plasma etching of silicon has been reported recently.<sup>10</sup>

There is still debate as to which physical factors contribute the most to the final profile. Simulation with various physical assumptions allows the theories to be compared with experimental evidence, and possibly validated. One approach to considering most (if not all) of these possible physical factors is to develop a generalized model with multiple terms that may be included or ignored depending on the particular processing environment. One feature that most dry etching models include is a dependence on the flux distribution of incident ions and neutral particles. Consideration of flux distributions is a good starting place for building a generalized model. Additional terms and factors may be included given different physical hypotheses.

The original etch simulation work in SAMPLE<sup>11</sup> used the vector sum of isotropic, directional, and ion milling etch components. Tazawa *et. al.* developed a unified model for the 2D etch simulator DEER<sup>12</sup> that built on this approach by considering four main components: (1) an isotropic rate, (2) directional etching due to directly incident ions, (3) etching due to directly incident neutral particles, and (4) etching or redeposition due to reflected particles. All of these effects were then related to the motion of points along the local surface normal vectors. Each component contributes to the velocity in the normal direction according to an angular flux distribution of incident particles  $f(\psi)$ , and a reaction velocity function  $g(\theta)$  where  $\psi$  and  $\theta$  are measured with respect to the wafer normal direction. The velocity along the surface normal is then expressed as:

$$v(\theta) = R \int f(\psi) g(\psi - \theta) d\psi \quad (6.1)$$

For example, in isotropic etching  $f = 1$ ,  $g = 1$ , and  $R$  gives the local etch rate. This expression is easily generalizable to three dimensions, and can be extended to include additional physical phenomena.

### 6.2.2. Theoretical Basis for Etch Velocity Along the Surface Normal

Before developing geometric models for the various etch rate components, the validity of relating orientation dependent etching to a velocity along the surface normal must be established. A surface evolution equation from fluid mechanics provides a convenient starting point.<sup>13</sup> If a surface is defined as  $F(x, t) = 0$  then the interface evolution may be given as

$$\frac{\partial F}{\partial t} + \mathbf{v} \cdot \nabla F = 0 \quad (6.2)$$

where  $\mathbf{v}$  is the interface velocity vector.  $F$  is a constant in a reference frame moving with the interface velocity. This equation is valid regardless of the velocity direction with respect to the surface normal  $\mathbf{n} \equiv \nabla F / |\nabla F|$ . Therefore, it is acceptable to consider velocity along the

interface normal.

The argument is further strengthened by considering the seminal paper presented by Frank in 1958 on crystal growth and dissolution.<sup>14</sup> In it Frank proved that for orientation dependent etching (or deposition) surface points moved along straight line trajectories. If the etch rate is given along the actual crystal normal then the trajectory of a point is parallel to the normal to the polar diagram of the reciprocal of the etch rate at the point of corresponding orientation. The result of this theorem is that the surface orientation will change as the surface evolves, but considering the growth to follow a path along the surface normal still gives the correct result since the change in surface orientation will alter the rate. The theorem was derived using crystal structures as a starting point, but it can be applied to any orientation dependent process. Barber *et. al.* showed that Frank's theory could be applied equally well to ion milling as to chemical etching.<sup>2</sup>

Strong mathematical arguments exist to support the hypothesis that etching processes may be modeled by velocity along a surface normal. It should be noted that the facet-motion algorithm of chapter 4 does NOT require that the facets be moved along their normal directions, however in practice it is easier to properly calculate the intersection of colliding facets and to locate the fastest and slowest directions of travel at convex and concave surface nodes if facets move normal to themselves.

### 6.2.3. The Foundation of the 3D Unified Model

Some basic assumptions are made in order to develop the generalized model: (1) particle trajectories are assumed to be linear on the length scale of the features being etched, and (2) the sputter yield is separable into functions of incident angle and energy. The geometry is

shown in Fig. 6.1 where  $\phi$  and  $\theta$  are angles in spherical coordinates and  $(\phi_x, \theta_x)$  gives the surface orientation at point  $\mathbf{x}$ . A unit direction given by the spherical coordinate angles may be converted to the Cartesian unit directions by:

$$\mathbf{r}_{(\phi, \theta)} = |\mathbf{r}| (\cos\theta \sin\phi \mathbf{i} + \sin\theta \sin\phi \mathbf{j} + \cos\phi \mathbf{k}) \quad (6.3)$$

since the  $z$ -axis is perpendicular to the wafer plane. It is also convenient to define an angle  $\alpha$  between an arbitrary direction and the surface normal. The cosine of this angle is given by:

$$\cos\alpha = \cos\theta_x \sin\phi_x \cos\theta \sin\phi + \sin\theta_x \sin\phi_x \sin\theta \sin\phi + \cos\phi_x \cos\phi \quad (6.4a)$$

which is reducible to:

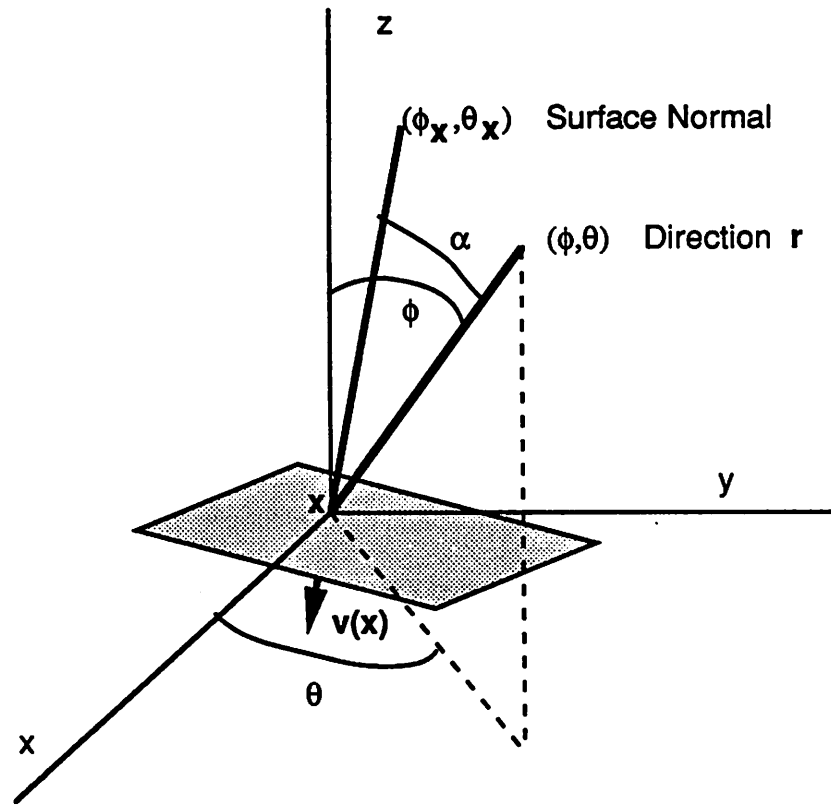
$$\cos\alpha = \cos\phi \cos\phi_x + \sin\phi \sin\phi_x \cos(\theta - \theta_x) \quad (6.4b)$$

Equation 6.1 is generalized to three dimensions as follows:

$$\mathbf{v}(\mathbf{x}) = R \iint_{\Omega(\mathbf{x})} f(\phi, \theta) \cdot g(\phi, \theta, \phi_x, \theta_x) \sin\phi d\phi d\theta \quad (6.5)$$

As in equation 6.1,  $f$  is the flux distribution of incident particles and reflected particles arriving at the point  $\mathbf{x}$ , and  $g$  is the function relating the differential rate along  $(\phi, \theta)$  to evolution along the surface normal, which may also include angular dependent sputter yield effects.  $\Omega(\mathbf{x})$  is the solid angle to the plasma visible from point  $\mathbf{x}$ . The differential solid angle element area is  $\sin\phi d\phi d\theta$ . The flux functions  $f$  and  $g$  are typically normalized so that  $R$  is the etch or deposition rate on a flat substrate with no shadowing.

At this point no restriction on the functions  $f$  and  $g$  is made, except that they must be defined over the  $4\pi$  solid angle. As will be seen, most models are independent of  $\theta$ . Also,  $\phi$  angles greater than  $\pi/2$  are not generally of interest, since that indicates a direction pointing below the wafer horizon. Now it is possible to develop specific expressions for the individual etch rate components.



**Fig. 6.1:** Coordinate system and angle definitions for geometry used in the 3D unified model for plasma etch simulation.

#### 6.2.4. Isotropic Etching In Plasma Processing

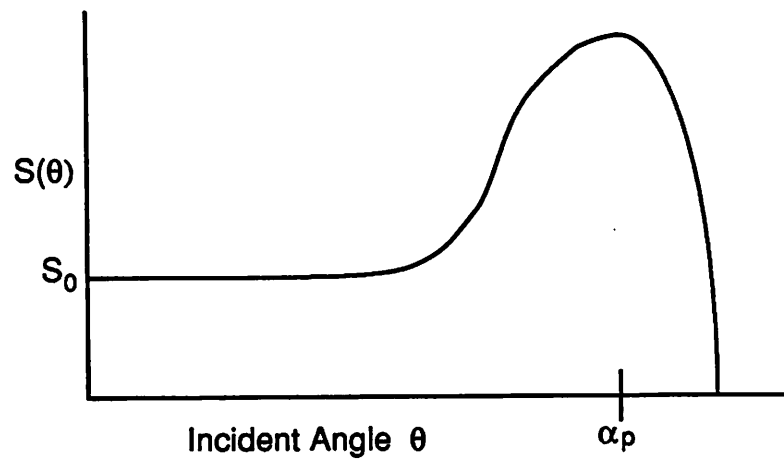
A glow discharge may produce a chemically reactive species from an inert molecular gas. This species then interacts chemically with the material surface. No physical momentum transfer is present so the process is isotropic for materials without orientation dependent properties. This component is included in the unified model as an etch rate along the normal direction that is independent of surface orientation or solid angle visibility. This component is expressed as a constant:

$$v_i = R_{is} \quad (6.6)$$

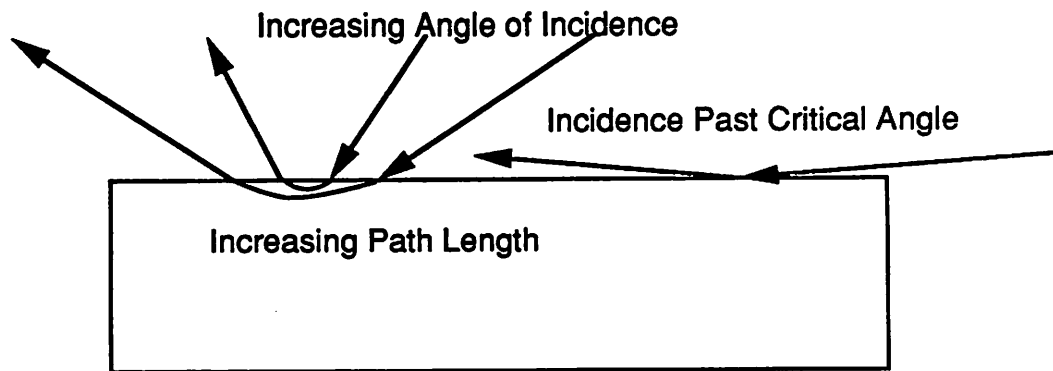
#### 6.2.5. Directly Incident Ions: Sputtering and Ion Milling

In reactive sputter etching and ion milling, energetic ions transfer momentum to the surface to either enhance the etch rate reactions or physically remove material. The sputter yield  $S(\alpha)$  is defined as the number of atoms ejected by each ion incident at an angle  $\alpha$  from the surface normal. The angular dependence of sputter yield was reported by H. Bach for quartz and other insulators.<sup>15</sup> The basic shape of the sputter yield vs. incident ion angle is shown in Fig. 6.2. For normal incidence, a certain sputter yield  $S_0$  is observed. As the angle increases, the sputter yield reaches a peak value and then suddenly drops off.

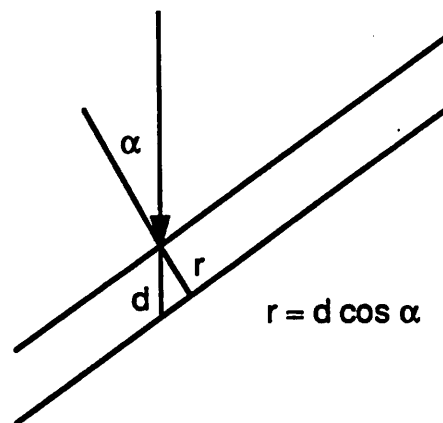
A theoretical argument for the sputter yield peak has been given by J. Lindhard.<sup>16</sup> The sputter yield is proportional to the energy deposited in the surface layer of a given depth, which is in turn related to the path length the ion travels in the material. As the angle increases, the path length increases as shown in Fig. 6.3 resulting in a higher deposited energy and a higher sputter yield. However, for very steep angles, the incident ion has an increased probability of reflection from the surface. After a certain critical angle, the yield drops off



**Fig. 6.2:** Basic shape of the sputter yield vs. incident angle curve.



**Fig. 6.3:** Increasing path length of ions in substrate material until critical angle is reached.



**Fig. 6.4:** Relation of depth of material removed along direction of incidence to etch distance along surface normal.



rapidly. Lindhard derived the critical angle  $\alpha_p$  with classical scattering theory on the assumption that the ion to surface atom potential follows an inverse square law:

$$\alpha_p = \frac{\pi}{2} - \sqrt{\frac{5\pi a_0^2 n^{2/3} Z_1 Z_2 E_R}{(Z_1^{2/3} + Z_2^{2/3})}} \quad (6.7)$$

where  $a_0$  is the Bohr radius 0.53 Å,  $Z_1$  and  $Z_2$  are the atomic numbers of the ion and surface atom,  $n$  is the density of atoms per unit volume  $E_R$  is the Rydberg energy 13.6 eV, and  $E_1$  is the ion energy.

Table 6.1: Sputter Yield Peak Angles: Theory and Experiment				
substrate material	incident ion	energy (keV)	measured peak angle (degrees)	theoretical peak peak (degrees)
Cu	Ar	27	83	83
Ni	H	4	85	85
Si	Ar	0.5	52	57
Si	Ar	1	65	66
SiO <sub>2</sub>	Ar	5.6	78	80
SiO <sub>2</sub>	Ar	32	82	86

In practice, the Lindhard theory gives a fairly good estimate for the peak angle. Table 6.1 lists some experimental peak angles for ions incident on various materials<sup>5, 17, 18, 19</sup> compared with theoretical values calculated using equation 6.7.

Sputter yield also has an energy dependence. Jurgensen and Shaqfeh have proposed that the angle dependence is separable from the energy dependence  $S(E, \phi, \theta) = S_{\phi, \theta}(\phi, \theta) \cdot S_E(E)$ .<sup>7</sup> Integrating the energy term over the energy distribution yields an energy independent factor that may be multiplied with  $S_{\phi, \theta}$ . This is expressed as:

$$S(\phi, \theta) = S_{\phi, \theta}(\phi, \theta) \int h(E; \phi, \theta) S_E(E) dE = S_{\phi, \theta}(\phi, \theta) \cdot \bar{S}_E(\phi, \theta) \quad (6.8)$$

where  $h(E; \phi, \theta)$  is the conditional energy distribution in spherical coordinates. The function  $h$  is determined by the collision processes in the plasma sheath and thus relates the surface evolution with the plasma sheath kinetic theory.

A variety of analytic functions have been proposed to fit the sputter yield curve.<sup>4, 5</sup>

Most use a series of sines or cosines raised to successive powers, for example:

$$S(\alpha) = \sum_{n=1}^m a_n \cos^n \alpha \quad (6.9)$$

where  $a_n$  are fitting parameters. For the treatment here, it does not matter what function is used to describe the sputter yield. All that matters is that the sputter yield function accurately describe the observed result.

The depth  $d$  of material removed from a plane surface is related to the sputter yield by:

$$d = \frac{\Phi t}{n} S(\alpha) \quad (6.10)$$

where  $\Phi$  is the number of ions per second striking the unit of area normal to the direction of incidence,  $n$  is the number of atoms per unit volume of the target material, and  $t$  is the bombardment time in seconds. Fig 6.4 shows how the depth along the incident ion direction direction is related to etching along the surface normal. Assuming that the incident ions all come from the same direction, (in which case the distribution function  $f = \delta(\phi, \theta)$ ), the velocity in the surface normal direction at  $\mathbf{x}$  is simply:

$$\mathbf{v}(\mathbf{x}) = \frac{\Phi}{n} \cdot S(\alpha) \cos \alpha \quad (6.11)$$

If the point is shadowed, the etch rate is 0. For ion milling, the ion mean free path length is long, and nearly all of the bombarding particles arrive along one direction. Variants of equation 6.11 have been successfully applied to 2D simulation of ion milling.<sup>20, 21</sup> In RSE processes some ions arrive from angles other than direct incidence, owing to the presence of collisions within the plasma. Monte Carlo simulation of ion transport in the plasma has produced plots of the ion flux vs. angle which show that most of the ions are directly incident, but some arrive at angles all the way out to about 60° incidence. The shape of the ion flux distribution is similar to a Gaussian with a very small standard deviation  $\sigma$ .<sup>22, 23</sup> Equation 6.11

should be modified to account for the angular extent of the distribution.

$$\mathbf{v}_{di}(\mathbf{x}) = \frac{R_{di}}{N_{di}} \iint_{\Omega(\mathbf{x})} F_i(\phi, \theta) \cdot S_i(\alpha) \cos \alpha \sin \phi d\phi d\theta \quad (6.12)$$

where  $F_i(\phi, \theta)$  gives the ion flux distribution over the solid angle of the visible plasma and  $N_{di}$  is a normalizing factor equal to the flux integrated over  $0 \leq \phi \leq \pi/2$  and  $0 \leq \theta \leq 2\pi$ . Here,  $S_i$  is a normalized yield function equal to  $S/S_0$ . The etch rate for a flat wafer with no shadowing is  $R_{di} = \Phi S_0/n$ .

For amorphous materials, the sputter yield depends only on the angle of incidence  $\alpha$ . However, for monocrystalline materials, the yield may vary with the crystal plane exposed.<sup>6</sup> The above equations can account for the dependence on  $\theta$  and  $\phi$  as well as  $\alpha$ , if the sputter yield curve is known for all planes.

#### 6.2.6. Directly Incident Neutral Particles

Neutral particles do not have a sharp peak in their flux distribution, since they are not affected by an electric field. Neutral particle distributions have also been derived by Monte Carlo simulation.<sup>22, 23</sup> Tazawa *et al.* used a hypercosine function to fit the distribution:

$$F(\phi, \theta) = \cos^m \phi \quad (6.13)$$

If the reaction proceeds in the incident direction, then the velocity due to neutral particles is:

$$\mathbf{v}_{dn}(\mathbf{x}) = \frac{R_{dn}}{N_{dn}} \iint_{\Omega(\mathbf{x})} F_n(\phi, \theta) \cdot \cos \alpha \sin \phi d\phi d\theta \quad (6.14)$$

where  $F_n(\phi, \theta)$  is the neutral flux distribution and  $N_{dn}$  is a normalization factor equal to the flux integrated over the entire hemisphere  $0 \leq \phi \leq \pi/2$  and  $0 \leq \theta \leq 2\pi$ .  $R_{dn}$  is a constant giving the etch rate on a flat wafer with no shadowing.  $R_{dn}$  is often modified by the presence of incident ions, surface damage, or passivating materials. Modifications to  $R_{dn}$  are presented further in section 6.3.

### 6.2.7. Indirectly Incident Etchant Flux

Indirectly incident particles include reflected and redeposited particles. Models have been developed to express these phenomena, for example work by Müller and Pelka.<sup>24</sup> A general expression to account for this effect is:

$$\mathbf{v}_{ii}(\mathbf{x}) = R_{ii} \iint_{H - \Omega(\mathbf{x})} F_{ii}(\phi, \theta) \cos \alpha \sin \phi d\phi d\theta \quad (6.15)$$

where  $H$  is the entire hemisphere,  $F_{ii}(\phi, \theta)$  is the flux of particles coming off a surface point in the  $(\phi, \theta)$  direction from the point of interest, and  $R_{ii}$  relates the fluxes to the actual etch rate. The surface reflection algorithm of chapter 5 may be applied to solve equation 6.15, but the time to calculate  $F_{ii}(\phi, \theta)$  will be quite large, unless simplifying assumptions are made.

Tazawa *et. al.* lumped neutral particles and ions into the same component, and assumed that particles arrive from all directions in which sidewalls exist. With these assumptions, the 3D velocity is:

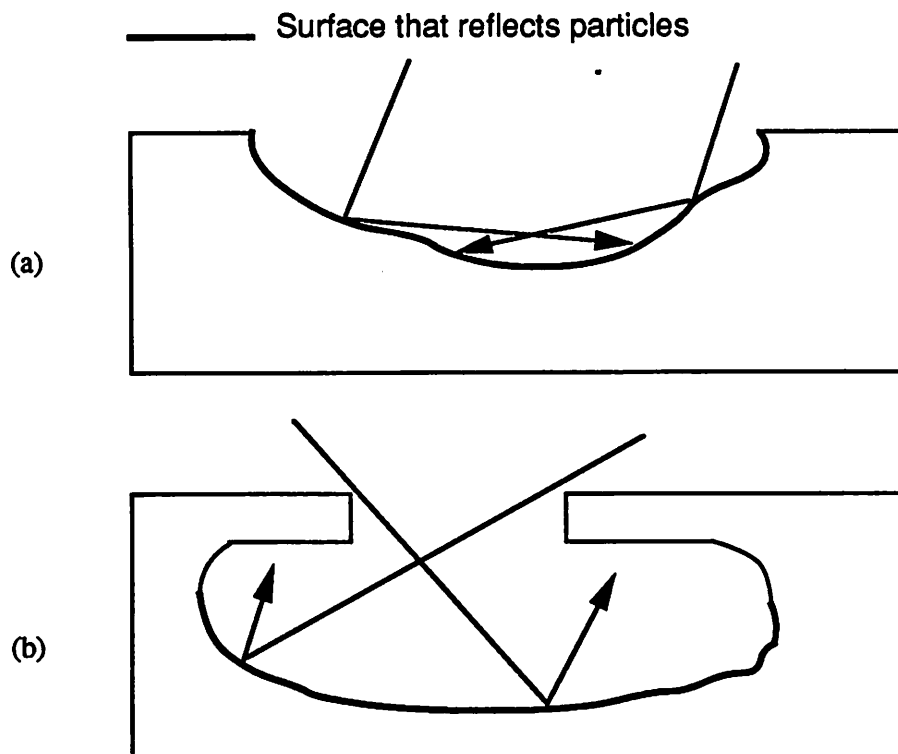
$$\mathbf{v}_{ii}(\mathbf{x}) = R_{ii} \iint_{H - \Omega(\mathbf{x})} \cos \alpha \sin \phi d\phi d\theta \quad (6.16)$$

where  $H - \Omega(\mathbf{x})$  represents the directions in which the plasma cannot be seen, and  $R_{ii}$  relates the flux to the actual etch rate.

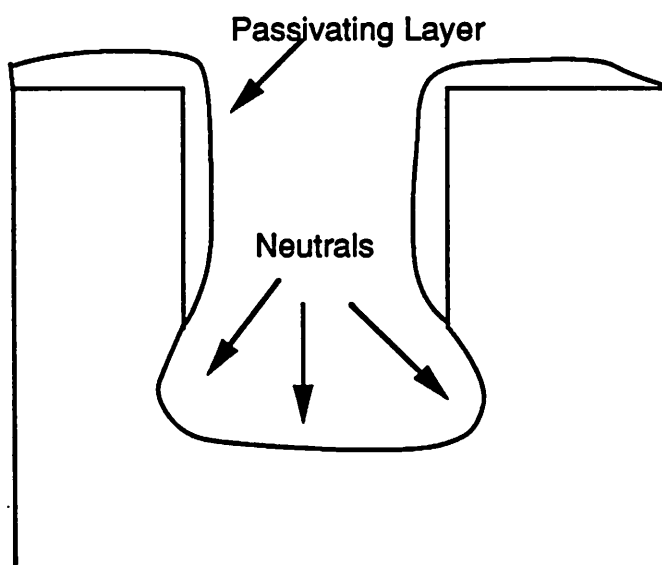
Equation 6.16 is strictly valid only in the case where there is no overhang, and all sidewalls receive directly incident particles as shown in Fig. 6.5a. However for a case like 6.5b a more complete formulation is necessary.

### 6.2.8. Unifying the Model Terms and Determining Constants

All the individual velocity terms may be summed together to give the etch velocity along the surface normal. By introducing a step function for the visibility  $V(\mathbf{x}, \phi, \theta)$  that is 1 if



**Fig. 6.5:** Calculating surface reflection. (a) Simple calculation is possible when there is no overhang. (b) Overhang requires more rigorous reflection calculation.



**Fig. 6.6:** Sidewall passivation due to presence of a thin layer that prevents etching.

the plasma is visible from point  $\mathbf{x}$  along  $(\phi, \theta)$  and zero elsewhere, the integrals may be combined yielding the full unified model for etch velocity along the surface normal:

$$\mathbf{v}(\mathbf{x}) = R_{is} + \int_0^{2\pi} \int_0^{\pi/2} [V(\phi, \theta) \frac{R_{di}}{N_{di}} F_i(\phi, \theta) S_i(\alpha) + V(\phi, \theta) \frac{R_{dn}}{N_{dn}} F_n(\phi, \theta) + (1 - V(\phi, \theta)) R_{ii} F_{ii}(\phi, \theta)] \cos \alpha \sin \phi d\phi d\theta \quad (6.17)$$

The rate parameters  $R_{is}$ ,  $R_{di}$ ,  $R_{dn}$ , and  $R_{ii}$  depend on the substrate materials, the etch gasses, the ion energies, and the pressure and temperature in the etch chamber. The flux distributions  $F_i$ ,  $F_n$ , and  $F_{ii}$ , are also related to the etch gasses and process conditions. Once the flux distributions are given, the normalizing factors  $N_{di}$ , and  $N_{dn}$  may be determined. Finally, the visibility function  $V$  and the integrations account for the changing surface topography.

The rate constants and flux functions may be determined by physical derivations, or by empirical parameter fitting. Full physical models for all the parameters have the advantage that any material in any system with any combination of plasmas and process conditions could be simulated. Unfortunately, such models do not exist. Even when derivations from first principles are possible, they are often limited. For example, Lindhard's formula for sputter yield only gives the peak angle, and not the entire sputter yield function. Even then, the derivation is based on many assumptions, including the applicability of classical mechanics, each of which introduce some error. Instead, an approach combining empirical parameter fits with hypothesized physical relationships is best for practical simulation. Already, several useful yield and flux functions have been mentioned. Data also exists giving sputter yield and etch rates for various materials under various conditions. Additionally, physical models for ion transport have been subjected to Monte Carlo simulation to produce flux distribution data.

Equation 6.17 can be converted into a practical simulation model by introducing some well known empirical or semi-empirical functions to describe the flux distributions and sputter

yield curves. Individual processes and pieces of equipment may be modeled by fitting constant parameters to the resulting unified model. The flux distribution of directly incident ions is given as a Gaussian† with a standard deviation as a fitting parameter:

$$F_i(\phi, \theta) = \exp \left[ -\frac{\phi^2}{\sigma^2} \right] \quad (6.18)$$

The sputter yield function for directly incident ions is similar to that used in SAMPLE, and first proposed by Catana *et. al.*,<sup>25, 26</sup> and requires three curve fitting parameters:

$$S_i(\alpha) = a_1 \cos \alpha + a_2 \cos^2 \alpha + a_3 \cos^4 \alpha \quad (6.19)$$

In the above,  $a_1 + a_2 + a_3 = 1$  to yield  $S_i = 1$  at  $\alpha = 0$ . The directly incident neutral flux yield distribution function is a hyper-cosine with  $m$  as a fitting parameter:

$$F_n(\phi, \theta) = \cos^m \phi \quad (6.20)$$

Finally, the indirectly incident flux distribution is simplified by the assumption that reflection tends to randomize the path that particles take resulting in a uniform arrival from all directions:

$$F_{ii} = 1 \quad (6.21)$$

Possible model parameters for different types of etching are given in table 6.2 below. The different model parameters are weighted according to their relative contribution to the final profile. A blank space indicates an entry of zero.

Table 6.2: Unified Model Parameters for Various Processes									
Process	$R_{is}$	$R_{di}$	$\sigma$	$a_1$	$a_2$	$a_3$	$R_{dn}$	$m$	$R_{ii}$
Plasma Etch	1								
Plasma Etch	0.1	2	10°	1.0			1	1	
RSE	0.01	0.2	10°	3.269	13.11	-15.38	0.7	1	
Ion Milling		1	1°	3.269	13.11	-15.38			

A similar approach to modeling the above processes was used by Tazawa *et. al.*<sup>12</sup> who also

† J. McVittie has suggested that a Gaussian may not fall off rapidly enough to describe the true ion flux distribution, and that a cosine raised to a high power (such as 10) may provide a better fit.

showed how the fitting parameters could be derived from analysis of test structures. They also demonstrated that the above modeling approach could be applied to ECR planarization and sputter deposition.

### 6.3. EXTENSIONS TO THE BASIC MODEL

The unified model presented thus far considers many effects in plasma etching, but leaves out some important additional mechanisms. Since the exact mechanisms for many these additional effects are not well known, it is difficult to develop good physical models. However, the unified rate model may be altered to provide an empirical description of some of these effects.

#### 6.3.1. Sidewall Passivation

The deposition of material on chamber walls and trench slopes tends to reduce the etch rate as depicted in Fig. 6.6. An approach to modeling sidewall passivation has been presented by Pelka *et. al.*<sup>27, 28</sup> The model depends on the ratio between local ion and neutral flux  $\Phi_I/\Phi_N$ . The effective redeposition rate  $R_{Nredap}$  is related to the deposition rate due to the neutral particles  $R_{Ndep}$  as:

$$R_{Nredap} = R_{Ndep} \exp(-\alpha \frac{\Phi_I}{\Phi_N}) \quad (6.22)$$

where  $\alpha$  is a fitting parameter. The etch rate due to the neutral ions is then reduced by the deposited layer. So the effective etch rate due to the neutrals becomes:

$$R_{dn} = R_{dn0} \exp(-\frac{l}{\lambda}) \quad (6.23)$$

where  $l$  is the thickness of passivating layer and  $\lambda$  is a fitting parameter relating  $l$  to the actual etch rate. Pelka *et. al.* found that  $\alpha$  is on the order of 1, and  $\lambda$  is about  $10^{-3} \mu\text{m}$ . Passivation



may be included in the unified model by altering the  $R_{nd}$  parameter in equation 6.14. The redeposited layer depth  $l$  must be stored in the data structure. The redeposition rate  $R_{Nred}$  is calculated from the neutral and incident ion flux distributions as:

$$R_{Nred} = R_{Ndep} \exp[-\alpha \iint_{\Omega(x)} \frac{F_{di}(\phi, \theta)}{F_{dn}(\phi, \theta)} \sin\phi d\phi d\theta] \quad (6.24)$$

which is calculated at each time step and used to increment the  $l$  variable at  $\mathbf{x}$ . If  $R_{Ndep} = 0$ , there is no surface passivation effect.

### 6.3.2. Damage and Ion-Enhanced Etching

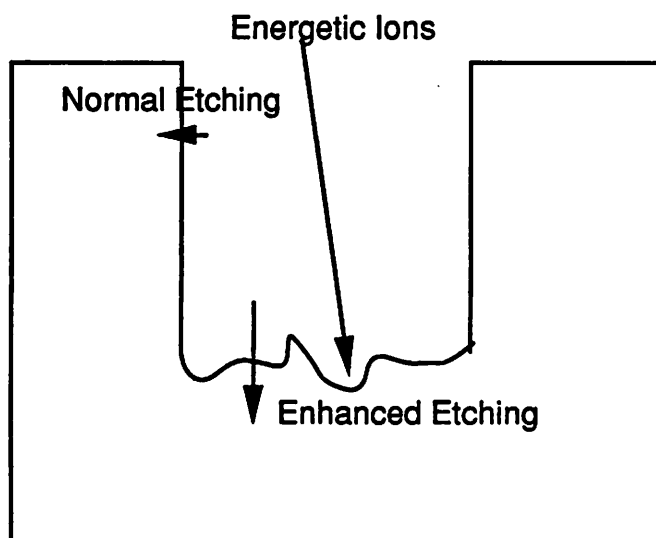
Energetic ions may damage the wafer surface. Reactive neutrals etch the damaged areas more vigorously than undamaged areas, as depicted in Fig. 6.7. In fact, it is possible that energetic ions contribute to anisotropic etching via this mechanism, instead of physical sputtering. Ion-enhanced etching may be modeled by adjusting the neutrals etch rate  $R_{dn}$  in equation 6.14 according to the number of directly incident ions:

$$R_{dn} = R_{dn0} [1 + \frac{D_i}{N_i} \iint_{\Omega(x)} F_{di}(\phi, \theta) \sin\phi d\phi d\theta] \quad (6.25)$$

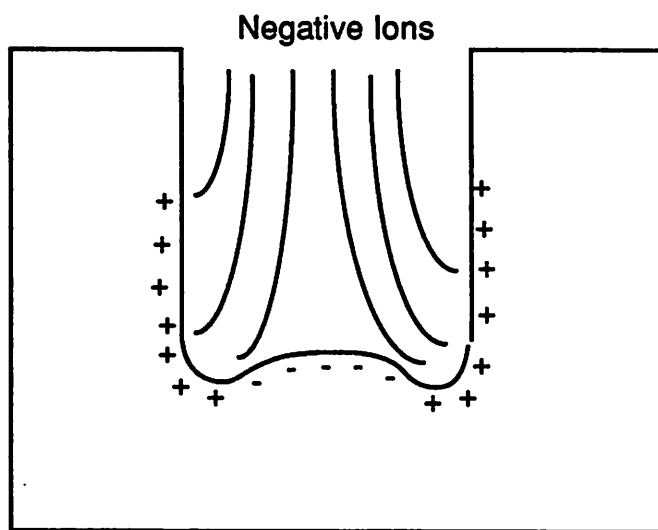
The above expression is one possible model where the etch rate on a flat wafer is  $R_{dn0}(1 + D_i)$ . Simulation results for another similar model have been presented,<sup>28</sup> although the exact model was not fully reported.

### 6.3.3. Microelectric Field Effects

The build-up of charges on the wafer surface may result in local electric fields which steer the incoming ions.<sup>29, 30</sup> This can result in enhanced etching of inside corners and grooves, as shown in Fig. 6.8. Charging and its effect on ion trajectories has been simulated<sup>31</sup> but models have not been fully developed to describe the effect on local etch rate. One



**Fig. 6.7:** Damage induced etching.



**Fig. 6.8:** Microelectric fields due to surface charging.

difficulty with the microelectric field effect is that it violates the assumption that ions travel linearly in the scale of the feature sizes. Nevertheless, a simple phenomenological model may be developed that relates the local surface charge to an increase or reduction of incoming ions. The charge buildup must be stored over the course of the simulation. Sawin *et. al.* have proposed that the etch rate becomes proportional to the charge density.<sup>30</sup> The unified model could be altered to account for this effect by storing a charging term in the data structure that grows with incoming ions. The local etch rate is then varied proportionally with the local charge density. Another more rigorous approach would be to solve the electric fields exactly with a Laplace equation solver to determine the actual ion flux at each surface point.

#### 6.3.4. A Surface Diffusion Model

Gérodolle and Pelletier have suggested that a purely reactive model for plasma etching of Si by  $\text{SF}_6$  may be sufficient to model observed phenomena such as barreling, trench size variation, and enhanced etching of corners and grooves.<sup>10</sup> The model is effective since a variety of observed effects may be reproduced given only simple assumptions about the particle fluxes. It also assumes no reflection of particles, which simplifies the required geometric calculations. However, the surface diffusion calculation does introduce additional numerical complexity. The Gérodolle-Pelletier hypotheses can be included in the 3D simulator, demonstrating the flexibility of the surface evolution algorithms in supporting different physical physical. 3D simulation may then be used to further test the model. This section describes the surface diffusion model and develops a three-dimensional extension suitable for implementation in SAMPLE-3D.

The surface diffusion model for plasma etching of Si is based on the balance of the flux  $v(\psi)$  of atomic fluorine adsorbed from the vapor phase and the flux  $\rho(\psi)$  of fluorine desorbed

from the surface as  $\text{SiF}_4$ , where  $\psi$  represents the flourine coverage, or percentage of the surface monolayer that consists of flourine. The fluxes  $v$  and  $\rho$  are related by a surface diffusion equation which is given in steady state as:

$$\sigma_0 \frac{\partial}{\partial x} (D \frac{\partial \psi}{\partial x}) + v(\psi) - \rho(\psi) = 0 \quad (6.26)$$

where  $\sigma_0$  is the density of adsorption sites in the surface monolayer,  $D$  is the diffusion constant, and  $x$  is a distance along the curved surface. The etch velocity along the surface normal is proportional to the flux of desorbed flourine

$$v = C \cdot \rho(\psi) \quad (6.27)$$

where  $C$  depends on the density of etched material.

The flux  $v(\psi)$  is equal to the random flux of atoms hitting the surface and their adsorption probability. The probability is the product of the sticking coefficient and the fraction of unoccupied adsorption sites.

$$v(\psi) = \kappa p \frac{s - \psi}{s} \quad (6.28)$$

where  $\kappa$  is the adsorption constant,  $p$  is the partial pressure of flourine, and  $s$  is the saturation coverage.

The flux of flourine desorbed as  $\text{SiF}_4$  is derived assuming associative desorption. For a flourine coverage less than a threshold value  $\psi_t$ , the spontaneous reaction rate is zero, and etching occurs only due to the physical sputtering of incident ions. Ion bombardment generates  $\text{SiF}_2$  pairs and introduces the associative desorption of  $\text{SiF}_4$ . The resulting desorption rate proportional to the current density of the ions  $j$  and the flourine density  $\sigma_0 \psi$ . If the flourine coverage is greater than the threshold coverage  $\psi_t$ , a spontaneous etch rate proportional to the coverage of  $\text{SiF}_2$  pairs may also exist.

The final expression for the adsorption, desorption and surface diffusion balance is given by Gerodolle and Pelletier as:

$$\sigma_0 \frac{\partial}{\partial x} \left( D \frac{\partial \psi}{\partial x} \right) + \kappa p \frac{s - \psi}{s} - \eta j \sigma_0 \psi - \frac{\sigma_0}{\tau} \max(\psi - \psi_i, 0) = 0 \quad (6.29)$$

where  $\tau$  is the adsorption lifetime of flourine before spontaneous desorption and  $\eta$  is the ion reaction rate constant. Multiplying each term by  $\tau/\sigma_0$  and defining  $A = \kappa p \tau/\sigma_0$ ,  $J = \eta j \tau$  and  $L^2 = D \tau$  results in an equation with three constant fitting parameter terms ( $A$  and  $J$  are dimensionless and  $L$  is a diffusion length):

$$\frac{\partial}{\partial x} \left( L^2 \frac{\partial \psi}{\partial x} \right) + A \frac{s - \psi}{s} - J \psi - \max(\psi - \psi_i, 0) = 0 \quad (6.30)$$

Equation 6.30 is solved numerically for  $\psi$  at each point. The etch rate is then given by:

$$v = C \frac{\sigma_0}{\tau} [\max(\psi - \psi_i, 0) + J \psi] \quad (6.31)$$

### 6.3.5. A 3D Extension of the Surface Diffusion Model

To extend the above analysis to 3D, the ion current and neutral flux terms must be related to the 3D visibility and particle flux distributions, and equations 6.26 and 6.30 must be generalized to account for 2D surface diffusion. The flux distributions may be incorporated directly. The ion current term  $j$  in equation 6.29 is the flux of directly incident ions directed along the surface normal arriving at a point per unit of time. At a point  $\mathbf{x}$ :

$$j(\mathbf{x}) = \frac{J_i}{N_{di}} \iint_{\Omega(\mathbf{x})} F_i(\phi, \theta) \cos \alpha \sin \phi d\phi d\theta \quad (6.32)$$

where  $J_i$  is the ion current density from the source,  $\alpha$  is the angle between the surface normal and the incoming ions, and  $N_{di}$  is a normalizing factor equal to the flux integrated over the visible hemisphere seen from a flat wafer. The partial pressure of flourine atoms  $p$  in equation 6.28 may be related to the flux of neutral particles:

$$p(\mathbf{x}) = \frac{P}{N_{dn}} \iint_{\Omega(\mathbf{x})} F_n(\phi, \theta) \cos \alpha \sin \phi d\phi d\theta \quad (6.33)$$

where  $P$  is the partial pressure on a flat wafer,  $N_{dn}$  is a normalizing factor equal to the flux seen at a point on a flat wafer, and  $\alpha$  is the angle between the surface normal and incident particles. With equations 6.32 and 6.33, the fluxes  $v$  and  $\rho$  are related to the changing surface geometry.

At each time step in the surface evolution, the incident ion flux and neutral flux at each surface point will change. If the surface reactions control the etch rate, it is possible to assume that the surface diffusion occurs much faster than the surface evolution. This allows the surface diffusion equation to be solved as a time independent process. To reduce the number of variables in the following analysis let  $\Gamma^2(\psi) = (u+A/s+J)/L^2$  and  $\Lambda(\psi) = [\psi, u-A]/L^2$  where  $u$  is a unit step function equal to 1 if  $\psi > \psi_i$  and equal to 0 if  $\psi \leq \psi_i$ . The variables  $\Gamma$  and  $\Lambda$  contain all the physical and geometric information in the previous analysis. The surface diffusion equation is now:

$$\nabla^2 \psi - \Gamma^2(\psi) \psi = \Lambda(\psi) \quad (6.34)$$

The laplacian operator  $\nabla^2$  represents second-order differentiation in a coordinate system defined on the material surface. The coefficients are functions defined on the on the surface that are essentially linear in  $\psi$  except for the unit step function dependence on  $\psi - \psi_i$ . Equation 6.34 is thus a piece-wise linear, time-independent diffusion equation.

### 6.3.6. An Approach to Solving the 2D Surface Diffusion Equation

The solution of the surface diffusion equation 6.34 gives the fluorine coverage  $\psi_i$  at each point on the surface. The etch velocity along the surface normal is a simple function of  $\psi_i$  exactly as given by equation 6.34, which includes no new terms. The surface diffusion equa-

tion may be solved numerically using information contained in the surface mesh. At each time step  $\Gamma_i$  and  $\Lambda_i$  must be calculated for each point based on the materials, surface geometry, and process environment. Once the surface constants are defined, equation 6.34 may be solved by well-known finite-element methods.<sup>32, 33, 34</sup>

The spatial discretization of the surface is somewhat complicated by its 3D nature. However, the surface is already defined as a mesh of triangles. Since the triangle edges have been split and merged during the surface evolution, the mesh is distributed fairly evenly over the surface. The surface mesh elements may be transformed to finite-element mesh elements according to principles used in the analysis of thin shells in structural mechanics.<sup>35, 36</sup> The problem is simpler than the general shell problem, since the surface diffusion equation does not require deformation and arbitrary 3D loading terms. Once the elements are defined, the matrix assignment may proceed.

The boundary condition for the problem is defined at material interfaces. In most simple etch simulations, the material interface is at a mask edge that is not etching. Then  $\Gamma$ ,  $\Lambda$ , and  $\psi$  are all constant at the interface. If the change of  $\psi$  is constant at every interface node then Neumann boundary conditions apply.

The time to set up the finite-element matrix is linearly proportional to the number of mesh nodes. For a full matrix, the time to calculate its inverse, necessary to solve the finite-element problem, is  $O(N^3)$  for  $N$  nodes,<sup>37</sup> which is actually worse than the time required for full reflection calculation. By properly ordering the nodes, a banded, sparse matrix will result, with only a few non-zero elements clustered near the diagonal. The matrix inversion then requires only  $O(NB^2)$  for  $N$  nodes and  $B$  elements giving the width of the band. In practice  $B \ll N$ , yielding very efficient computation times.<sup>33</sup>

The non-linearity due to the unit step function may be handled by solving successive linear problems until the threshold is reached, at which point the step function reverts to 0 or 1. This amounts to replacing the  $\Gamma$  and  $\Lambda$  values for any point where  $\psi$  exceeds  $\psi_i$ , and then continuing iterations. The finite-element solution converges when the maximum difference for  $\psi$  between two successive iteration steps is less than some defined value for any node.

Except at the point where  $\psi = \psi_i$ ,  $\Gamma$  and  $\Lambda$  are largely independent of  $\psi$  over a small time step. Thus it is possible to find an approximate solution to equation 6.34 by convolving a diffusion length  $\sigma$  dependent Gaussian function with the surface points:

$$\psi_i = \iint_S \psi(s) e^{-\frac{1}{2}(\frac{s-r}{\sigma})^2} dA \quad (6.35)$$

where  $S$  is the surface,  $s$  is a surface coordinate, and  $r$  is a curvilinear distance on the surface. The initial value of  $\psi$  before the convolution depends on  $\Gamma$  and  $\Lambda$ , which in turn depend on the ion and neutral flux distributions. At the start of the simulation,  $\psi$  is zero everywhere.

In order to apply the Gaussian convolution, the curvilinear distance between two points must be known. Unfortunately, it is not always easy to find the shortest distance between two points on a curved surface. The problem is how to find the distance between two arbitrary points given an arbitrary mesh description of the surface. Some approaches to this problem are described in chapter 7, where a similar convolution is used for surface migration calculations. Another problem is the time to calculate the convolution integral. In practice, the problem may be reduced by considering only the surface within a  $3\sigma$  distance of the point of interest. The convolution may be evaluated by applying a fast Fourier transform. This results in  $O(M \log M)$  time where  $M$  is the number of points within  $3\sigma$  distance of the point of interest, yielding  $O(NM \log M)$  time for the entire surface.<sup>37</sup> Interestingly, the time to calculate the convolution is only slightly better than that needed for the finite element solution of the diffusion



equation. In fact, for a large diffusion length, the finite element solution is likely to be faster. A finite-difference method may also be applicable, but the irregular distribution of the surface points does not lend itself easily to an accurate finite-difference solution method

## 6.4. PLASMA ETCHING AND ION MILLING SIMULATIONS

### 6.4.1. Coupling The Models and Algorithms

The unified model of the section 6.4 and the surface diffusion model of section 6.5 have been coupled with the facet-motion algorithm and the surface-cell hybrid data representation to perform 3D simulation of etching processes. This section presents some discussion of the computer implementation and the effort to improve computational efficiency. The section concludes with several simulation results.

It is desirable to reduce the number of calculations by avoiding redundant operations. On initialization, the hemisphere above the wafer surface is divided into patches of  $\Delta\phi$  and  $\Delta\theta$ . The differential area, and average values of  $F_i$  and  $F_n$  may be determined for each patch. Then the solid angle for each node must be determined. The integration for each facet attached to a triangle must be performed separately, since  $\cos\alpha$  may vary. However, the solid angle calculation need only be performed once at which time a flag on each patch is set to 1 or 0. This is equivalent to defining the function  $V$  in equation 6.17. A total of  $3T$  integrations must be performed, where  $T$  is the number of surface triangles.

If the integrand is a separable function of differential flux orientation and surface orientation, then the constant surface orientation terms may be taken outside of the integral. This is true for all the terms in equation 6.17 which do not have an angle of incidence for the rate

along the incident direction, *i.e.* the neutral flux term and the reflected particle term. In that case, the etch vector at a point may be determined solely from the incoming flux distribution. The etch rate at each facet may then be determined by projecting the etch vector onto the facet normal. The rates along the three Cartesian coordinate directions are:

$$v_x = R \iint_{\Omega} f(\phi, \theta) \sin \theta \sin \phi \sin \phi d\phi d\theta \quad (6.36a)$$

$$v_y = R \iint_{\Omega} f(\phi, \theta) \cos \theta \sin \phi \sin \phi d\phi d\theta \quad (6.36b)$$

$$v_z = R \iint_{\Omega} f(\phi, \theta) \cos \phi \sin \phi d\phi d\theta \quad (6.36c)$$

The rate along the surface normal is then given as:

$$v(\mathbf{x}) = \sin \phi_x \cos \theta_x v_x + \sin \phi_x \sin \theta_x v_y + \cos \phi_x v_z \quad (6.37)$$

It is useful to preserve the direction given by equation 6.36 since that is the maximum rate direction at the point. Since the maximum rate direction is derived from the integral over the visibility solid angle, it will often be contained within the facet-motion solid angle, thus describing the exact location of the new point. The computation time is also reduced by calculating  $\sin \theta \sin \phi$ ,  $\cos \theta \sin \phi$ , and  $\cos \phi$  in advance for each hemisphere patch.

The integration must be performed separately for each facet if the rate is dependent on angle of incidence, as in sputtering by ion bombardment. However, it is still necessary to determine the fastest rate direction to apply the facet-motion algorithm. If the standard deviation of the ion flux distribution is small, the directly incident ion component may be assumed to proceed largely in the direction from the source. The maximum rate direction falls on the angle giving the maximum of  $S_i(\alpha) \cos \alpha$ . If  $S_i$  is given by equation 6.19, the maximum may be found by solving for  $0 \leq \alpha \leq \pi/2$ :

$$\frac{d}{d\alpha} [a_1 \cos^2 \alpha + a_2 \cos^3 \alpha + a_3 \cos^5 \alpha] = 0 \quad (6.38)$$

Differentiating gives:

$$(-2a_1 - 3a_2\cos\alpha - 5a_3\cos^3\alpha)\cos\alpha\sin\alpha = 0 \quad (6.39)$$

The cubic polynomial factor has the following real root:

$$\cos\alpha = \frac{-a_2}{5a_3} \left[ \frac{-a_1}{5a_3} + \sqrt{\frac{a_2^3}{125a_3^3} + \frac{a_1^2}{25a_3^2}} \right]^{-1/3} + \left[ \frac{-a_1}{5a_3} + \sqrt{\frac{a_2^3}{125a_3^3} + \frac{a_1^2}{25a_3^2}} \right]^{1/3} \quad (6.40)$$

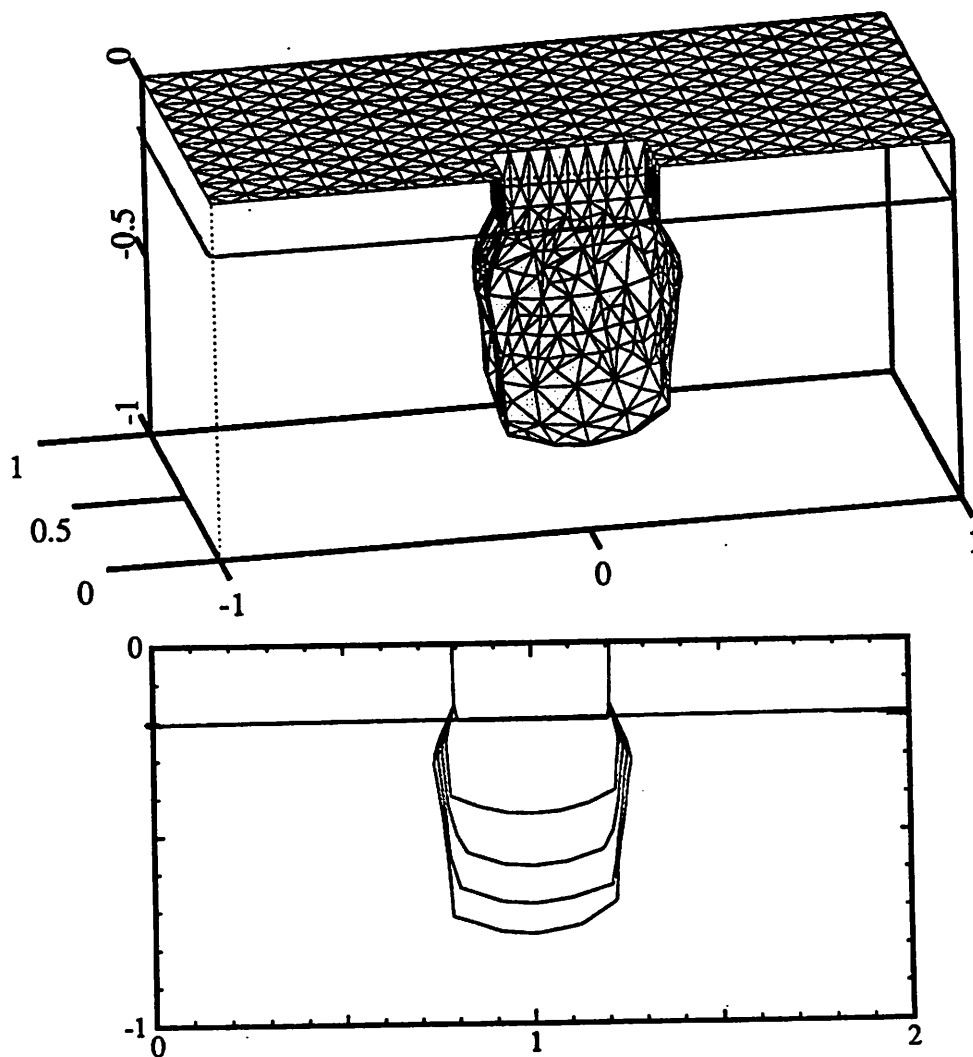
The two parts of the right hand side of equation 6.46 may be complex, but the imaginary parts cancel each other out if the fitting parameters describe a valid sputter yield curve.

If the various components have equivalent contributions to the overall etch rate, the maximum rate direction can be found by summing the maximum rate vectors due to each component. If the flux functions are not known analytically, the maximum direction may still be found numerically by testing several possible directions in the facet-motion solid angle, although this step would be more time consuming.

#### 6.4.2. Simulation Examples

The following examples are not meant to fully describe real processes, but rather to exercise several of the possible etch components described thus far. The specific conditions are included with each figure. The 3D plots are cutaway views of full 3D simulations.

**Ion Enhanced Plasma Etching:** Fig. 6.9 shows the barreling phenomenon which results in ion enhanced plasma etching. In this example, energetic ions hit the bottom of the trench resulting in a higher etch rate there. Reactive neutrals can still attack the sidewalls, however, resulting in a negative profile slope just under the mask. The simulation used  $45 \times 10$  hemisphere patches and  $100 \times 100 \times 100$  cells and required 923 seconds of CPU time (IBM RS600/530) to reach the final profile.



**Fig. 6.9: 3D Dry Etch Simulation Model Parameters**

Rates	$R_{is}=0.0$	$R_{di}=0.0$	$R_{dn}=1.0$	$D_n=6.0$	$R_{ii}=0.0$
Fluxes	neutral $m=1.0$	ion $\sigma=0.0$	ion current= $0.5\text{mA/cm}^2$		$F_{ii}=0.0$
Variations	$R_{Ndep}$	$\alpha$	$\lambda$	charging= $0.0$	
Sputter Yield	$S_0=0.0$	density= $1.0$	$a_1$	$a_2$	$a_3$

**RIE Lag:** The well known phenomenon of RIE lag is shown in Fig. 6.10. For two different size contact holes, a different etch rate results. The larger hole allows a greater number of particles to contribute to the etch rate, thus resulting in a deeper trench. The simulation used 45x10 hemisphere patches and 100x100x100 cells and required 1501 seconds of CPU time (IBM RS6000/530) to reach the final profile.

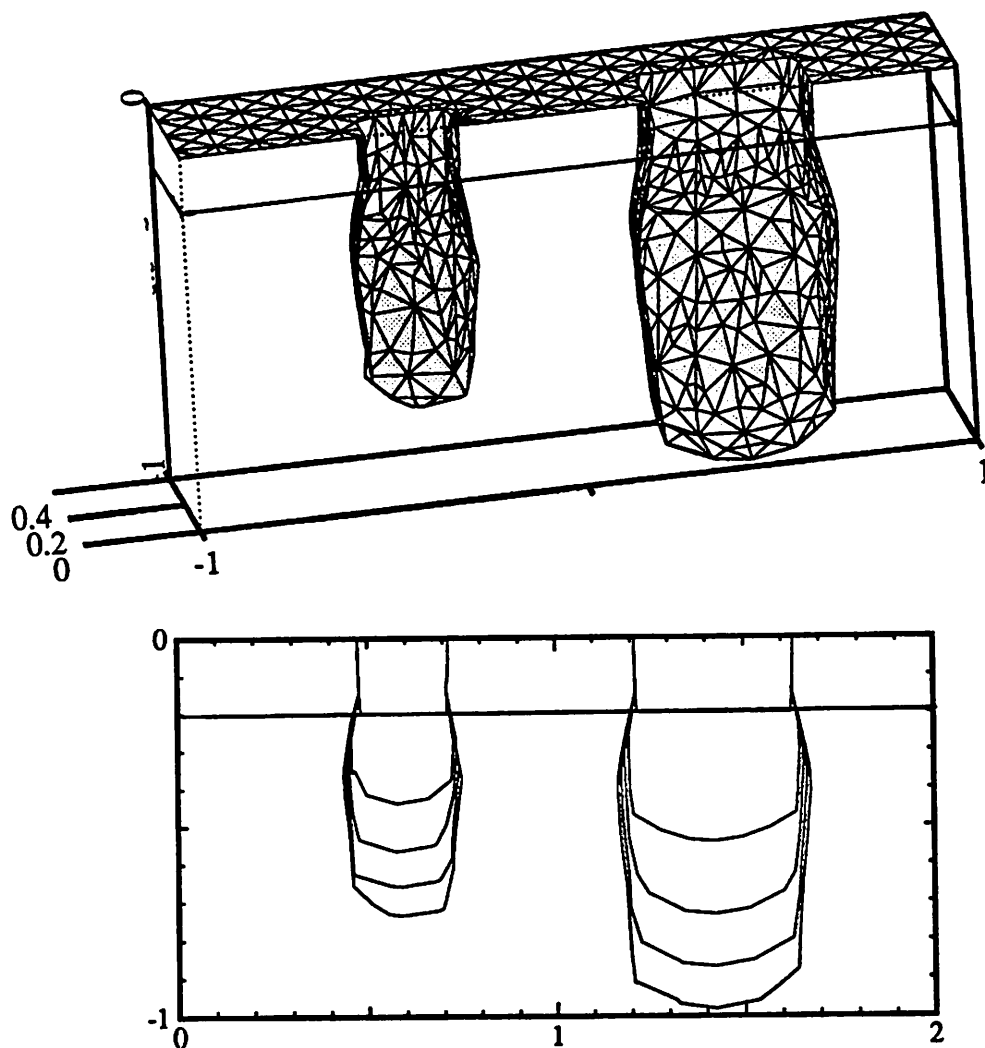
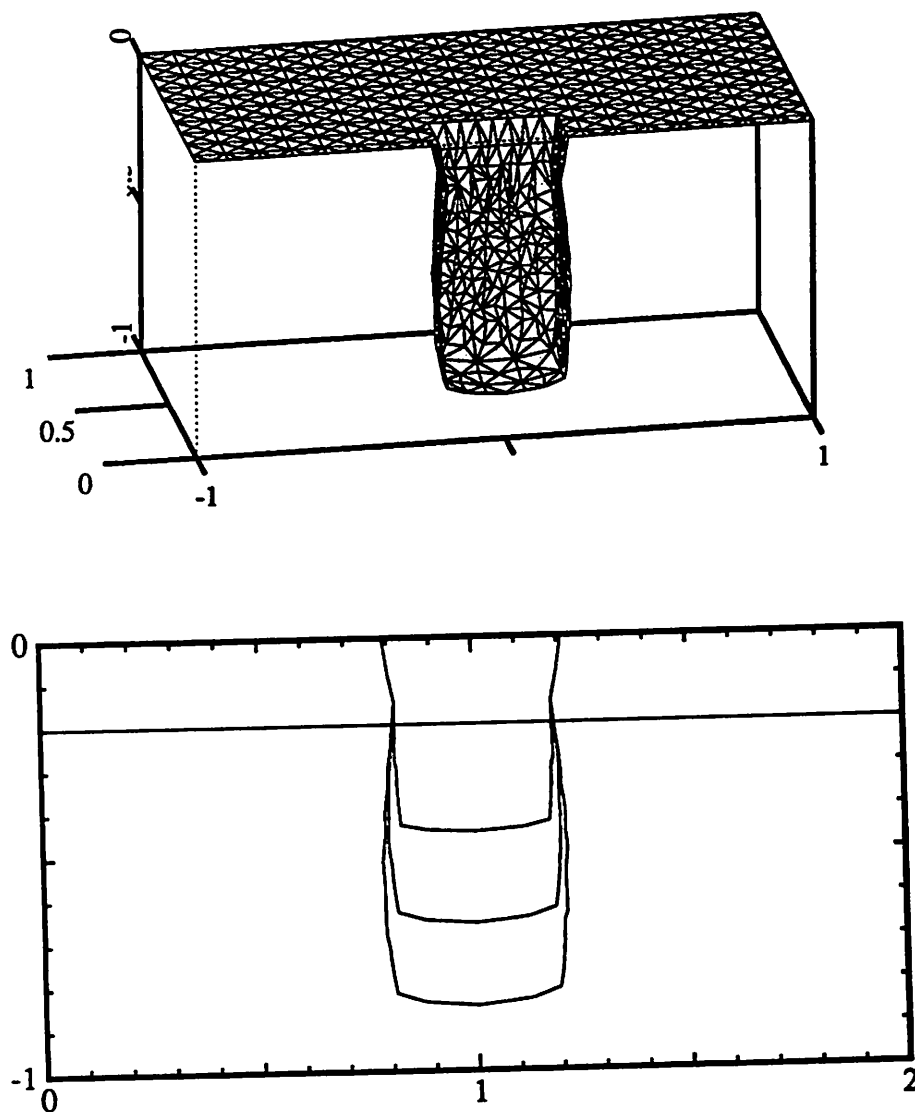


Fig. 6.10: 3D Dry Etch Simulation Model Parameters					
Rates	$R_{is}=0.0$	$R_{di}=0.0$	$R_{dn}=1.0$	$D_n=5.0$	$R_{ii}=0.05$
Fluxes	neutral $m=4.0$	ion $\sigma=0.0$	ion current= $0.5\text{mA}/\text{cm}^2$		$F_{ii}=1.0$
Sputter Yield	$S_0=0.0$	density=1.0	$a_1$	$a_2$	$a_3$

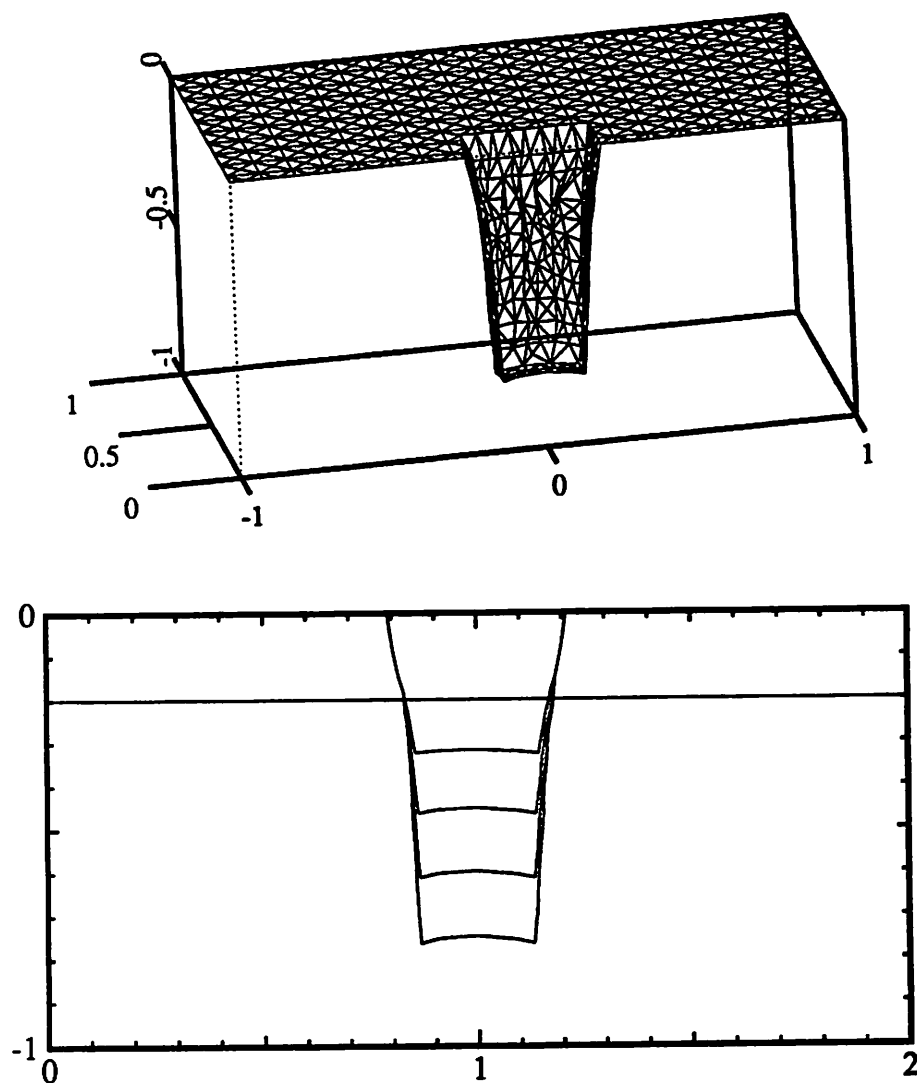
**Reflection:** Fig. 6.11 shows the anisotropy which results when the simple reflection model is added to the simulation. The etch rate at the bottom of the trench is enhanced.



**Fig. 6.11: 3D Dry Etch Simulation Model Parameters**

Rates	$R_{ii}=0.0$	$R_{di}=0.0$	$R_{dn}=1.0$	$D_n=3.0$	$R_{ii}=0.5$
Fluxes	neutral $m=1.0$	ion $\sigma=0.0$	ion current= $0.5\text{mA}/\text{cm}^2$		$F_{ii}=1.0$
Sputter Yield	$S_0=1.0$	density=1.0	$a_1$	$a_2$	$a_3$

**Dovetail in Reflection:** the result of Fig. 6.12. When the ion enhancement term is removed, the inside corners of the trench etch faster due to reflection. This results in a convex taper to the bottom of the trench.



**Fig. 6.12: 3D Dry Etch Simulation Model Parameters**

Rates	$R_{is}=0.0$	$R_{di}=0.0$	$R_{dn}=1.0$	$D_n=0.0$	$R_{ii}=0.5$
Fluxes	neutral $m=1.0$	ion $\sigma=0.0$	ion current= $0.5\text{mA}/\text{cm}^2$		$F_{ii}=1.0$
Sputter Yield	$S_0=0.0$	density=1.0	$a_1$	$a_2$	$a_3$

**Ion Milling:** In Fig. 6.13 only energetic ions etch the surface. In the absence of no reflection or redeposition, very straight sidewalls, and sharp inside corners occur. The effect of the boundary conditions in the algorithm is also visible at the inside corner. Different models, may require changing that boundary condition.

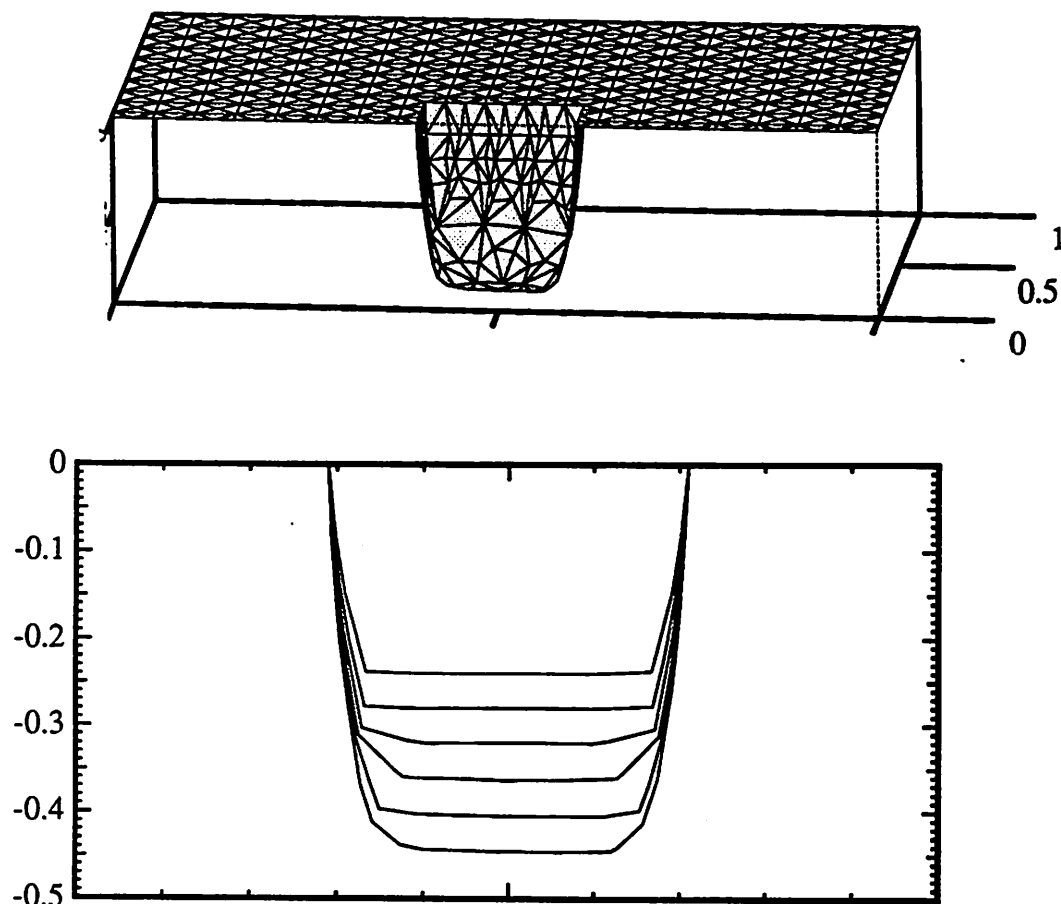


Fig. 6.13: 3D Dry Etch Simulation Model Parameters					
Rates	$R_{is}=0.0$	$R_{di}=1.0$	$R_{dn}=0.0$	$D_n=0.0$	$R_{ii}=0.0$
Fluxes	neutral $m=1.0$	ion $\sigma=0.0$	ion current= $0.5\text{mA}/\text{cm}^2$		$F_{ii}=0.0$
Sputter Yield	$S_0=1.0$	density=1.0	$a_1=3.27$	$a_2=13.11$	$a_3=-15.38$



## 6.5. CRYSTAL ETCHING

Monocrystalline materials show different etch rates along different directions.<sup>38</sup> The etch rate dependence on surface orientation is mathematically equivalent to incident angle dependence in ion milling, but with a different rate versus orientation function. Foote and Séquin have developed a model for the computation of geometric offset surfaces in crystal etching.<sup>39, 40</sup> The model begins by determining etch rates for several key crystallographic planes based on the geometry of the crystal lattice, atomic spacings, and the angles between bonds. The geometric model is then related to known experimental etch rate data. Finally an interpolation rule is applied to calculate the etch rates for all possible directions, by assuming that the etch slowness surface between extrema directions is nearly planar.

Once the rates for all directions are known, it is possible to determine the behavior of facet formation and collapse at corners and edges. Foote applied Frank's theorems of crystal dissolution<sup>14</sup> to determine the formation or disappearance of faces. The facet-motion algorithm is an approximation of the rigorous analysis by Foote, and achieves similar results as long as the time step is sufficiently small and new points are added to the surface mesh as the surface expands.

Fig 6.14 shows wet etching of a masked (001) Si plane. The 3D etch slowness diagram was constructed by defining the etch rates along the {001}, {011}, {111}, {112}, {122}, and {012} directions, yielding 19 vectors, and a surface of 24 triangles in quadrant I. Locating the intersection of the surface with a given direction, and finding the inverse of the length from the origin to the intersection, gives the etch rate for the given direction. In this example the etch rate along the {001} planes was 100 times greater than along the {111} planes, resulting in v-shaped grooves oriented 54.7° from the z-axis.

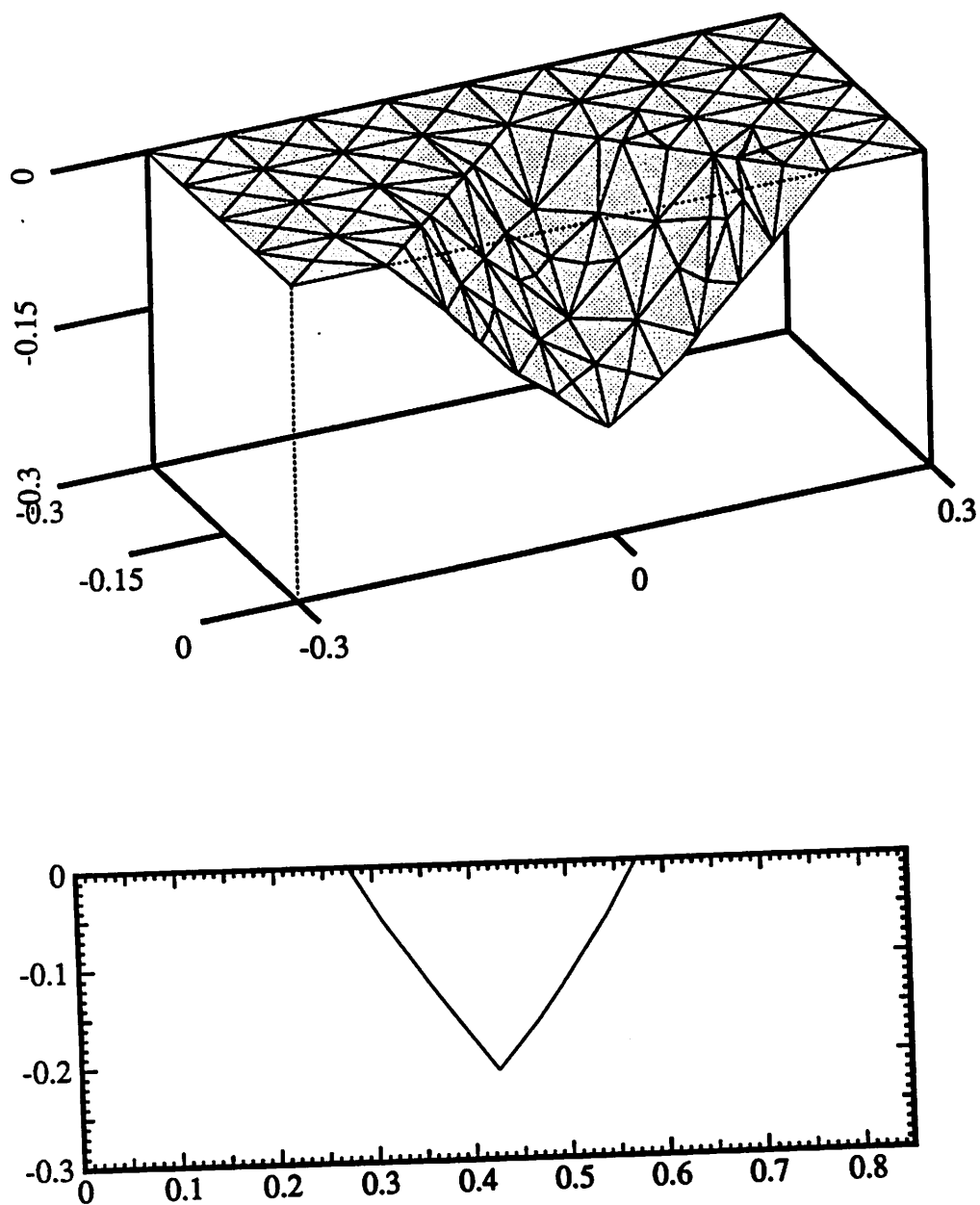


Fig. 6.14: Wet etching of Si with 100 face exposed. Side view is along plane  $y=x$ .

## 6.6. SUMMARY AND CONCLUSION

There is still much discussion among researchers as to the physical mechanisms responsible for etch processes used in integrated circuit fabrication. Work must be done to separate the dominant effects from those of lesser importance. Simulation has an important role to play since it allows the testing of hypothesis under controlled conditions. A comparison of simulation results with experimental evidence may then yield new insight into the fundamental physics. Simulation also has a practical role when a process is well characterized. Even if all the physical effects are not known in detail, a set of well-chosen fitting parameters is often enough to describe a process completely. A sufficiently accurate model yields useful simulation results at much lower cost than a series of experiments.

Given the great number of etch processes and the rate at which new models and physical mechanisms are introduced, process simulation must be amenable to continual improvement, refinement, and growth. The primary goal of this chapter has been to demonstrate how a well-chosen set of surface advancement algorithms and geometric operations may be applied to a large class of models for isotropic etching, plasma etching, ion milling, and crystal etching. Another goal of this chapter was to develop process models which were not restricted to a particular set of algorithms. All of the models presented here depend solely on the surface shape, the surface materials, and the types and distributions of etchants arriving at the surface. The simulation examples result from a set of algorithms and data structures which allow the efficient evaluation of mathematical expressions, but which did not depend on the algorithms to perform the evaluations. In this sense, the models and algorithms are decoupled. This allows improvements to be made to individual models or algorithms without having to rebuild the entire simulator.

If there are limitations to the generalized approach presented here, they exist in three areas. First, the models have not explicitly considered fundamental atomic or molecular reactions. Instead, more basic phenomena are contained within the etch rate constants, sputter yield curves, and surface reflection and diffusion models. It is conceivable that models for more basic physical phenomena could replace the rate constants, so it is possible to build on the philosophy of this chapter to include atomic level reactions as they become known. Second, the algorithms for surface advancement and visibility do introduce some numerical error. Decreasing the time step and increasing the density of information results in more accurate simulations, but also in longer simulation times, and more demanding memory requirements. Practical considerations limit the accuracy which may be achieved. Fortunately, currently available engineering workstations are sufficiently powerful to produce useful results as demonstrated by several simulation examples. Furthermore, there is always room for continued refinement and improvement of the algorithms to further enhance the simulation capability. Third, the general organization of the algorithms may be slower than implementations focused on simulating a small class of specific process models. For example, it may be possible to carry out certain integrals analytically. Fortunately, it is possible to turn on and off certain elements of the various algorithms depending on the model constants. For example, it is not necessary to calculate the full integration over the ion flux distribution if the ion spread is very narrow. Chapter 8 provides additional discussion of how the algorithms may be organized so that generality does not unduly sacrifice efficiency.

Many of the topics raised in etching simulation also apply to the deposition of thin films on topographies, which is the subject of the next chapter.

## REFERENCES

1. A.D.G. Stewart and M.W. Thompson, "Microtopography of Surfaces Eroded by Ion-Bombardment," *Journal of Materials Science*, vol. 4, pp. 56-60, 1969.
2. D.J. Barber, F.C. Frank, M. Moss, J.W. Steeds, I.S.T. Tsong, "Prediction of Ion-bombarded Surface Topographies Using Franks's Kinematic Theory of Crystal Dissolution," *Journal of Materials Science*, vol. 8, pp. 1030-1040, 1973.
3. J.P. Ducommun, M. Cantagrel, and M. Marchal, "Development of a general surface contour by ion erosion. Theory and computer simulation," *Journal of Materials Science*, vol. 9, pp. 725-736, 1974.
4. S. Matsuo, "An Analytical Treatment on the Pattern Formation Process By Sputter Etching," *Japanese Journal of Applied Physics*, vol. 15, no. 7, pp. 1253-1262, July 1976.
5. J.P. Ducommun, M. Cantagrel, and M. Moulin, "Evolution of well-defined surface contour submitted to ion bombardment: computer simulation and experimental investigation," *Journal of Materials Science*, vol. 10, pp. 52-62, 1975.
6. R. Smith, G. Carter, and M.J. Nobes, "The theory of surface erosion by ion bombardment," *Proceedings of the Royal Society of London A*, vol. 407, pp. 405-433, 1986.
7. C.W. Jurgensen and E.S.G. Shaqfeh, "Kinetic Theory of Bombardment Induced Interface Evolution," *Journal of Vacuum Science Technology B*, vol. 7, no. 6, pp. 1488-1492, Nov/Dec 1989.
8. J. McVittie, J. Rey, L.-Y. Cheng, A. Bariya, S. Ravi, and K. Saraswat, "SPEEDIE: A Profile Simulator for Etching and Deposition," *TECHCON '90, Extended Abstract Volume*, pp. 16-19, Semiconductor Research Corporation, San Jose, California, October

16-18, 1990.

9. J.P. McVittie, J.C. Rey, L.Y. Cheng, M.M. IslamRaja, and K.C. Saraswat, "LPCVD Profile Simulation Using a Re-Emission Model," *IEDM Technical Digest*, pp. 917-920, San Francisco, CA, December 1990.
10. A. Gerodolle and J. Pelletier, "Two-Dimensional Implications of a Purely Reactive Model for Plasma Etching," *IEEE Transactions on Electron Devices*, vol. 38, no. 9, pp. 2025-2032, September 1991.
11. W.G. Oldham, A.R. Neureuther, C. Sung, J.L. Reynolds, and S.N. Nandgaonakar, "A General Simulator for VLSI Lithography and Etching Processes: Part II - Application to Deposition and Etching," *IEEE Transactions on Electron Devices*, vol. ED-27, no. 8, pp. 1455-1459, August 1980.
12. S. Tazawa, S. Matsuo, and K. Saito, "Unified Topography Simulator for Complex Reaction Including Both Deposition and Etching," *1989 Symposium on VLSI Technology, Digest of Technical Papers*, pp. 45-46, May 1989.
13. H. Lamb, *Hydrodynamics*, p. 7, Dover, New York, 1945.
14. F.C. Frank, "On the Kinematic Theory of Crystal Growth and Dissolution Processes," in *Growth and Perfection of Crystals*, ed. R.H. Doremus, B.W. Roberts, D. Turnbull, pp. 411-419, Wiley and Sons, New York, 1958.
15. H. Bach, "Messung der Zerstaebungsraten an Nichtleitern und Bestimmung der Bindungsenthalpie von Kieselglas," *Naturwissenschaften*, vol. 9, pp. 429-430, 1968.
16. J. Lindhard, "Influence of Crystal Lattice on Motion of Energetic Charged Particles," *Matematisk-Fysiske Meddelelser*, vol. 34, no. 14, 1965.
17. H.H. Andersen and H.L. Bay, "Sputtering Yield Measurements," in *Sputtering by Particle Bombardment I*, ed. R. Behrisch, vol. 47, pp. 201-202, Springer-Verlag, Berlin,

1981.

18. H. Bach, *Journal of Non-Crystalline Solids*, vol. 3, p. 1, 1970.
19. R. Smith, S.J. Wilde, G. Carter, I.V. Katardjiev, and M.J. Nobes, "The simulation of two-dimensional surface erosion and deposition processes," *Journal of Vacuum Science and Technology B*, vol. 5, no. 2, pp. 579-585, March/April 1987.
20. A.R. Neureuther, C.Y. Liu, and C.H. Ting, "Modeling Ion Milling," *Journal of Vacuum Science Technology*, vol. 16, no. 6, pp. 1767-1771, November/December 1979.
21. N. Yamauchi, T. Yashi, and T. Wada, "A pattern edge profile simulation for oblique ion milling," *Journal of Vacuum Science Technology A*, vol. 2, no. 4, pp. 1552-1557, October-December 1984.
22. J. Pelka, H.-C. Scheer, P. Hoffmann, W. Hoppe, and C. Huth, *Microelectronic Engineering*, vol. 9, p. 491, North-Holland, 1989.
23. J.I. Ulacia-Fresnedo and J.P. McVittie, "A two-dimensional computer simulation for dry etching using Monte Carlo techniques," *Journal of Applied Physics*, vol. 65, no. (4), pp. 1484-1491, February 15, 1989.
24. K.P. Mueller and J. Pelka, "Redeposition in ion milling," *Microelectronic Engineering*, vol. 7, pp. 91-101, North-Holland, 1987.
25. *SAMPLE 1.8 User's Guide*, Electronics Research Laboratory, University of California, Berkeley, 1991.
26. C. Catana, J.S. Colligon, and G. Carter, *Journal of Materials Science*, vol. 7, p. 467, 1972.
27. J. Pelka, M. Weiss, W. Hoppe, and D. Mewes, "Influence of ion scattering on dry etch profiles," *Journal of Vacuum Science and Technology B*, vol. 7, no. 6, pp. 1483-1487, November/December 1989.

28. J. Pelka, "Simulation of Ion-Enhanced Dry-etch Processes," *Microelectronic Engineering*, vol. 13, pp. 487-491, 1991.
29. R.H. Bruce and A.R. Reinberg, *Journal of the Electrochemical Society*, vol. 129, p. 393, 1982.
30. H. Sawin, D. Gray, T. Dalton, and J. Arnold, "Mechanisms of Pattern Dependencies in Plasma Etching Processes," *Proceedings of Sematech SCOE Coordination Meeting*, pp. 167-179, Austin, Texas, April 30, 1991 to May 1, 1991 .
31. A. Yamano, K. Harafuji, M. Kubota, and N. Nomura, "Simulation Study of Ion Energy Distribution in a Plasma Sheath Including Charging Effects," *submitted for publication*, May 1991.
32. K.H. Huebner, *The Finite Element Method for Engineers*, John Wiley & Sons, New York, 1975.
33. P. Tong and J.N. Rossettos, *Finite-Element Method: Basic Technique and Implementation*, MIT Press, Cambridge, Massachusetts, 1977.
34. O.C. Zienkiewicz and R.L. Taylor, "The Finite Element Method," 3rd edition, McGraw-Hill, New York, 1983.
35. R.W. Clough and C.P. Johnson, "A Finite Element Approximation for the Analysis of Thin Shells," *International Journal of Solids Structures*, vol. 4, pp. 43-60, 1968.
36. P.G. Ciarlet, "Conforming Finite Element Methods for Shell Problems," in *Mathematics of Finite Elements and Applications II*, ed. J. Whitemann, Academic Press, 1977.
37. R. Sedgewick, *Algorithms*, p. 65, Addison-Wesley, Reading, Massachusetts, 1983.
38. K.E. Bean, "Anisotropic Etching of Si," *IEEE Transactions on Electron Devices*, vol. ED-25, p. 1185, 1978.



39. W.F. Foote, "Simulation of Anisotropic Crystal Etching," *M.S. Project Report*, University of California, Berkeley, September 12, 1990 .
40. C.H. Sequin, "Microfabrication on the Macintosh," *USENIX Computer Graphics Workshop*, Monterey, California, November 1989.

## CHAPTER 7

### THREE-DIMENSIONAL DEPOSITION MODELING AND SIMULATION

#### 7.1. GEOMETRIC DEPOSITION MODELING

Deposition simulation encounters many of the geometric effects already seen in etching simulation. The global flux distributions and arrival rates of materials depend on process conditions, which are independent of the local surface topography. At a point on the surface, however, shadows and solid angle visibility restrict the arrival of deposited material. Surface orientation may also affect how fast and in what direction deposited layers grow. Particles may migrate along the surface before stopping, or may bounce off to adhere to some other part of the surface. All of these processes involve the interaction of material properties with the surface geometry.

This chapter presents deposition modeling with the algorithms and data structures described in previous chapters. Basic models assume that all material striking the surface sticks. Different material flux distributions and the shadowing effects result in evaporation, and sputter deposition processes. Extensions to the basic models include the effects of surface migration, changes in film column orientation with surface orientation, variations in film density due to the topography, and considerations of processes with surface migration or a sticking coefficient less than one.

Throughout this chapter, a consideration of results from published Monte Carlo simulations of ballistic deposition provides insight into surface advancement models for thin film growth over topographies.<sup>1, 2, 3, 4, 5, 6, 7, 8, 9, 10</sup>

## 7.2. BASIC PHYSICAL VAPOR DEPOSITION

A method for constructing two-dimensional profiles of evaporated films over steps, developed by Blech in 1970,<sup>11</sup> has been used successfully in several simulators.<sup>12, 13</sup> The method is directly applicable to three-dimensional simulation. The construction method begins with some assumptions, which are applicable to many cases. The following are similar to Blech's original assumptions but allow an arbitrary flux distribution. 1) The mean free path of incident particles is larger than the source to substrate distance. 2) The source to substrate distance is larger than the step height. 3) The sticking coefficient is 1.0 throughout the evaporation. 4) The film grows towards the the direction of the vapor stream, and no roughening effects due to large incident angles are considered. 5) The rate of material arriving is a function of incident direction and is proportional to the inverse-square of the distance to the evaporation source.

In three-dimensions, the direction and rate of growth for a point on the surface depend on flux contributions arriving from several directions.<sup>14</sup> The following integrals express the three components of the growth vector (the angles and coordinate directions are the same as those shown in Fig. 6.1):

$$v_x = \frac{R_d}{N_d} \iint_{\Omega} F_d(\phi, \theta) \sin \theta \sin \phi \sin \phi d\phi d\theta \quad (7.1a)$$

$$v_y = \frac{R_d}{N_d} \iint_{\Omega} F_d(\phi, \theta) \cos \theta \sin \phi \sin \phi d\phi d\theta \quad (7.1b)$$

$$v_z = \frac{R_d}{N_d} \iint_{\Omega} F_d(\phi, \theta) \cos \phi \sin \phi d\phi d\theta \quad (7.1c)$$

where  $R_d$  is the deposition rate on a flat surface perpendicular to the  $z$ -axis with no shadowing,  $F_d$  is the flux distribution of arriving particles,  $N_d$  is a normalizing factor of the flux distribution over the hemisphere, and  $\Omega$  is the visible part of the vapor stream. Each differential flux element contributes to growth along the direction toward that element. The overall growth

depends on the relative contribution of each flux element to the whole.

Simple unidirectional evaporation corresponds to replacing  $F_d(\phi, \theta)$  with a delta function along a given direction. The end result is that the film grows only in one direction. If a point is shadowed, the deposition rate there is zero. Fig. 7.1 shows unidirectional evaporation in a contact hole for various source orientations. For 1810 initial triangles, CPU times were approximately 240 seconds in these examples.

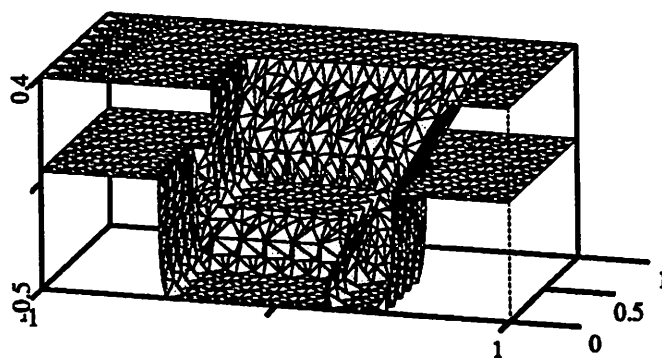
Many evaporation processes show flux distributions distributed over a wide area. Blech's original work assumed the incident particles to arrive equally from all directions. Subsequent work for sputter deposition systems showed a cosine dependence on  $\phi$ ,<sup>13</sup> variations of which have been used by other researchers.<sup>6</sup> A general cosine-based flux distribution function may be expressed as:

$$F_d(\phi) = \cos^n(A\phi) \text{ for } \phi \leq \pi/2A, \text{ otherwise } 0 \quad (7.2)$$

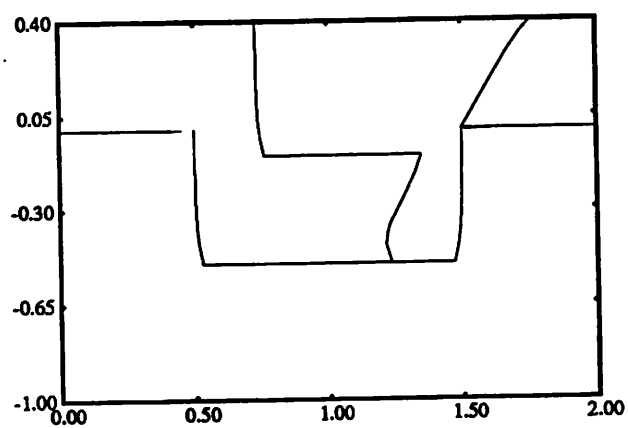
The parameter  $A$  leads to a limiting angle, beyond which no flux arrives. The parameter  $n$  allows over-cosine and under-cosine distributions. Figure 7.2 shows sputter deposition in a contact hole for various  $n$  and  $A$  values. For 1810 initial triangles and 45x10 hemisphere elements, run times were approximately 630 seconds. There is almost no difference when 45x90 hemisphere patches are used, requiring nearly 2 hours!

Recent work has demonstrated flux distributions in sputter deposition systems which do not follow cosine distributions. For example, Park *et. al.* presented a flux distribution for magnetron sputtering of aluminum which showed a strong preferential direction along  $\phi=40^\circ$ .<sup>15</sup> It is thus highly probable that many situations exist where equation 7.2 does not apply. However, equation 7.1 is valid any time the conditions listed at the beginning of this section apply, for any arbitrary flux distribution.

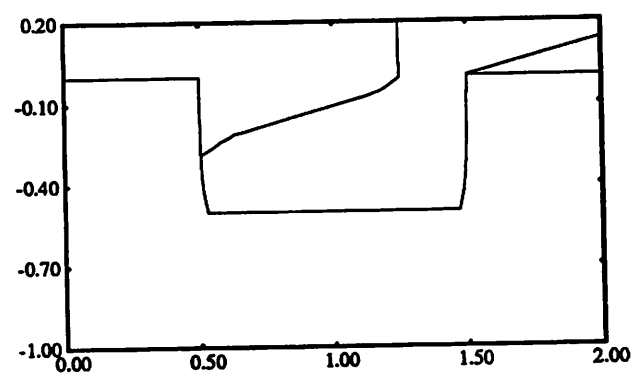
(a)



(b)



(c)



**Fig. 7.1:** Evaporation simulation on a cold substrate. (a) cutaway 3D view  $\phi=30^\circ$ .  
 (b) Cross-section  $\phi=30^\circ$ , (c) Cross-section  $\phi=75^\circ$

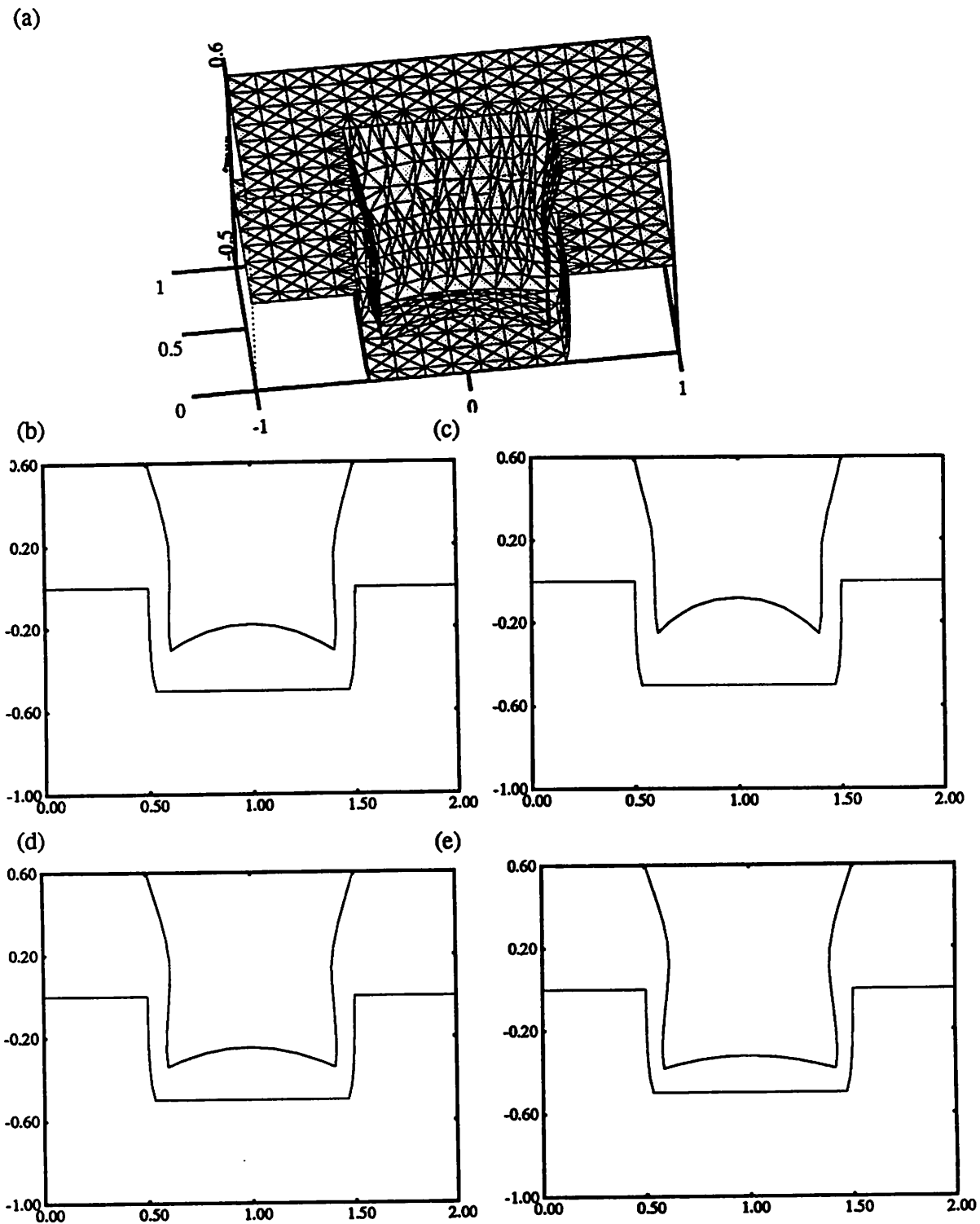


Fig. 7.2: Sputter deposition simulation,  $\text{flux} = \cos^n(A\phi)$ . (a) cutaway 3D view,  $A=1.2, n=1$ .  
 (b) Cross-section  $A=1.2, n=1$ , (c) Cross-section  $A=1.2, n=2$ ,  
 (d) Cross-section  $A=1.2, n=0.5$ , (e) Cross-section  $A=0, n=1$ .

### 7.3. EXTENSIONS TO THE BASIC MODELS

Unfortunately, there are many cases in which simple assumptions are not sufficient to model thin film deposition. For example, it has been suggested that many films do not grow towards the vapor stream, but instead follow a tangent dependence on the angle between the vapor stream and the surface normal.<sup>16</sup> Furthermore, the simple theory does not account for density variation with incident angle nor surface migration effects on heated substrates. Fortunately, it is possible to extend and improve the basic models to include additional observed phenomena.

#### 7.3.1. Column Orientation

Microfractography of evaporated thin films has demonstrated columnar microstructures by Nieuwenhuizen and Haanstra.<sup>16</sup> The columns are not oriented in the direction of the vapor stream for off-normal incidence but instead follow a half tangent rule. According to work by Nakhodkin and Shaldervan, this trend holds true for low temperature substrates where the surface mobility of atoms is limited.<sup>17</sup> For the geometry shown in Fig. 7.3, a vapor stream with angle of incidence  $\alpha$  results in column growth along direction  $\beta$ , where  $\alpha$  and  $\beta$  are related by:

$$\tan\beta = \frac{1}{2}\tan\alpha \quad (7.3)$$

Dirks and Leamy observed similar results and demonstrated, that although many physical mechanisms appear to contribute to column formation, a simple geometric argument based on local shadowing of the vapor stream is sufficient to explain the observed "tangent rule" columnar growth.<sup>18</sup> Two-dimensional simulations based on the packing of hard discs supported their hypothesis.

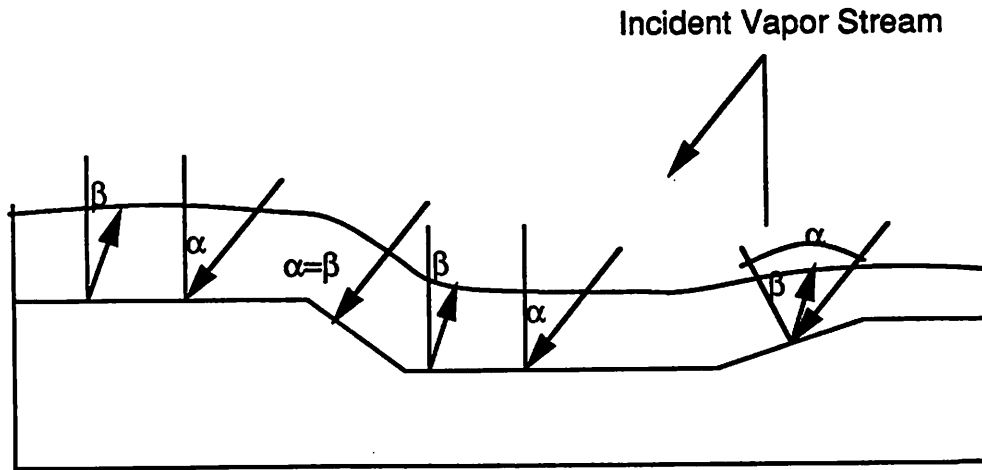


Fig. 7.3: Film columns grow along a different direction than the incident vapor stream for evaporation on cold substrates.

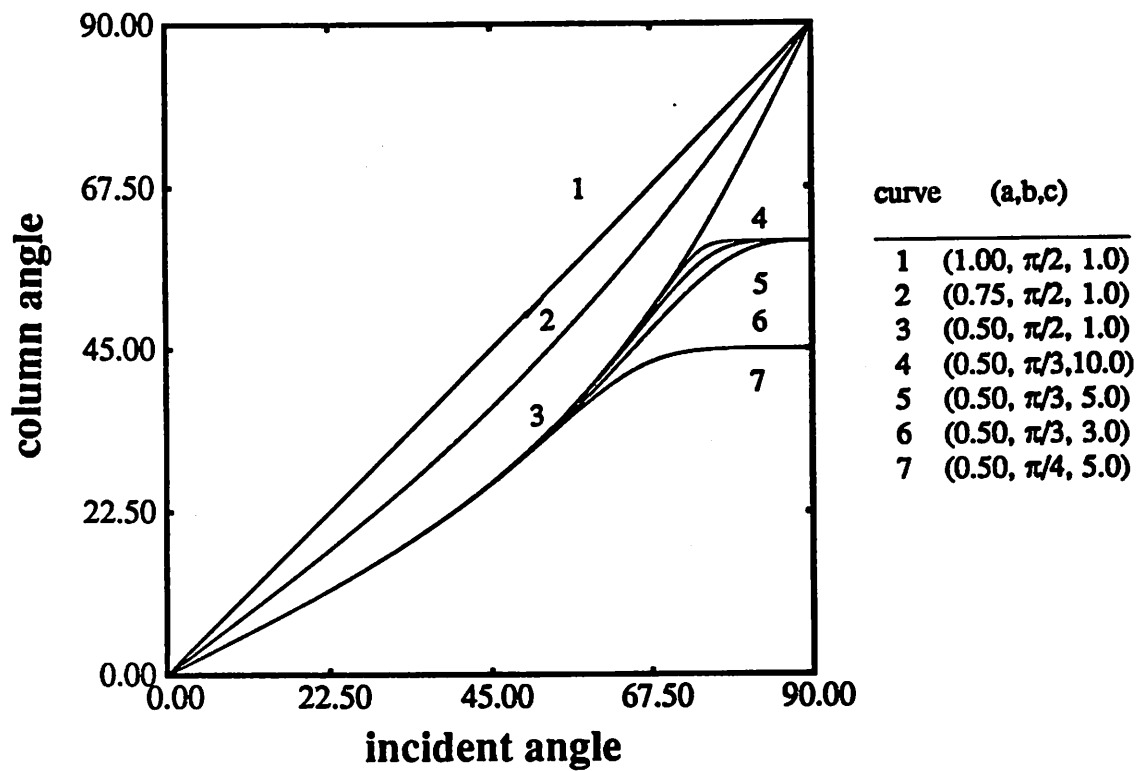


Fig. 7.4: Column orientation vs. incident angle using modified tangent rule.



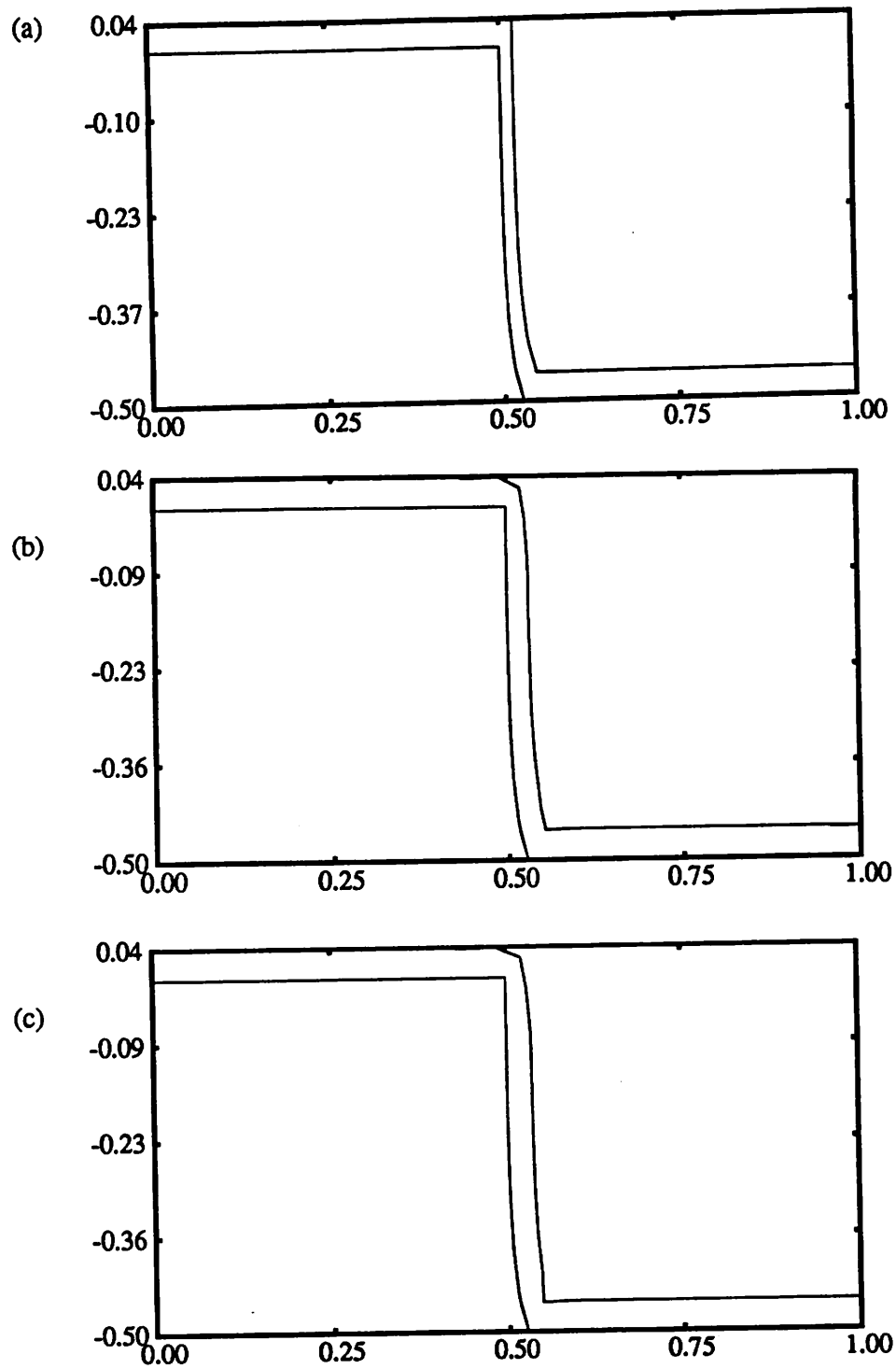
Recently, the simple tangent rule has been challenged.<sup>19</sup> Work by Tait, Smy and Brett using the ballistic deposition simulator SIMBAD and experimental evidence for  $\text{MgF}_2$  thin films showed that  $\beta$  may be less than the value predicted by equation 7.3 for some cases.<sup>7</sup> Dirks and Leamy reported that the tangent rule appears to overestimate  $\beta$  when  $\alpha > 60^\circ$ . This is in agreement with the SIMBAD simulations which show columns within a few degrees of the tangent rule for  $\alpha=41^\circ$ , but deviate significantly from the tangent rule for  $\alpha=78^\circ$ . An extension to the simple tangent rule allows an empirical fit to experimental data. Fig. 7.4 shows plots of  $\beta$  versus  $\alpha$  using the following modified tangent rule:

$$\tan\beta = \frac{\tan B \tan\alpha}{\left[\left(\frac{\tan B}{A}\right)^C + \tan^C \alpha\right]^{\frac{1}{C}}} \quad (7.4)$$

where  $A$  and  $B$  and  $C$  are fitting parameters. The original factor  $(1/2)$  is now given by  $A$ , which describes the curvature of the model for low values of  $\alpha$ . The parameter  $B$  is the maximum value of angle  $\beta$ . The parameter  $C$  determines how slowly the plot deviates from the original model. The original model is retrieved if  $A=1/2$ ,  $B=\pi/2$  and  $C=1$ .

Including the column orientation variation in the 3D simulation requires a determination of the actual direction corresponding to  $\beta$  for a given surface orientation. Rotating the coordinate system so the z-axis lines up with the surface normal gives  $\alpha$  and  $\beta$  as directions with different  $\phi$ , but constant  $\theta$  values. Equation 7.3 or 7.4 is used to find  $(\phi_\beta, \theta_\beta)$  in the new coordinate system. A reverse transformation back to the original coordinate system gives the  $\beta$  direction in the original system.

Blech has suggested that the column orientation phenomenon is not seen when the vapor flux is distributed over a wide area.<sup>20</sup> This is plausible since particles arriving randomly from several directions will not encounter well defined local shadows. There is thus no reason to expect that the geometric effects leading to the tangent rule play a role in anything but low



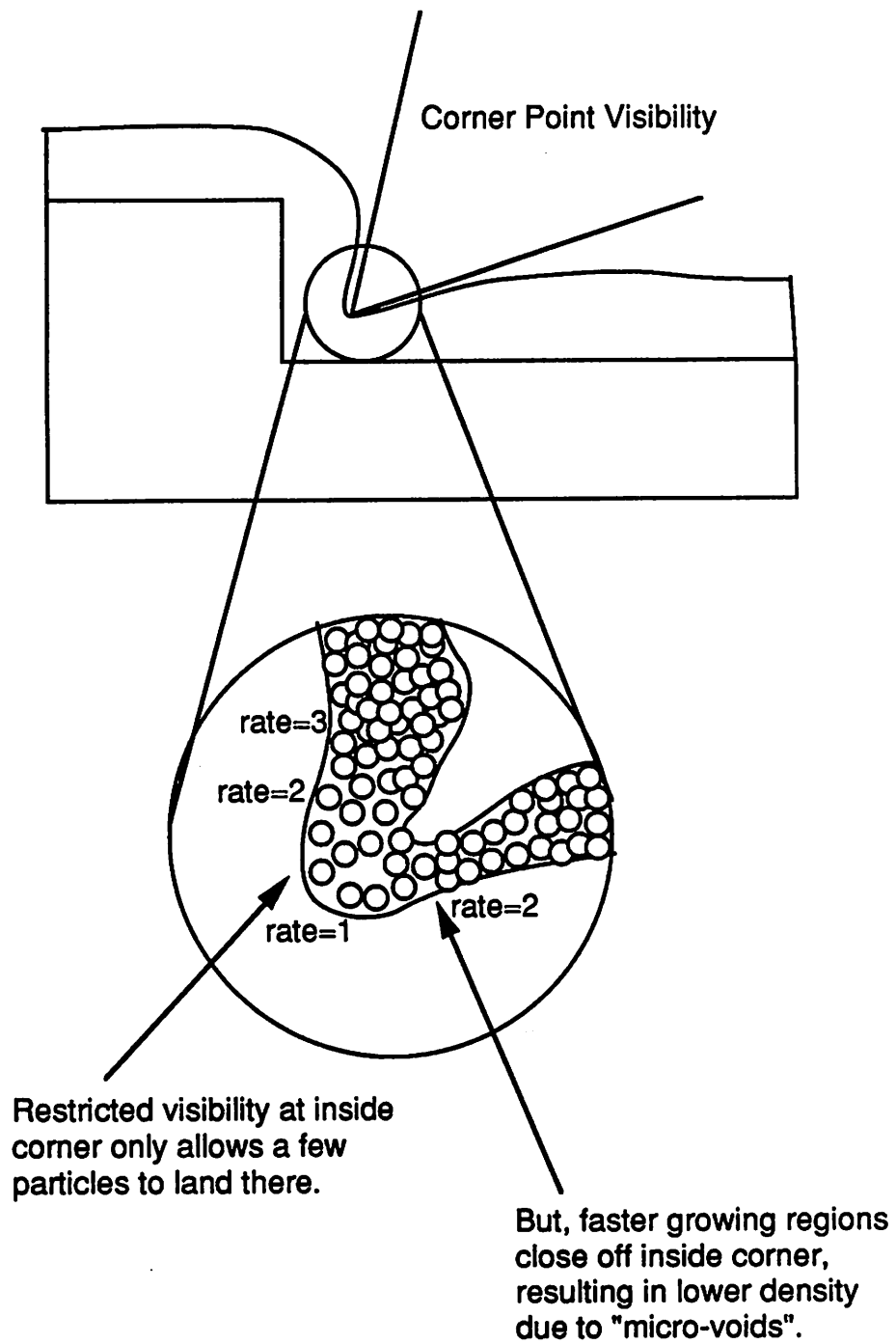
**Fig. 7.5:** Examples of deposition simulation with column variation  $\phi=25^\circ$ . (a)no variation, (b)  $A=0.5, B=90^\circ, C=1.0$ , (c)  $A=0.5, B=45^\circ, C=2.0$ . Cross-section views of line edge profile - columns themselves are not returned by the simulation.

temperature, point source evaporation. Given the limits of validity of the model, three cross-sections of simulations, using equation 7.4 are depicted in Fig. 7.5.

### 7.3.2. Variable Film Density

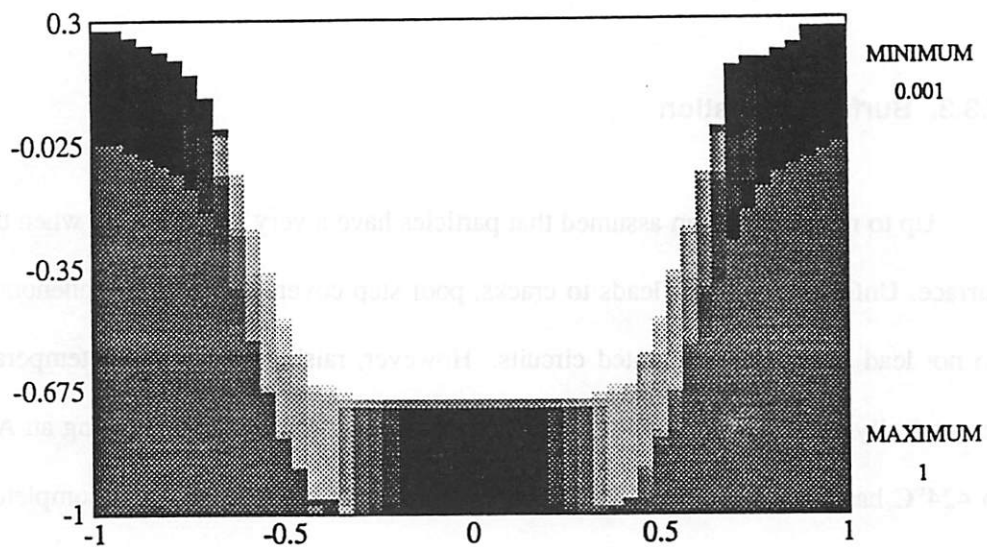
The same microshadowing effects which lead to variations in columnar orientation also lead to variations in film density. It is thus not surprising that density also follows a dependence on the tangent of the angle of incidence.<sup>6, 7, 9</sup> Density variation is also apparent in sputter deposition and uniform evaporation, even when column formation does not follow a tangent rule. In those cases, it is not clear that density variation still follows a tangent rule. A more likely explanation is depicted in Fig. 7.6 In regions such as concave corners, the inside of the corner has a more restricted view of the incoming flux than surface points slightly removed from the corner. The inside corner is continually shadowed by the faster growing neighboring regions. This results in the continuous formation of regions with low density. This simple geometric argument demonstrates how cracks may form at inside corners. Two problems remain for simulation: 1) how is the film thickness affected by density variation, and 2) how does the visibility angle at a point affect the local density?

A first-order model for these effects proceeds as follows. First, the growth rate is taken to be inversely proportional to the film density, although never greater than some maximum rate. The local density is taken to be proportional to the local concavity. Convex corners achieve the maximum possible density. In the case of evaporation, the local density on flat surfaces is proportional to the tangent of angle between the surface normal and the vapor stream. These simple models are implemented using the concavity tests in the facet-motion algorithm. As the surface advances, cells are updated as material. Each time a new cell is updated, it is assigned a density depending on the orientation and concavity of the surface in

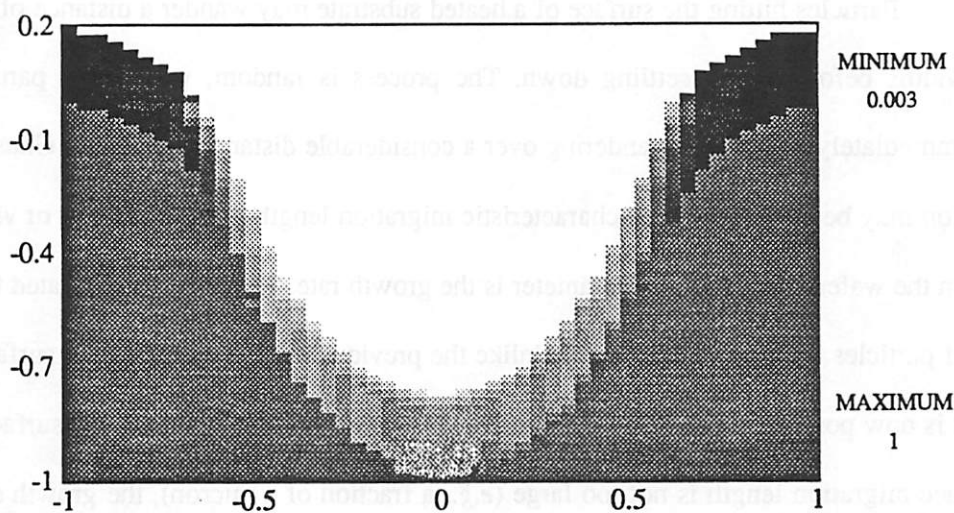


**Fig. 7.6: Density variation in thin-film deposition**

(a)



(b)



**Fig. 7.7:** Examples of deposition simulation with density variation. (a) Evaporation, (b) sputter deposition.

the vicinity of that cell. Fig. 7.7 shows some grey scale plots of contact hole cross-sections for both evaporation and sputter deposition. A low density appears at the bottom of the feature in Fig. 7.7b, due to the microvoids when columns grow towards each other.

### 7.3.3. Surface Migration

Up to now it has been assumed that particles have a very low mobility when they hit the surface. Unfortunately, this leads to cracks, poor step coverage, and other phenomena which do not lead to reliable integrated circuits. However, raising the substrate temperature often dramatically increases the mobility of surface particles. For example, heating an Al-Ge alloy to 424°C has allowed 0.25µm 4:1 aspect ratio contact holes to be filled completely.<sup>21</sup> This effect, in the absence of viscous flow or recrystallization, can be modeled by considering simple surface migration.†

Particles hitting the surface of a heated substrate may wander a distance of many atomic widths before finally settling down. The process is random, with some particles sticking immediately and others wandering over a considerable distance. A simple Gaussian distribution may be assumed with a characteristic migration length. From the point of view of a point on the wafer, the important parameter is the growth rate. This is directly related to the number of particles arriving at that point. Unlike the previous models assuming no surface migration, it is now possible for some of the particles to arrive from other parts of the surface. If the surface migration length is not too large (*e.g.* a fraction of a micron), the growth direction may be assumed to be the same as for the case with no surface migration. However, if the surface

---

† Similar effects are seen in some oxide deposition systems, although it is not clear that this is due to migration of oxide, or some other catalyst species. Only surface migration in metal thin films will be considered here.

migration length is large, the film will tend towards isotropic growth.

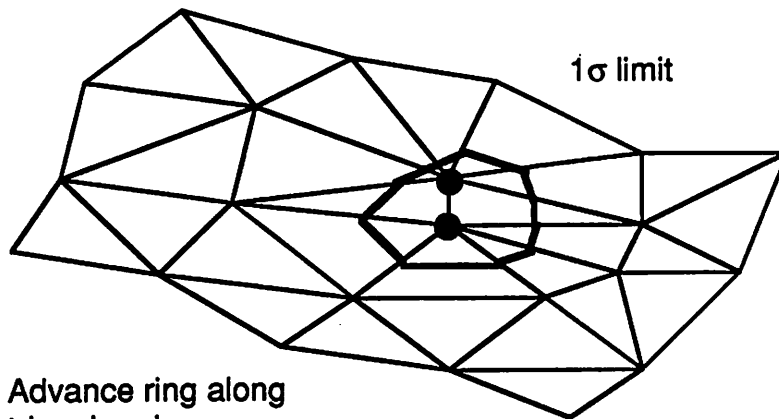
Basic surface migration may be simulated by means of a simple two-step model. First, the flux arrival for every surface point is made using equation 7.2. Then the redistribution is calculated by applying an equation of the form:

$$\mathbf{v}' = \frac{1}{N} \sum_{i=0}^n \exp\left[-\frac{s_i^2}{2\sigma^2}\right] \mathbf{v}_i \Delta A_i \quad (7.5)$$

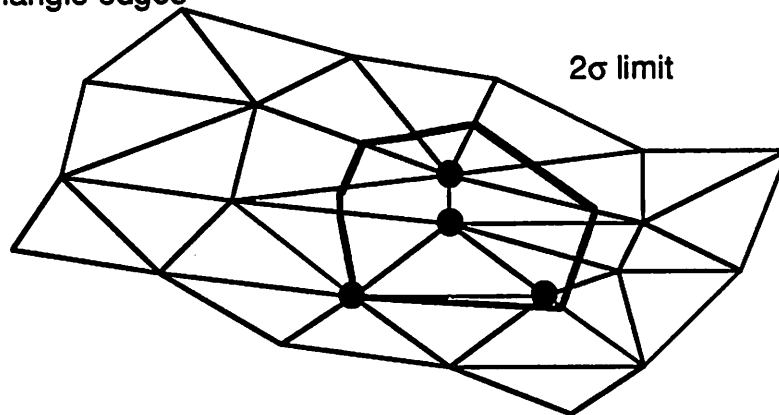
The above equation is a simple discretization of Gaussian convolution where  $s_i$  is the distance on the surface from node  $i$  to the node of interest,  $\sigma$  expresses the surface migration characteristic length,  $n$  is the number of surface points, and  $\Delta A_i$  is an area element giving a region of influence for  $\mathbf{v}_i$ .  $N$  is a normalization factor, which should be calculated separately for each point, to compensate for numerical inaccuracy. The vectors represent length only. Growth is assumed to continue along the original direction, but at a different rate.

In practice, it is not necessary to average over all the surface mesh nodes, but only over those that are within  $3\sigma$  of the node of interest. For a characteristic migration length greater than the minimum segment length, determining the range of points within  $3\sigma$  is complicated by the difficulty of calculating the distance along the curved surface between any two points. For the purposes here, the problem is how to find all the points within a  $3\sigma$  distance of a point of interest, and the distances to each point. An approach to this problem is to construct a circular front emanating outwards along the surface from the point of interest, as shown in Fig. 7.8. As it passes by a mesh node, the radius of the circle is stored in the node. When the radius reaches  $3\sigma$ , all the nodes of interest will have been found and the evaluation of equation 7.5 may proceed. The etch front may be represented as a series of intersections of the front with mesh triangle edges and the front orientations at those intersections. The propagation must be performed iteratively, incrementing a distance no greater than that to the next node at any one

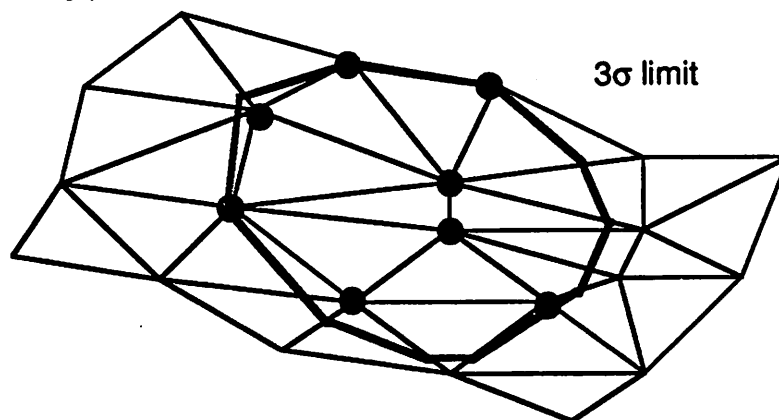
Use a ring approximating the locus of points a given surface distance from a point of interest.



Advance ring along triangle edges



Flag points as ring passes them



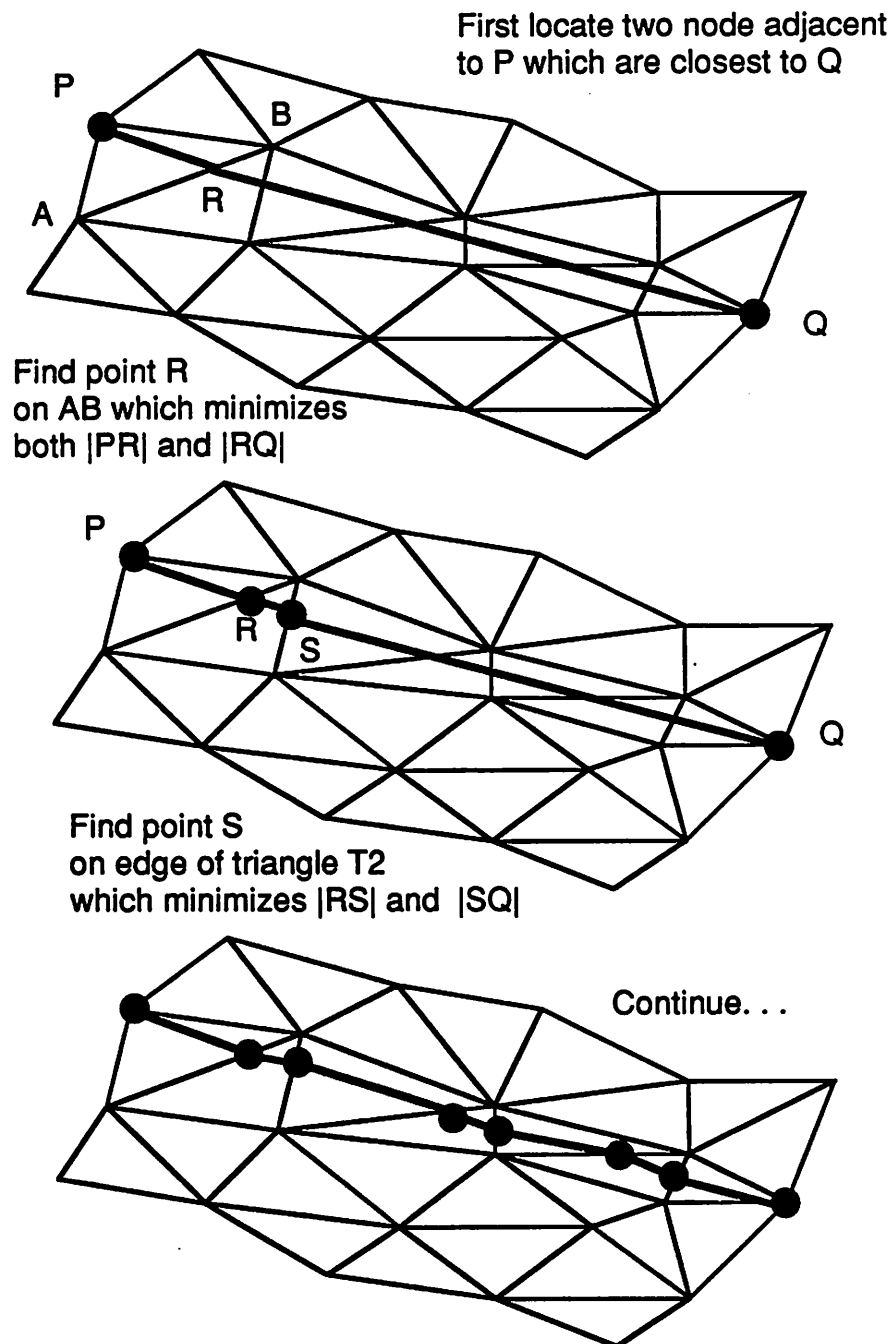
**Fig. 7.8:** A method for locating nodes within  $3\sigma$  distance from a node of interest.



step. The number of required iterations equals the number of nodes within a  $3\sigma$  radius of the point of interest. Care must be taken to calculate the correct advancement distance along the mesh edges, so the polygon correctly approximates a circular etch front. Some error is introduced by this method, but for well-behaved surfaces within a  $3\sigma$  distance, this error is likely to be within acceptable limits.

Another approach for calculating the surface distance between two points, may be used if the surface is not too rugged. An algorithm for this calculation relies on traversing the path between an initial node  $P$  and a final node  $Q$ , and measuring the distance. From the initial point  $P$ , shown in Fig. 7.9, locate the two neighbor nodes,  $A$  and  $B$ , which have a minimum absolute distance to the final point. Then locate the point  $R$  along the segment  $\overline{AB}$ , which minimizes both  $\overline{PR}$  and  $\overline{RQ}$ . From this new point  $R$ , find a point  $S$  on one of the other segments of the triangle containing  $R$  but not containing  $Q$ , such that  $\overline{RS}$  and  $\overline{SQ}$  are minimized. Continue until node  $Q$  is reached. The sum of the distances of line segments spanning the triangles from  $P$  to  $Q$  gives an accurate measure of the surface distance. This procedure must be performed for every mesh node within a  $3\sigma$  distance of the point of interest. Unfortunately, cases may arise where the shortest path is not the one taken by the algorithm, however these cases do not occur frequently in practice. For complicated surfaces however, a better surface distance algorithm is needed. Perhaps a Monte Carlo random walk approach would be best.

Results for sputter deposition with surface migration are shown in Fig. 7.10, for several different values of  $\sigma$ . The step coverage improves with an increasing  $\sigma$ . Crude approximations of the area elements and surface distances were used to generate the examples. The CPU time for 1810 initial triangles, 45x10 hemisphere elements and  $\sigma = 0.2\mu\text{m}$  was 1948 seconds.



**Fig. 7.9:** A method for calculating the surface distance between two nodes.

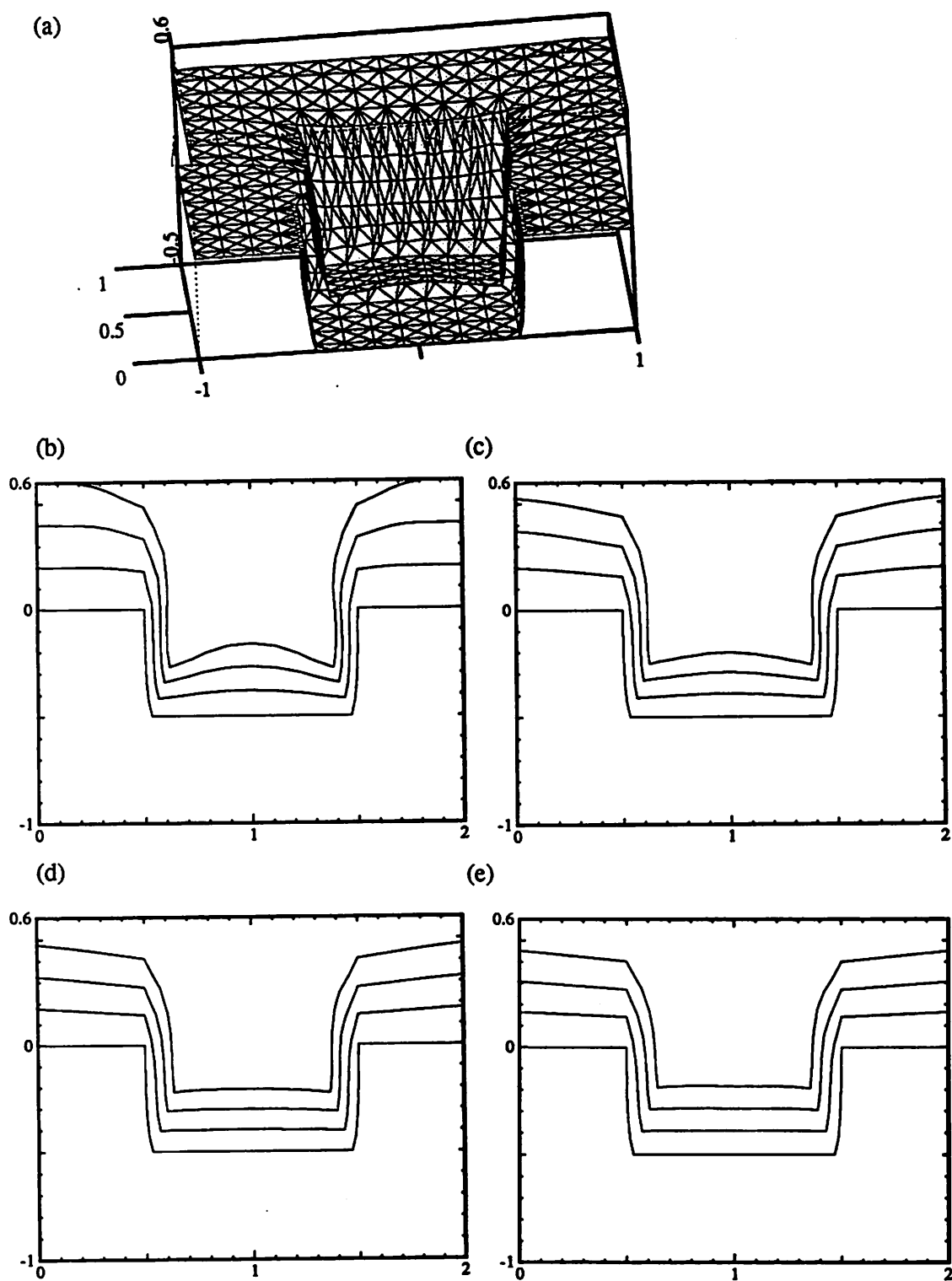


Fig. 7.10: Sputter deposition simulation, flux= $\cos(1.2\phi)$ . (a) cutaway 3D view,  $\sigma=0.2$ , (b)  $\sigma=0.1$ , (c)  $\sigma=0.2$ , (d)  $\sigma=0.3$ , (e)  $\sigma=0.4$ .

A model for surface migration in physical vapor deposition has been proposed by Cale *et. al.*,<sup>22</sup> which uses a diffusion equation very similar in form to that used by G  rodolle and Pelletier for plasma etching simulation, although with different physical constants.

$$D \frac{\partial^2 \zeta}{\partial s^2} + \eta - k \zeta = 0 \quad (7.6)$$

The variable  $\zeta$  is the surface concentration of mobile adatoms,  $\eta$  is the incoming flux,  $k$  is a rate constant,  $D$  is the diffusion constant, and  $s$  is a length along the surface. The diffusion equation gives the surface concentration of mobile atoms as a function of position at any time in the simulation process. Although the model provides a physical framework for physical vapor deposition,  $k$ ,  $D$ , and the relationship of  $\zeta$  to deposition rate are not known. The model also does not account for column orientation or density variation effects when the diffusivity is low, and does not indicate the degree to which a deposition process is isotropic or directional. Nevertheless, the model may be extended into three dimensions following a development similar to that used in chapter 6 for plasma etching. It might also be possible to consider local temperature variations due to shadow effects with extensions of the above model.

#### 7.3.4. Reflection, and CVD Models

If particles reflect off of the surface before becoming part of the film, then equation 7.2 must be extended to include this effect. Chemical vapor deposition processes appear to include this phenomenon, and McVittie *et. al.* have demonstrated 2D LPCVD simulations by including reflection and a variable sticking coefficient in their models.<sup>23</sup> For a sticking coefficient less than one, particles may hit and reflect several times before settling down. This may be calculated by first determining the directly incident flux arriving at each surface point for a time step. Then all the surface points visible from each point must be determined. Each surface point is now a source which emits a certain amount of material depending on the

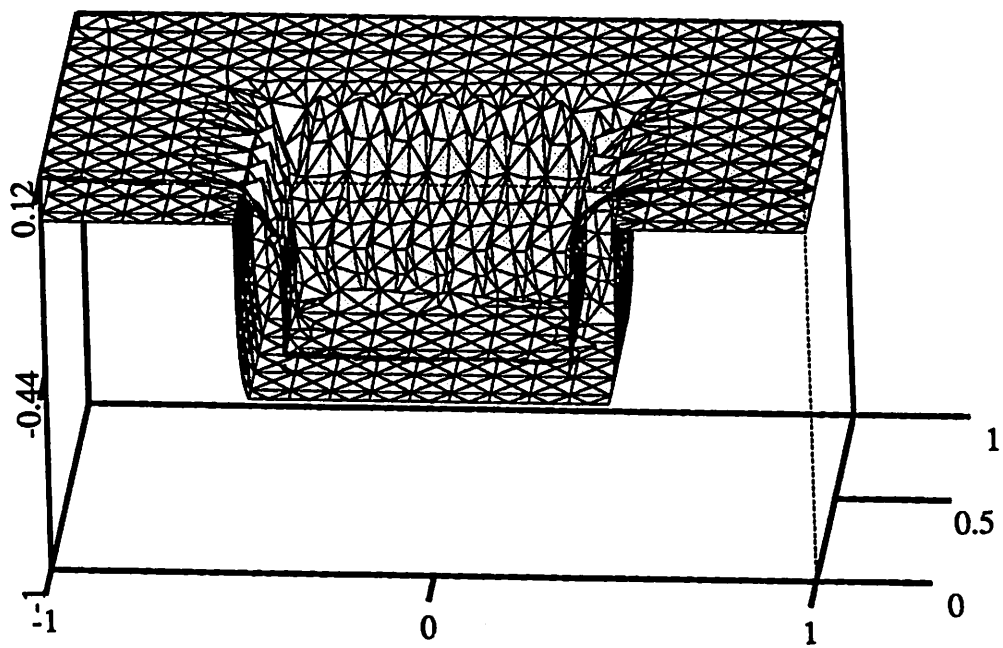


Fig. 7.11: Isotropic deposition to simulate a CVD process.

sticking coefficient and the amount of material received. For a small sticking coefficient, this step must be repeated several times until the etch rate difference between iterations at any point is less than some threshold value. this is a very computationally intensive problem in 3D.

In the limit of an infinitesimally small sticking coefficient, the deposition process becomes almost perfectly isotropic. In this case it is not necessary to calculate the full 3D reflection. Instead, simple isotropic deposition gives results as shown in Fig. 7.11.

#### **7.4. SUMMARY AND CONCLUSION**

Topography simulation of thin film deposition is possible with a relatively small set of surface advancement algorithms and visibility functions. The main difficulty is in the modeling of phenomena which are directly related to the microstructure of thin films, since the surface representation does not contain that information. The cells may be used to record information such as density variation, but only in the presence of a suitable model relating the surface topography, flux distributions, and materials to changes in microscopic properties.

Monte Carlo ballistic deposition can answer many of the questions which surface advancement algorithms do not address directly. M. Brett and colleagues at the University of Alberta, have demonstrated good agreement with simulation and experiment for many deposition processes using SIMBAD. One might ask, why bother with surface advancement simulation at all? The surface advancement algorithms used here have some useful advantages. First, they are much faster than full 3D ballistic simulation models. Second, they directly provide information about the overall surface shape, which is of interest to IC technologists. Third, they allow phenomenological and empirical models to be incorporated into 3D

simulation, even if those models do not contain microscopic information.

Perhaps the interaction of ballistic models and surface advancement simulations should be seen in analogy to the interaction of device and circuit simulation. Device simulators allow the solution of fundamental relationships in semiconductor physics, although at the expense of significant computation time. The result is a collection of current versus voltage data for the device. This data is then converted to compact analytic models for use in circuit simulation. The device model may be used thousands of times in a circuit simulator, thus it is important not to have to run the device simulator each time information about the device is needed. In a similar way, a ballistic deposition simulator may provide insight into specific modes of film growth. From this insight, more compact and generalized models may be developed for use in surface advancement simulation. Indeed, the many published simulation examples from the University of Alberta using SIMBAD, have been a particular source of inspiration for the simple deposition models implemented in this work.

Another problem raised in this chapter is the issue of surface migration on a curved surface, which is mathematically and geometrically related to the surface diffusion problem in plasma etching simulation. Given the difficulty of implementing an accurate surface diffusion algorithm using Gaussian convolution, it may be worthwhile to perform a full finite element calculation of the diffusion equation. Although SAMPLE-3D does not at this time include a finite element solver, it may prove a worthwhile addition since it can be applied to both etching and deposition simulation.

## REFERENCES

1. M.J. Brett, "Structural transitions in aggregation simulation of thin film growth," *Journal of Vacuum Science Technology A*, vol. 6, p. 1749, 1988.
2. M.J. Brett, K.L. Westra, and T. Smy, *Proceedings International Electron Device Meeting*, pp. 336-339, San Francisco, CA, December 1988.
3. K.L. Westra, T. Smy, and M.J. Brett, *IEEE Electron Device Letters*, vol. EDL-10, p. 198, 1989.
4. M.J. Brett, "Simulation of structural transitions in thin films," *Journal of Materials Science*, vol. 24, p. 623, 1989.
5. T. Smy, R.N. Tait, K.L. Westra, and M.J. Brett, "Simulation of Density Variation and Step Coverage for Via Metallization," *Proceedings Sixth International VLSI Multilevel Interconnection Conference*, pp. 292-298, Santa Clara, CA, June 12-13, 1989.
6. T. Smy, K.L. Westra, and M.J. Brett, "Simulation of Density Variation and Step Coverage for a Variety of Via/Contact Geometries Using SIMBAD," *IEEE Transactions on Electron Devices*, vol. ED-37, no. 3, pp. 591-598, March 1990.
7. R.N. Tait, T. Smy, and M.J. Brett, "A Ballistic Deposition Model for Films Evaporated Over Topography," *Thin Solid Films*, vol. 187, Elsevier/Sequoia, The Netherlands, 1990.
8. T. Smy, R.N. Tait, and M.J. Brett, "Ballistic Deposition Simulation of Via Metallization Using a Quasi-Three-Dimensional Model," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. CAD-10, no. 1, pp. 130-135, January 1991.



9. R.N. Tait, S.K. Dew, T. Smy, and M.J. Brett, "Density Variation of Tungsten Films Sputtered Over Topography," *Journal of Applied Physics (submitted)*, 1991.
10. S.K. Dew, T. Smy, and M.J. Brett, "Thin Film Microstructure Simulation of RF Bias Planarized Metal Interconnects using a Ballistic Deposition Model," *Proceedings Eighth International VLSI Multilevel Interconnection Conference*, pp. 353-355, Santa Clara, CA, June 11-12, 1991.
11. I.A. Blech, "Evaporated Film Profiles Over Steps in Substrates," *Thin Solid Films*, vol. 6, pp. 113-118, Elsevier Sequoia S.A., 1970.
12. A.R. Neureuther, C.H. Ting, and C.-Y. Liu, "Application of Line-Edge Profile Simulation to Thin-Film Deposition Processes," *IEEE Transactions on Electron Devices*, vol. ED-27, no. 8, pp. 1449-1455, August 1980.
13. I.A. Blech and H.A. VanderPlas, "Step coverage simulation and measurement in a dc planar magnetron sputtering system," *Journal of Applied Physics*, vol. 54, p. 3489ff, 1983.
14. P. Thoren, I.W. Rangelow, R. Kassing, and P. Kuechner, "Three-dimensional simulation of sputter deposition processes for sub-um technology," *Microelectronic Engineering*, vol. 8, North-Holland, 1988.
15. Y.H. Park, A.H. Chung, and M.A. Ward, "Step Coverage Evaluation of Copper Films Prepared in Magnetron Sputtering," *Proceedings Eighth International VLSI Multilevel Interconnection Conference*, pp. 295-297, Santa Clara, CA, June 11-12, 1991.
16. J.M. Nieuwenhuizen and H.B. Haanstra, "Microfractography of thin films," *Philips Technical Review*, vol. 27, no. 3/4, pp. 87-91, 1966.
17. N.G. Nakhodkin and A.I. Shaldervan, "Effect of vapour incidence angles on profile and properties of condensed films," *Thin Solid Films*, vol. 10, p. 109, 1972.

18. A.G. Dirks and H.J. Leamy, "Columnar Microstructure in Vapor-Deposited Thin Films," *Thin Solid Films*, vol. 47, pp. 219-233, 1977.
19. P. Meakin, P. Ramanial, L.M. Sander, and R.C. Ball, "Ballistic deposition on surfaces," *Physics Review A*, vol. 34, p. 5091, 1986.
20. I.A. Blech, "Step Coverage of Vapor Deposited Thin Aluminum Films," *Solid State Technology*, pp. 123-125, December 1983.
21. K. Kikuta, T. Kikkawa, and H. Aoki, "0.25  $\mu$ m Contact Hole Filling by Al-Ge Reflow Sputtering," *1991 Symposium on VLSI Technology, Digest of Technical Papers*, pp. 35-36, Oiso, Japan, May 28-30, 1991.
22. T.S. Cale, T.H. Gandy, M.K. Jain, M. Ramaswami, and G.B. Raupp, "A General Model for PVD Deposition," *Proceedings Eight International VLSI Multilevel Interconnection Conference*, pp. 350-352, Santa Clara, June 11-12, 1991.
23. J.P. McVittie, J.C. Rey, L.Y. Cheng, M.M. IslamRaja, and K.C. Saraswat, "LPCVD Profile Simulation Using a Re-Emission Model," *IEDM Technical Digest*, pp. 917-920, San Francisco, CA, December 1990.

## CHAPTER 8

# SYSTEM ORGANIZATION, USER-INTERFACE, & INTEGRATION

### 8.1. TOPOGRAPHY SIMULATION SOFTWARE ORGANIZATION

Practical 3D topography simulation must bring together algorithms, data structures, physical models, and a user interface in a coherent and consistent manner. Describing the software as a hierarchical organization of operations leads to a clear understanding of the program structure. The highest level view considers topography simulation in a general form consisting of a wafer state, models to change the wafer state, and algorithms to implement the models. At the lowest level, the program consists of several operations, classified by functionality. SAMPLE-3D is organized in a hierarchical and modular way. This organization lends itself to modification and continued development, so that SAMPLE-3D provides a software platform for exploring new algorithms and physical models. A user interface exists as a separate, additional module.

This chapter first gives the organization of SAMPLE-3D by describing the overall program flow, showing how K.K.H. Toh's ray-trace lithography simulation is included, by listing the operations classified by functionality, and by describing data structure and global variable requirements to support the operations. The procedures for implementing new physical models and algorithms, and for estimating CPU and memory requirements for various models follow. The chapter continues with user interface issues, considered from the perspective of both programmers and users. Suggestions for future integration with other technology CAD software, discussions of the relationship of SAMPLE-3D to the CAD Frameworks Initiative TCAD Semiconductor Wafer Representation effort, and proposals for the continued

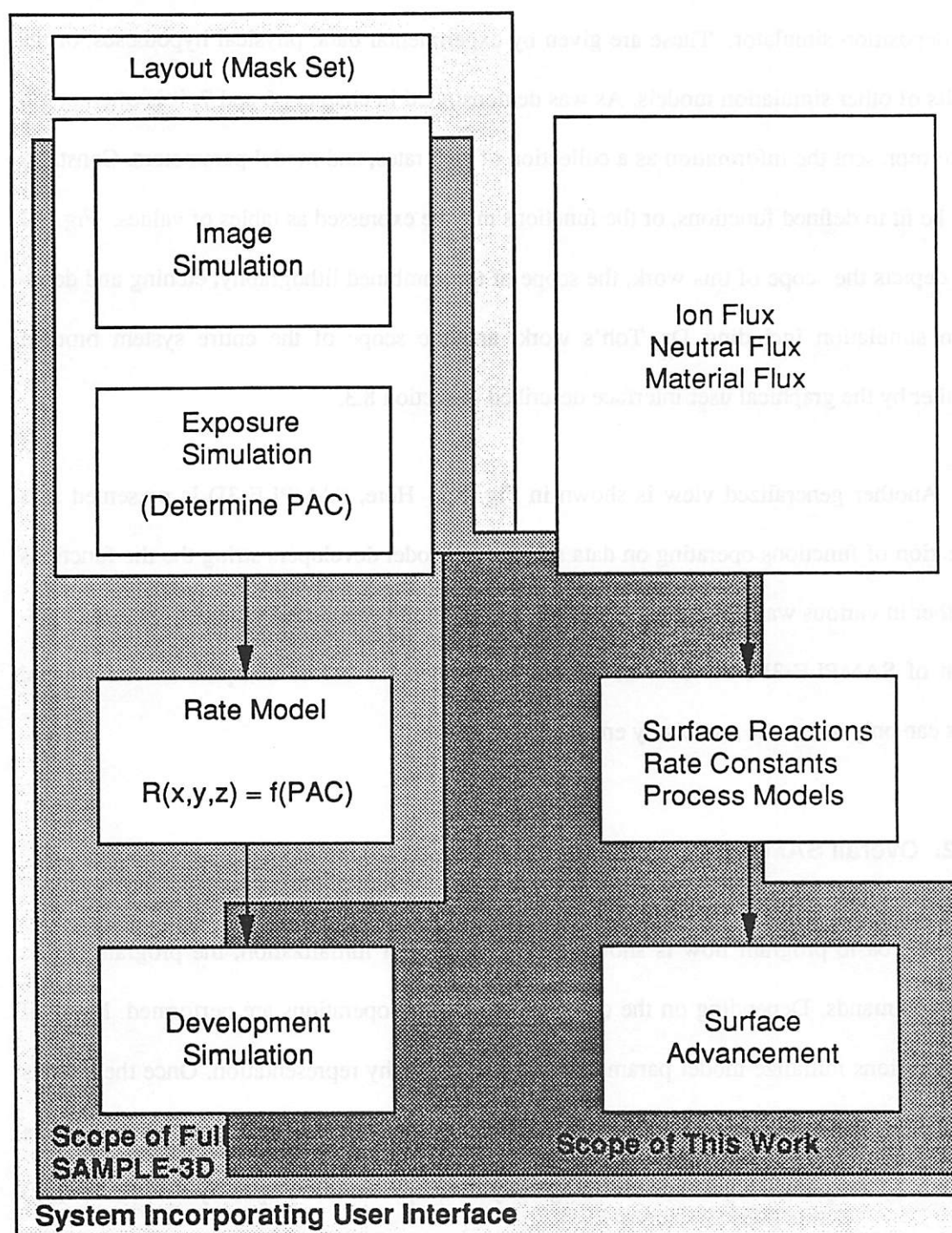
development of SAMPLE-3D as a functional library are also presented.

## **8.2. SAMPLE-3D ORGANIZATION**

SAMPLE-3D can be viewed at three levels of abstraction. From most general to most specific: 1) SAMPLE-3D maintains a wafer state representation and alters it according to process models. 2) SAMPLE-3D consists of a well-defined program flow for executing operations on a wafer state representation. 3) SAMPLE-3D is a collection of surface evolution, visibility, and geometric operations for coupling process models to the advancement of a topography representation. These views, taken together with details on the data structure, provide a complete description of the software. The program organization provides the context for implementing physical models and estimating CPU and memory requirements for specific cases.

### **8.2.1. A High Level View**

Fig. 8.1 represents a general view of etching and deposition simulation and compares it with lithography simulation, as organized by K.K.H. Toh and others.<sup>1</sup> The parallels are evident. Lithography simulation begins with an aerial image which exposes a photoresist. Rate models convert the exposed photoresist to an array of etch rates distributed in space, which are available as input to a development (wet-etching) simulator. Instead of an image, general dry-etch and deposition simulation uses distributions of particle fluxes. Instead of an exposure model, information about materials and surface topography relates the fluxes to the etch rates. The etching and deposition surface advancement module is analogous to the development simulator in lithography.



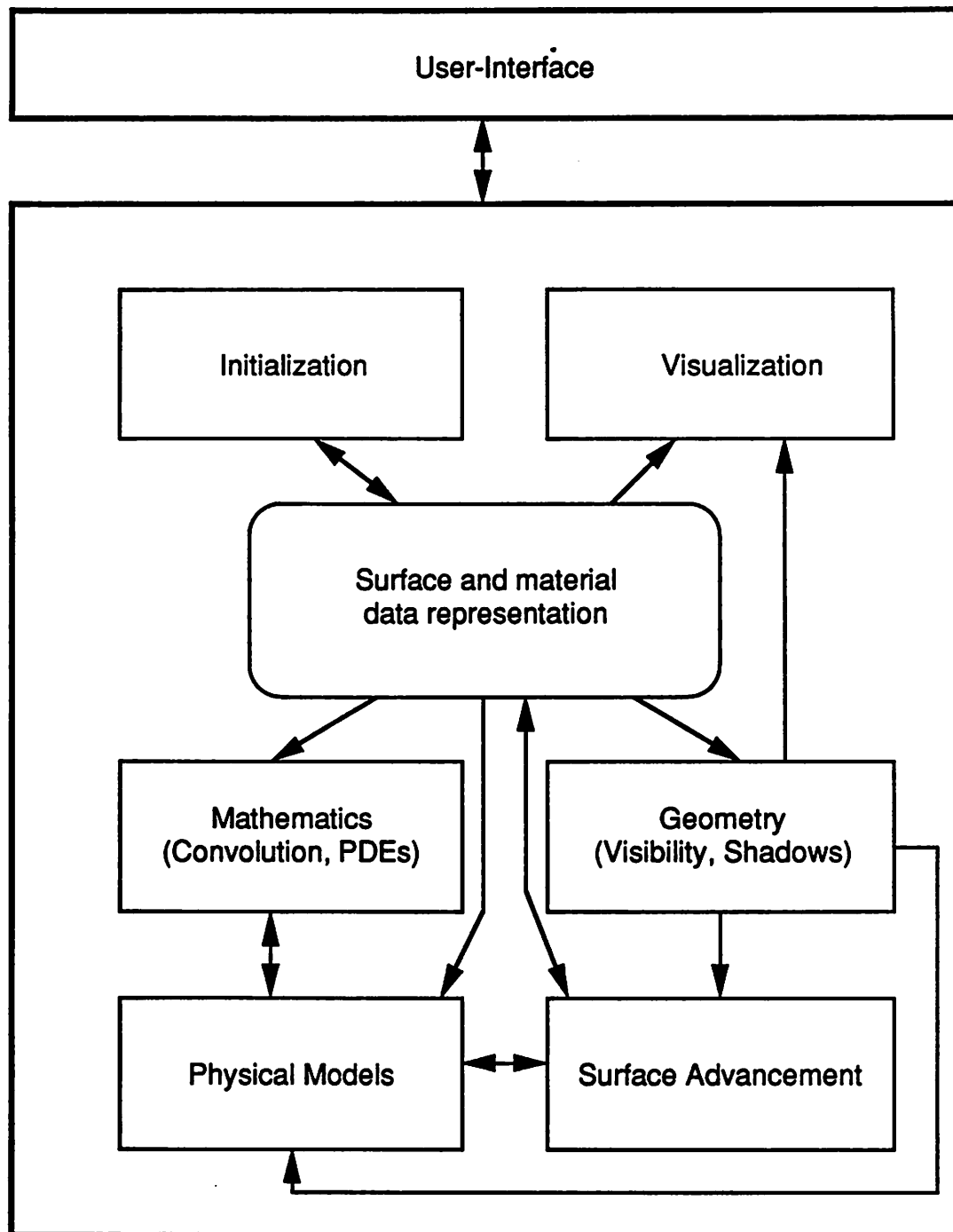
**Fig. 8.1:** A general view of deposition and etching simulation compared with lithography simulation. Outlines show scope of this work, and the full SAMPLE-3D system including the work of K.K.H. Toh and the user interface.

The flux distributions and material information are external to the SAMPLE-3D etching and deposition simulator. These are given by experimental data, physical hypotheses, or the results of other simulation models. As was demonstrated in chapters 6 and 7, it is often possible to represent the information as a collection of etch rates, and model parameters. Constants may be fit to defined functions, or the functions may be expressed as tables of values. Fig. 8.1 also depicts the scope of this work, the scope of the combined lithography, etching and deposition simulation including Dr. Toh's work, and the scope of the entire system brought together by the graphical user interface described in section 8.3.

Another generalized view is shown in Fig. 8.2. Here, SAMPLE-3D is presented as a collection of functions operating on data structures. Model developers string the the functions together in various ways to implement process simulation capability. In the current implementation of SAMPLE-3D, the flow of information is decided before compilation and general users can only affect the models by entering parameters

### **8.2.2. Overall SAMPLE-3D Program Flow**

The basic program flow is shown in Fig. 8.3a After initialization, the program parses input commands. Depending on the commands, different operations are performed. Many of the operations initialize model parameters or the topography representation. Once the models are entered, the program may execute a "run" command. Three different run operations are possible: 1) a ray-trace algorithm for lithography development (developed by K.K.H. Toh), 2) a facet-motion algorithm for general etching and deposition simulation, or 3) a cell-removal algorithm for lithography development (from chapter 3). The particular run operation is chosen by the user, although the ray-trace and cell removal algorithms are only valid for lithography development simulation. Fig. 8.3b presents the three branches in greater detail.



**Fig. 8.2:** Modular organization of functions for 3D process simulation, showing possible paths for flow of information.

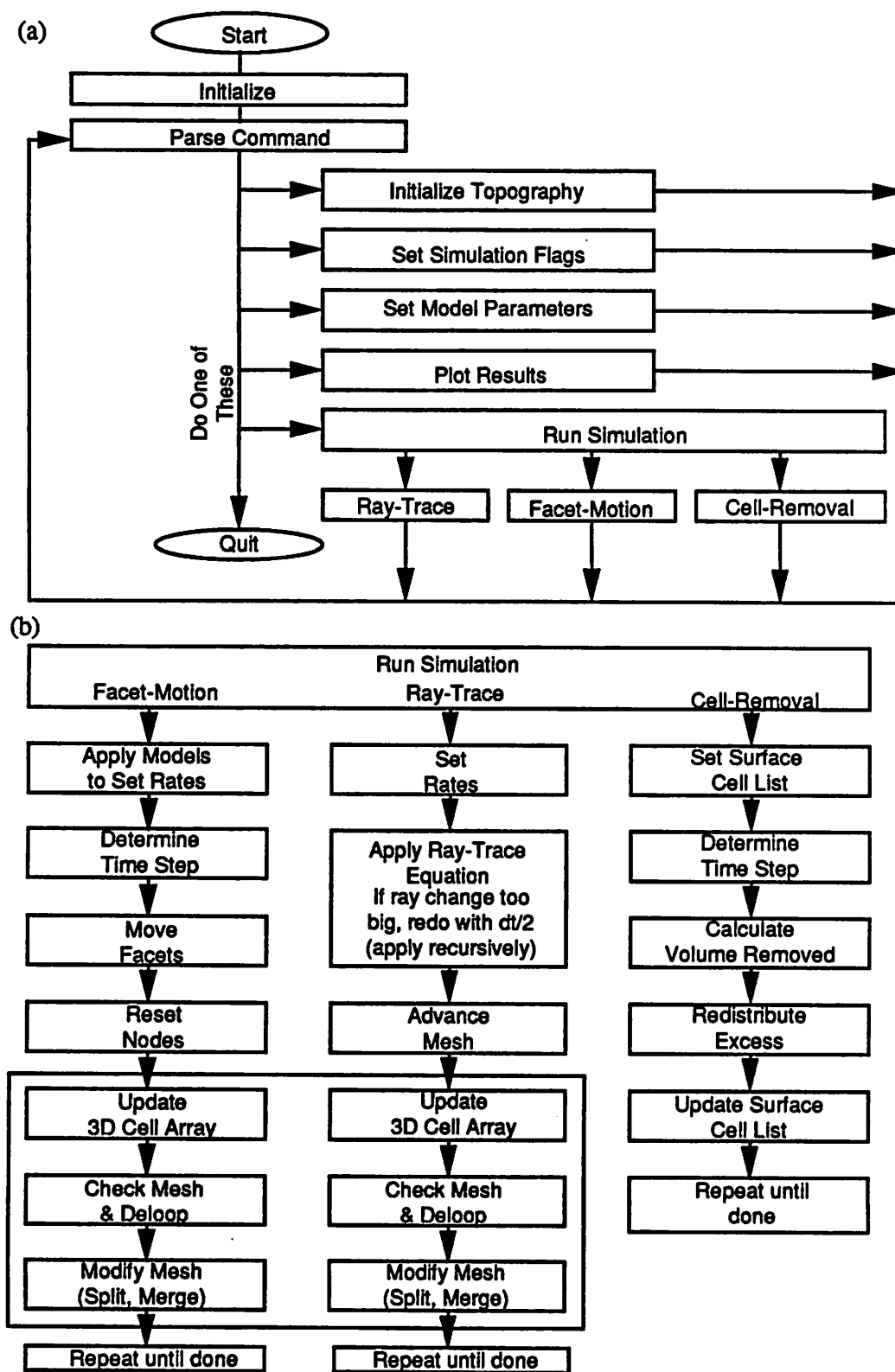


Fig. 8.3: (a) SAMPLE-3D program flow. (b) Detail of program flow for run commands.



Each consists of a loop wherein rates are derived from a process model and applied to a topography deformation algorithm. Each also includes modules for refining and updating the topography representation. Many of the modules in the ray-trace and facet branch are the same, indicating how common functions are re-used to support various algorithms.

The physical models are connected with the algorithms in the step in which the rate for each surface element node is set. In the ray-trace and cell-removal algorithms, setting the rates is simply a matter of locating the etch rate at each surface element. The facet-motion algorithm allows a more general description of the node movement. At each time step, each node requires a rate, each facet requires a direction, and the rate along the facet direction must be related to the rate at the node. In some simple cases, it is possible to ignore the facets and only supply the rate and direction at each node.

Once the rate information is fully specified, the facets are advanced and the surface is reconstructed. In the general case, the full surface motion algorithm must be applied to reconstruct the surface. For the isotropic or simple directional case, a subset of the full facet-motion algorithm is all that is needed. In the simple case, where the node velocity is given directly, the new surface is already properly constructed.

### **8.2.3. Operations In SAMPLE-3D**

Operations are organized according to functionality and complexity. SAMPLE-3D includes 6 major classes of operations.

*High Level Geometric Operations:* These include shadow calculation, visibility determination, integrals over visible solid angles, Gaussian convolution, and simple reflection.

***Mesh and Cell Operations:*** These are operations for advancing and maintaining the surface mesh, including, ray-trace surface advancement, facet-motion surface advancement (for simple, isotropic, directional, and general advancement), segment splitting and merging, clip at boundary, basic deloop, and update cells with advancing surface. (ray-trace, and basis for segment splitting, merging, and clipping developed by K.K.H. Toh).

***Low Level Geometric Operations:*** The high level operations are built out of more basic operations such as, triangle-triangle intersection, line-triangle intersection, point in triangle (on a plan), point in volume test, cell containing point, layer containing point, distance along surface, and direction contained in solid angle.

***Vector Operations:*** These include dot product, cross product, length of a vector, vector sum, vector difference, scalar multiplication, advance along a direction (multiply and add), normalize vector, test if two vectors are equal, equate vectors, midpoint of two points, and angle between two vectors.

***Plotting:*** Routines to generate plots read the data representation and provide information to plotting programs. SAMPLE-3D has 3D plots, 2D slices, 3D and 2D plots of the cell array, and plots of shadow or solid angle flux as grey scale values on the surface.

***Initialization and Management:*** These include data structure initialization, mask initialization, model initialization, error-checking, data structure and memory management, input command parsing, simulation flow management, and other miscellaneous commands.

Physical models apply low and high level operations to external and topography information in order to describe the surface evolution over a small time step. SAMPLE-3D has models for isotropic etching, plasma etching and ion milling, and deposition including many

second-order effects.

The operations themselves have specific implementations built around the data structures in SAMPLE-3D. Low level routines tend to be completely modularized, in that they only act on explicitly provided arguments. The high level operations generally require access to many global variables, not the least of which is the surface mesh itself. In the course of developing SAMPLE-3D, many functions have been written which are specific to a particular model or particular task. Some functions may also fit more than one of the seven classes above. Conversely there are also multiple functions which perform similar operations in some cases. This redundancy has resulted primarily from the evolution of SAMPLE-3D as a merger of K.K.H. Toh's development simulator and the initial version of the facet-motion algorithm.

#### 8.2.4. Data Structure Organization

Four major data structures are required to support the operations and algorithms in SAMPLE-3D: 1) A surface mesh consists of nodes, edges, and triangular faces. Each element contains connectivity information about neighbor elements (as described in chapter 4). Each node and triangle also stores rate and direction information. Multiple surface layers are also supported. 2) A rate array and material layer arrays store information about layer thicknesses, layer etch rates, and inhomogeneous material properties (including inhomogeneous etch rates). 3) An additional cell array provides an additional topography representation. Its use is described in chapter 5. 4) The visibility hemisphere above the wafer is given as an array of  $N_p \times M_\theta$  patches. Each patch contains its location, its area, fluxes at that point, and additional flags to assist in efficient visibility and flux integral calculations.

There are also several global variables and flags which are used throughout the program. They are classified as follows: 1) The simulation region is specified by minimum and maximum x, y, and z coordinates, cell sizes, and maximum array dimensions. 2) Model information includes the etch or deposition model type, etch and deposition source orientations, and the surface advancement mode. 3) Simulation control options consist largely of several logical flags controlling whether deloop routines are called, what type of mesh control routines are used, etc. 4) User and debug information consists largely of counters and other variables to keep track of the current time step, the size of the surface mesh, the names of external files, etc.

The material layers, cell array, and hemisphere array dimensions are set when SAMPLE-3D is compiled. The rate array is dynamically allocated according to the information contained in an external rate file. The surface mesh is also dynamically allocated, although the element sizes are set when SAMPLE-3D is compiled. The global variables and flags are set to default values when SAMPLE-3D is executed. User commands (described in the next section on user interfaces) update the various variables for specific simulation cases.

#### **8.2.5. Implementing Specific Models**

To connect a physical process model to the facet-motion algorithm requires: 1) the determination of the rate at each node, 2) the determination of the direction at each facet, 3) the specification of the relationship between the node rate and the facet direction, and 4) the specification of the level of generality which the facet-motion algorithm must handle. The possible levels of generality are: simple (S), isotropic (I), cosine-directional (D), or general (G). The general case can also handle the result when different vector components are summed together. Nodes at the simulation boundary, or at the edge of a mask are handled as

special cases.

The rate and direction at each node are determined by the location in the simulation region (*i. e.* the material), the visibility and distribution of incident flux, and surface phenomenon (*e. g.* surface migration or surface diffusion).

To handle the different requirements of different models it is convenient to use a large case statement in the function which sets the node rates. Each deposition or etch model is a separate case. The most general model requires a series of C functions proceeding according to the following outline:

```
case MODEL:
    initialize_or_reset_flags();
    set_motion_type(MODEL);
        set_shadows();
    set_visibility();

    for(pt=node_listhead[surface];pt!=NULL;pt=pt->next){
        get_material_information(pt); /*rate from location*/
        integrate_fluxes_over_visibility(pt);
    }

    for(pt=node_listhead[surface];pt!=NULL;pt=pt->next){
        do_surface_diffusion_or_migration(pt);
        do_reflection(pt);
    }

    for(tr=tria_listhead[surface];tr!=NULL;tr=tr->next){
        set_facet_direction(tr);
    }

    find_maximum_rate(&max_rate);
    break;
```

The various functions set variables contained within the surface data structure, or certain global variables such as the surface advancement type. For specific models, eliminating some of the above lines, or altering their order may result in improved computation times. Also,

within some of the above functions, time saving tests may be made. For example, if the etch rate at a particular layer is zero, no integrations are necessary for that point. The following table indicates which operations must be included for several different process models. The advancement modes are as before: simple (S), isotropic (I), directional including isotropic component (D), and general (G). A different number of fluxes is possible, depending on the number of particle types of interest (ions, neutrals or both).

<b>Table 8.1: Required Operations for Various Models</b>					
<b>Model</b>	<b>Advance Mode</b>	<b>Fluxes</b>	<b>Shadow</b>	<b>Solid Angle Visibility</b>	<b>Gaussian Convolution</b>
isotropic etch	I				
directional etch	D		X		
ion milling	G	1	X	(X)	
plasma etch	S	1	X	X	
RSE/RIE	G	2	X	X	
isotropic depo	I				
evaporation	S		X		
sputter depo.	S	1	X	X	
surface migration	S	1	X	X	X

Turning off certain operations can lead to considerable improvement in CPU time. For example, full visibility calculations are generally hundreds of times longer than shadow tests. Thus, if an ion beam closely resembles a delta-function distribution, a simple shadow test is all that is needed. In the current implementation, operations are turned off based on the type of model, or in some cases, based on user commands.

It is possible that certain models might introduce new modes of surface advancement. In that case, it is necessary to alter the C function which reconstructs the surface mesh from the advanced facets, or bypass it completely in favor of a new function. Suggestions for future SAMPLE-3D functional organization is given in section 8.5.

### 8.2.6. Estimating CPU Requirements

A theoretical estimate for the order of the CPU time required for a given problem is easily derived from the analysis of previous chapters. In this section,  $N$  is the number of surface nodes,  $\bar{N}_{3\sigma}$  is the average number of nodes within a  $3\sigma$  distance of any node,  $T$  is the number of surface triangles,  $\bar{T}$  is the average number of triangles attached to any node,  $H_\phi$  and  $H_\theta$  are the number of patches for each  $\Delta\phi$  and  $\Delta\theta$  strip in the hemisphere above the wafer,  $K$  is the total number of cells in the surface-cell hybrid cell array, and  $K_T$  is the worst case number of cells in the vicinity of one triangle.

The total time for one time step is the sum of the time to apply the facet motion algorithm, to set the surface-cell hybrid, to set shadows, to set visibility, to perform flux integrations, and to perform reflection, diffusion, or convolution operations. In chapter 4, the time to set the facet-motion algorithm was  $O(N)$  for the simple case,  $O(N(\bar{T}^2 + \bar{T}))$  for the isotropic and directional cases, and  $O(N(\bar{T}^2 + \bar{T} + \log\bar{T}))$  for the general case. From chapter 5, the time to update the surface-cell hybrid cell array is  $O(T \cdot K_T)$  and is performed every several time steps. Test cases in chapter 5 showed that up to 75% of the time to advance the surface is spent updating the cells. Once the cells are in place, all the shadowed nodes can be found in worst case  $O(N \cdot \sqrt{3}K^{1/3})$  time, and the visibility at each node can be found in  $O(N \cdot H_\theta \sqrt{3}K^{1/3})$  time, using the method that updates the shadow border along each  $\Delta\theta$  strip. Each integration for all the surface nodes requires  $O(N \cdot H_\phi \cdot H_\theta)$  time. Full reflection calculations require  $O(N^2)$  time. From chapter 6 and 7, a finite element surface diffusion calculation requires approximately  $O(N \cdot \bar{T}^2)$  time, and a convolution calculation requires approximately  $O(N \cdot \bar{N}_{3\sigma})$  time.

The total run time for a simulation is the sum of the CPU time for each step over all the time steps. The time step is in turn related to the minimum segment length. For a constant

maximum rate, doubling the number of nodes per micron, will double the number of time steps required, in addition to quadrupling the number of surface nodes. Thus a problem running in  $O(N)$  time with 10 nodes per micron and requiring 1 second, will require  $(4 \times 2) = 8$  seconds for 20 nodes per micron. A problem with four times as much area, but the same number of nodes per micron will require only 4 seconds.

The exact CPU time is hardware dependent and includes the mesh control time (including deloop). With the above analysis however, it is clear that reflection and convolution calculations are likely to dominate large problems. Full visibility and flux integrations are the next most dominant operations, although these may be controlled by choosing the density of patches in the hemisphere above the surface. The cell update and shadow tests are less dominant, and the facet-motion algorithm is the least time consuming step. As an approximate bench mark for further estimates, the following run times are typical for a  $1.0 \times 1.0 \mu\text{m}^2$  problem with 20 nodes/micron initially on an IBM Powerstation 530. Photoresist development over 300 time steps requires about 5 minutes. Plasma etching with a distributed flux and  $45 \times 90$  patches, requires about 60 minutes to reach a depth of  $1 \mu\text{m}$  for a  $0.5 \mu\text{m}$  diameter contact hole. For  $45 \times 10$  patches this is reduced to less than 8 minutes. Including full reflection results in over 300 minutes CPU time.

### 8.2.7. Estimating Memory Requirements

The memory required for a given simulation is related to the problem size, the density of surface information, and the density of cell information. On typical workstations such as a DECstation 3100, or IBM Powerstation 530, SAMPLE-3D has the following memory requirements:



Node element :  $292 + (4 \times 90) = 652$  bytes  
 Edge element : 44 bytes  
 Triangle : 268 bytes  
 Segment at node : 12 bytes  
 Triangle at node: 12 bytes  
 Hemisphere Patch: 100 bytes  
 Cell Element : 16 bytes  
 Rate Element : 8 bytes

An initial surface for a  $1.0 \times 1.0 \mu\text{m}^2$  region with 20 nodes per micron has  $20 \times 20 + (20-1) \times (20-1) = 761$  initial nodes, 2204 initial edges, and 1444 initial triangles. Each node is attached to from 2 to 6 edges and from 1 to 6 triangles. The total surface requires about 1.073MBytes of memory. A hemisphere array with  $45 \times 90$  patches requires 405Kbytes. A  $100 \times 100 \times 100$  cell array requires 8 Mbytes, and a similar size rate array requires 2 Mbytes. The executable code itself is roughly 500Kbytes to 1Mbytes depending on the compiler and the optimization. Thus the total memory needed is on the order of 12 Mbytes for this case. Increasing the mesh density or initial surface region will also increase the number of cells required for accurate simulation. For example, doubling the surface area requires twice as many cells. Thus the cell array requirements ultimately limit the size of the problem. Reducing the number of hemisphere patches reduces the node size, thus using  $45 \times 10$  patches results in significant memory savings over  $45 \times 90$  patches.

### 8.3. USER INTERFACE

The purpose of a user interface is to allow people to interact with a computer program. This concept is so obvious that it is often forgotten. Indeed the formal study of user interfaces and human computer interaction is a relatively new endeavor.<sup>2,3</sup> Before human factors research could begin, computers first had to work. The development of SAMPLE-3D follows a similar evolution in that most of the effort was concentrated on developing topography

simulation models and algorithms, and less attention was paid to user interface issues. However, it was still necessary to develop an input command language and take advantage of 3D plotting facilities to develop and test the topography simulation software. Once the basic user interaction capability was in place it was possible to develop a graphical user interface building on existing software libraries. This section discusses the SAMPLE-3D command interpreter, plotting capabilities, and the design and development of a basic graphical user interface taking the perspective of both the software developer and the user.

### **8.3.1. A Command Interpreter**

SAMPLE-3D supports nearly fifty commands. The command parser is fairly limited, following a strict syntax. The commands take the form of a trial statement followed by a list of arguments terminated by a semicolon. SAMPLE-3D also accepts some command line arguments, in the manner of typical unix programs, to allow or suppress certain features, or read certain files.

Command languages are among the least attractive forms of user interfaces from a user's point of view since they are hard to learn, prone to error, and time-consuming to use.<sup>4</sup> Nevertheless, they are among the easiest to develop and maintain, and simple modifications to a list of input commands are easy to make.

Implementing a new command in SAMPLE-3D requires nothing more than adding a keyword to the list of recognized commands in the command parser, and specifying the operation to be performed when that command is reached. The operation can be something as simple as setting a global variable or flag, or it may call a new function. The command interpreter executes instructions in the order they are entered.

### 8.3.2. Using the Command Interpreter.

Care must be taken to enter the necessary information in order, *i.e.*, initialization commands must be entered before execution commands. Most initialization commands may be entered in any order, except the etch model or deposition model command must be entered before the rate information is specified. The advancement method must also be set before the etch or deposition time. It is possible to reset most initialization commands after a simulation has been performed with the exception of the mesh density and simulation region parameters. The data structures, including the surface mesh, are not fully initialized until the first run command is executed. Thus, a plot command will not produce anything until after the first simulation is performed. Some limited error checking is implemented, for example if any argument is out of bounds, the entire line is ignored. Entering the wrong number of arguments or entering commands out of order will yield unpredictable results.

An example of an input command file and the result are shown in Fig. 8.4. The first line loads a binary rate matrix called "rval.oh". The next line causes the surface-cell hybrid to be activated. Line 3 to 6 specify no clipping of the mesh, set the advancement method to ray-trace, set isotropic etching, and set the mesh density to 7 nodes per micron. Line 7 sets the run time to 20 seconds, with 0.1 second time steps and line 8 runs the simulation. The next two lines turn off the no clipping flag, and initialize a new layer directly on top of the old one. This new layer becomes the active layer. Line 11 specifies a deposition model for unidirectional evaporation, an etch rate of  $-0.002\mu\text{m}$  per second, normal incidence, and no column effects. The advancement method is set to the facet-motion algorithm with an adaptive time step. The deposition time is 100 seconds with the time step set so no node travels more than 20% of the minimum segment length. Line 14 executes the deposition simulation. Line 15 writes the surface data structure to the file "depo.plot" and invokes the PDRAW program to display the

```

loadrate 0 rval.oh;
meshaccur 0 0 0 2;
noclip 1;
adv_method 1;
etchmodel 0;
meshdensity 7;
etchtime 20.0 0.1;
run;
noclip 0;
setnewlayer;
depomodel 1 -0.002 0.0 0.0 0;
adv_method 3;
depotime 100.0 0.2;
run;
plot3D depo.plot;
quit;

```

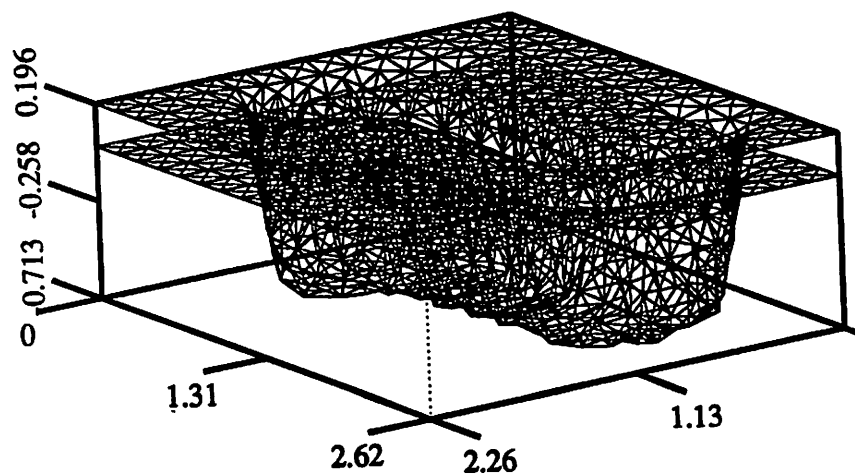


Fig. 8.4: SAMPLE-3D command input file and result.

result. Line 16 ends the simulation. A complete listing of SAMPLE-3D trial statements and required arguments is given in the appendix.

### 8.3.3. Graphics and Visualization

Throughout this dissertation, many 2D and 3D plots have been presented. These were created by extracting information from the SAMPLE-3D data structure, and writing into a file format used by an external plotter. Most of the 2D and 3D plots were generated with the programs PDRAW and DRAWPLOT by K.K.H. Toh.<sup>1</sup> Some extensions to the 3D drawing program were made to allow additional visualization capability.

PDRAW already had a hidden-line sort algorithm to draw triangles from back to front, according to the distance from the viewpoint to the triangle center. The run time was improved by implementing a better sort algorithm, capable of sorting thousands of triangles in less than a few minutes. The hidden-line algorithm does not properly handle many cases of overlapping triangles, but is sufficient for most surfaces generated by SAMPLE-3D.

A simple illumination algorithm was added which shades triangles according to their orientation with respect to a light source. The shading is proportional to the cosine of the angle between the surface normal and the light ray. This simple shading reveals some interesting features. As seen in Fig. 8.5, the feature with shading reveals a corrugated edge, which is not apparent in the plot without shading.

Another addition was the capability to add a grey scale value to each polygon in the PDRAW input file. Shading was used to generate figures 5.14 and 5.15, showing shadows and solid angle visibility as calculated internally in SAMPLE-3D.

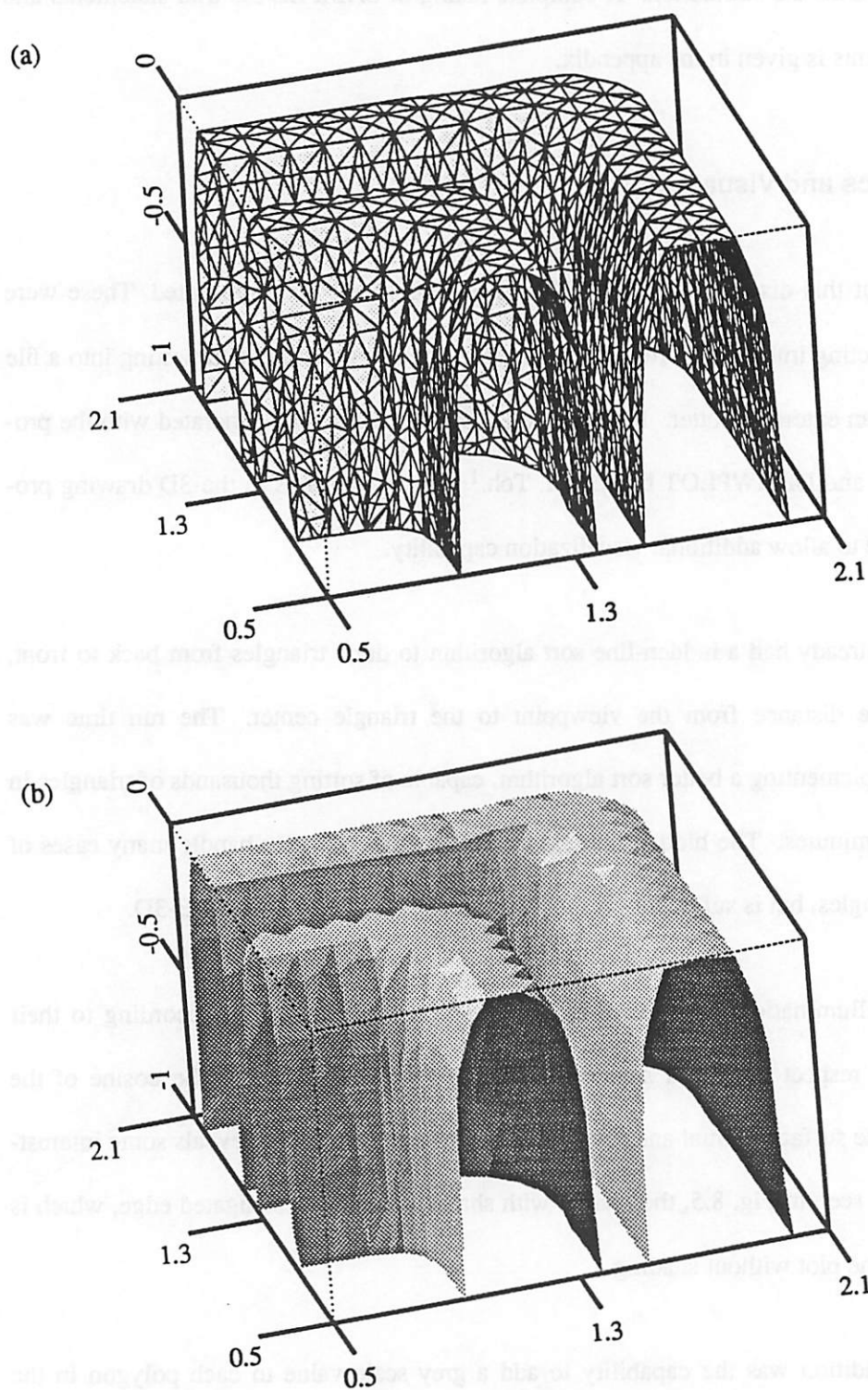


Fig. 8.5: 3D visualization for SAMPLE-3D. (a) Pdraw with hidden-line removal.  
 (b) Pdraw with illumination, revealing "corrugated" edge.

Finally, a file converter program was developed to translate PDRAW input files into a format readable by Mathematica<sup>TM</sup>. All of these modifications are listed in the appendixes.

#### 8.3.4. A Graphical User Interface

Previous work at UC-Berkeley in developing a graphical user interfaces for 2D topography simulation has demonstrated the power and applicability of such software.<sup>5, 6, 7</sup> SIMPL-DIX<sup>8</sup> in particular, offers several functions directly applicable to a user interface for 3D topography simulation. SIMPL-DIX reads a layout description in CIF format,<sup>9</sup> provides dialogue boxes and command menus, and is written using the X-Window System.<sup>10</sup> Extensions to SIMPL-DIX, by J.R. Camagna, allow regions of a mask to be selected and represented as constituent rectangles as used by SPLAT.<sup>11</sup>

A graphical user interface was developed using many of the functions already provided in SIMPL-DIX. The matrix transformation and 3D plotting capabilities of PDRAW were ported into the user interface. Finally, two modes of communication between the user interface and external programs were developed. In one mode of operation, the interface writes a command file and then makes a system call to the external program. This is similar to how SIMPL-DIX has been used previously to call other software.<sup>12</sup> The user interface also uses sockets to open a data path to an external program, in much the same way SIMPL-DIX communicates with SIMPL-2.<sup>13</sup> Sockets are the preferred mode of communication when an external data structure must be kept active, and is well suited to sending interpreter commands from the user interface to SAMPLE-3D.

It is more difficult to add menu options to the graphical user interface than to update the command interpreter in SAMPLE-3D since more code must be changed. The procedure for

adding new menu operations consists of 1) updating header files describing menu sizes and displays, 2) adding a function to perform the operation when a menu option is selected, and 3) ensuring that the new menu operation is placed in the appropriate location. The procedure is very similar to that described by H.C. Wu for SIMPL-DIX.

The basic menu structure and prompt-response operation of SIMPL-DIX is adequate for many simple queries, but is ineffective for entering large amounts of information. An alternate approach is to invoke a form fill when large amounts of data are needed. Fig. 8.6 is an example of a particularly complex form for entering photoresist exposure information. The form was created using the Athena Widget set available in the X11 libraries.<sup>14</sup> The form is invoked as a separate program. Model libraries may be loaded from external files, and the variable field may be edited. When the user is satisfied with the data entries, the ACCEPT button causes the form data to be written to a command file and sent to the appropriate simulation module. Forms are quite easy to build using widget sets, for example the form in Fig. 8.6 was coded and debugged in 4 hours.

### 8.3.5. Using the Graphical User Interface

The user interface prompts the user for information (when possible with forms), and translates the responses into the strict syntax understood by the command interpreters of SAMPLE-3D, SPLAT and BLEACH. To some extent the user interface limits the possible choices to valid operations.

The user interface is organized to allow access to lithography, etching and deposition simulation, as well as to external plotters and the HUNCH and WORST utilities of the original SIMPL-DIX program. Selecting the lithography button results in a new command menu



ExposeForm		Photoresist Exposure Form - E.W. Scheckler 10/22/91	
<div> <div>Select Photoresist</div> <div> <div>load model</div> <div>save model</div> </div> <div> <div>cancel</div> <div>ACCEPT</div> </div> </div>		<div> <div>Develop Model:</div> <div>K1nR3</div> </div>	
<div>Resist Name:</div> <div>H1PR65121</div>		<div>R1:</div> <div>0.0759</div>	
<div>lambda (um):</div> <div>0.365</div>		<div>R2:</div> <div>0.0001</div>	
<div>A (1/um):</div> <div>1.05</div>		<div>R3:</div> <div>7.46</div>	
<div>B (1/um):</div> <div>0.11</div>		<div>Diffusion (um):</div> <div>0.08</div>	
<div>C (cm<sup>2</sup>/mJ):</div> <div>0.012</div>			
<div>resist n:</div> <div>1.82</div>			
<div>resist k:</div> <div>-0.033</div>			
<div>thickness (um):</div> <div>1.0</div>		<div>Layers</div> <div>add layer</div> <div>remove layer</div>	
<div>substrate n:</div> <div>6.522</div>			
<div>substrate k:</div> <div>-2.705</div>			
<div>dose (mJ/cm<sup>2</sup>):</div> <div>180.0</div>			
<div>BLEACH File:</div> <div>Bleach.input</div>			

Fig. 8.6: A form for entering photoresist exposure model data.

which allows a user to call software to simulate an aerial image, calculate the resist exposure, load an external rate file, run photoresist development simulation or display the result. Similarly, selecting the etching or deposition buttons allows models to be initialized, simulation parameters to be entered, and simulations to be executed. The user is guided through the simulation session without having to know the SAMPLE-3D command interpreter language or syntax. However, some knowledge of the meaning of model parameters is assumed. Fig. 8.7 shows the user interface window with the highest level command menu, the mask layout and the end result of a simulation session.

For many operations, SAMPLE-3D is still too slow to be a fully interactive program. In those cases, it is better to generate input command files and run SAMPLE-3D as a batch job. SAMPLE-3D also does not yet fully behave like a piece of processing equipment. This fact is a result of the limits of the models, which are not yet sophisticated enough to translate temperatures, pressures, gas concentrations, and the like into etch rates. Some specification of intermediate model parameters is still required. In the future, it might be desirable to incorporate libraries relating physical information to process model parameters. One promising approach, which has been implemented in SIMPL-IPX by T. Luan, is to use manufacturing based models to generate input parameters for simulation from process recipe specifications.<sup>15</sup> A user interface by itself cannot address all the issues involved in coordinating process flow information, physical models, and the flow of information among different simulation programs. This task is properly the role of more sophisticated technology CAD integration systems.

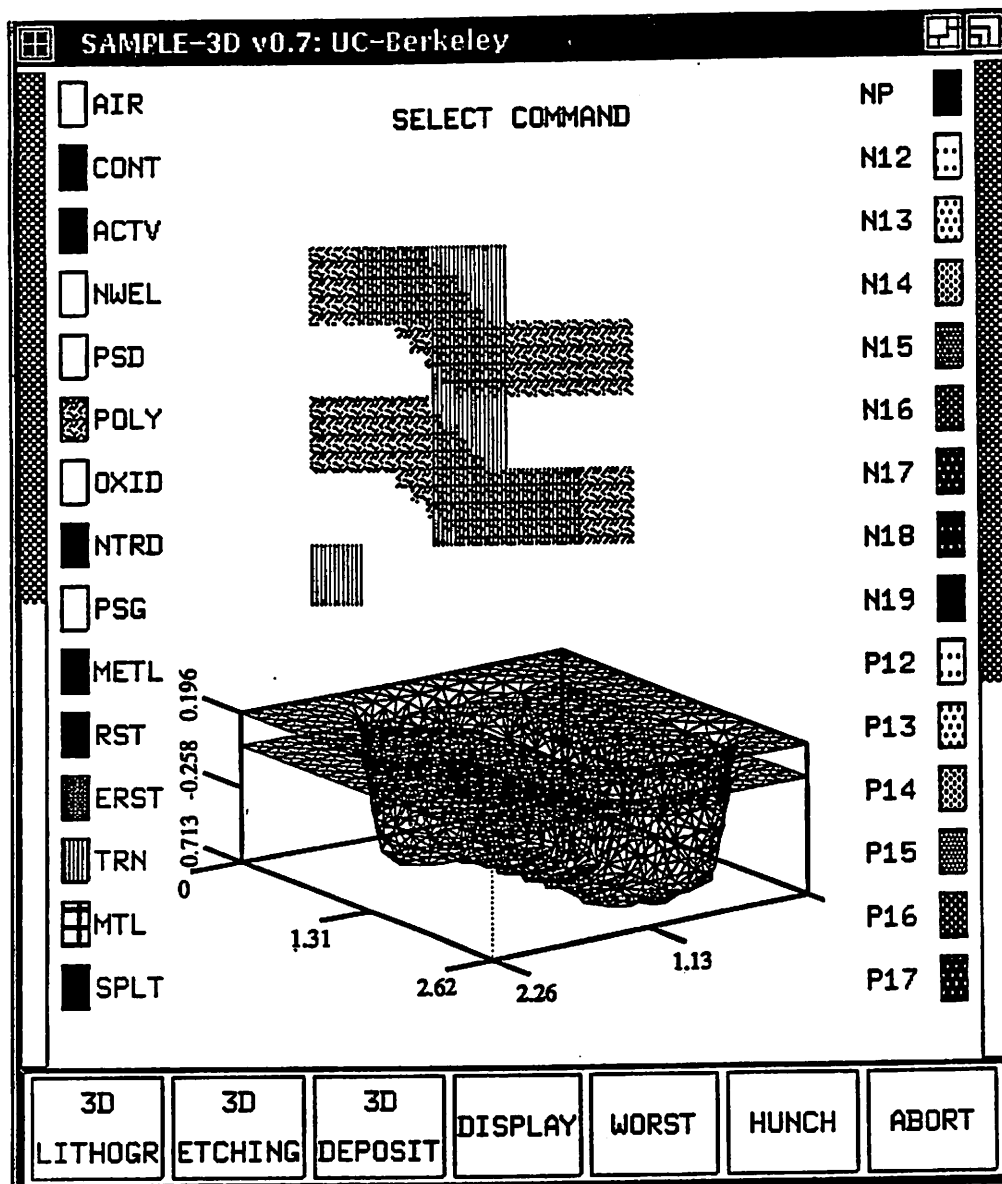


Fig. 8.7: X-Window graphical user interface for SAMPLE-3D at end of session.

#### 8.4. TECHNOLOGY CAD INTEGRATION

Initially, process simulation capability consisted of individual programs specific to particular applications. Some programs eventually encompassed multiple processes. Nevertheless many simulation users wished to integrate a wide range of capabilities into a single software framework, so that entire process flows could be simulated. Many solutions to this problem have been presented, including SIMPL-IPX<sup>11</sup> which integrates such disparate simulators as SAMPLE, SUPREM-IV<sup>16</sup> and CREEP.<sup>17</sup> Other work has proceeded in both Europe and Japan.<sup>18, 19, 20, 21, 22</sup> Many of the reported systems either integrate different simulators by translating information among the programs, or put all of the modeling capability together in one large monolithic structure. While this has been useful in achieving many of the goals of TCAD integration, it has resulted in systems which are difficult to develop and maintain. Standardized profile interchange formats<sup>23</sup> address some of the problems, but also introduce others such as the issue of maintaining compatibility as formats evolve. These problems are only increased when 3D simulation is considered.

An improved approach is to introduce toolkits of functions which operate on data objects. SIMPL-IPX attempted this using the 2D data structure provide in SIMPL-2.<sup>24</sup> This approach has been further developed into a library of utilities well suited to integrating 2D topography and impurity simulators.<sup>25</sup> Another advance has been to further develop and extend this approach to generalized data objects based on the PIF wafer representation, including work at the University of California at Berkeley,<sup>26</sup> Massachusetts Institute of Technology,<sup>27</sup> and the Technical University of Vienna.<sup>28</sup> The most advance stage on this evolutionary path (and at the moment the most theoretical) is to envision a framework based on a client/server model.<sup>29</sup> The wafer state is maintained as an active data object by a server. The server provides functions for maintaining and altering the wafer state. Simulation tools are

clients, developed using the server functions, which communicate with the server to alter the wafer state. External tools could also be incorporated in the system by providing a "wrapper" function to handle communication between the external tool and the server. A prototype system using this approach, and integrating new and existing 2D tools from several industry and university groups on two continents has already been demonstrated at Stanford University on August 9, 1991.

In a broad sense, SAMPLE-3D is a rudimentary geometry server, with physical models acting as clients which describe the surface deformation. The difference is that SAMPLE-3D is not completely general, nor are the server functions completely defined as a library of operations. Furthermore, the delineation between server and client is not as "clean" as might be expected for a robust framework. The models developed in chapters 6 and 7 may be applied to topography simulation with other geometry servers, just as other models may be incorporated in SAMPLE-3D.

The experience of developing SAMPLE-3D does raise an important issue for a full 3D server. It is by no means clear that the best algorithms for performing 3D topography evolution are known in general. Indeed the great variety of 3D simulation algorithms (cell based, diffusion equation based, network model, ray-trace, facet-motion) point to the complexity of issues involved. There are many variations on surface advancement algorithms which are best applied in certain cases. For example, a ray-trace algorithm with an efficient deloop capability is well suited to isotropic etching in photoresist development, but a more general case like ion milling requires the capability provided by the facet-motion algorithm. The ability to activate and deactivate certain aspects of the surface deformation algorithm, based on the process physics, is important to achieve the best accuracy and efficiency for practical simulation. Developers must also have access to primitive server functions to create new geometric

operations should they be necessary. It may also be necessary to add new data elements to the server data structure in order to improve certain operations. SAMPLE-3D has evolved along both the algorithmic and modeling fronts. There is every reason to expect that the future evolution of SAMPLE-3D will continue along both fronts. General client/server frameworks should expect to do the same. Failure to fully consider the complexity of 3D geometric algorithms, or to provide an open architecture for improving the server, will seriously limit the progress which the TCAD SWR initiative is well positioned to achieve.

## 8.5. ORGANIZING SAMPLE-3D AS A FUNCTIONAL LIBRARY

SAMPLE-3D is already organized as a hierarchy of modules which perform operations on data structures. The current implementation however lacks rigorous adherence to the rules of strict modularity. This is a direct result of the evolutionary path towards the development of SAMPLE-3D. This section is an attempt to provide an outline architecture for the reorganization and continued development of SAMPLE-3D as a library of functions working on data objects. The emphasis here is on the facet-motion algorithm acting on a single surface mesh, with code written in the C language. Future work may employ an object oriented language like C++ and use more complex data representations, but the simpler approach provides a clearer connection with the rest of this work.

The general high level simulation function has the form:

```
do_simulation(surface,cells,model,flags,max_time)
struct SurfaceMesh surface;
struct CellArray cells;
struct Model model;
struct SimulationFlags flags;
double max_time;
```

```

{
    double dt,time=0.0;

    initialize_geometry(surface,cells);
    initialize_model_parameters(model);
    while(time <= max_time){
        set_rate_vectors(surface,cells,model,flags,&dt);
        advance_mesh_and_modify(surface,cells,model,flags,dt);
        time += dt;
    }
}

```

After initialization, the rate vectors are set and the time step is determined from the maximum rate and minimum allowed segment length. The mesh advances using the facet-motion (or any other) algorithm. Finally, the mesh is modified by merging and splitting segments and eliminating loops, and the cells are updated. The time counter increments before proceeding.

All the necessary information is contained in the two geometry data structures "surface", and "cells". All the parameters for the physical models are contained in the data structure "model". The "flags" structure contains any other variables which control the simulation. Each function has an argument list containing data structures which are used as input and may also be altered, data structures which are used only as input, and variables which contain data returned by the function.

As described in previous sections, the models are connected to the simulation by specifying the rate and direction at every facet node. The specific operations needed to set the rates depend on the specific model and are built out of lower level functions. The visibility operations are:

```

set_shadow_by_source_direction(surface,
    cells, flags, phi, theta)
set_shadow_point_source(surface,
    cells, flags, location, extent)
set_visibility(surface, cells, flags)

```

Before the rates can be set, the material and related information at each surface node must be set:

```

get_material_type(nd, model, flags, &material)
get_sputter_yield(material, alpha, model, flags, &yield)
get_crystal_rate(material, alpha, model, flags, &rate)
get_inhomogeneous_rate(nd, model, flags, &rate)
set_rates(surface, model, flags)

```

Once the visibility and rates are known, integrations may be performed:

```

integrate_flux_at_node(nd, flux_function,
    flags, &magnitude, &direction)
integrate_flux(surface, flux_function, flags)

```

It is sometimes necessary to include direction dependent rate information inside the flux integrations. In that case, the rate functions must be used inside the flux integration functions. Also, other operations such as reflection and convolution may require the initial rates to be set prior to invocation.

```

integrate_flux_with_yield(surface, flux_function, model, flags)
reset_rates_given_reflection(surface, refl_func, model, flags)
convolve_rates(surface, model, flags);

```

Once the rate vectors are completely specified, the mesh advancement proceeds:



```

advance_mesh_and_modify(surface, cells, model, flags, dt);
struct SurfaceMesh surface;
struct CellArray cells;
struct Model model;
struct SimulationFlags flags;
double dt;

{
    int mode;

    find_mode_from_model(model, flags, &mode);
    move_facets(surface, dt, mode, flags);
    reset_nodes(surface, mode, flags);
    update_cells(surface, cells, flags);
    remove_loops(surface, cells, flags);
    modify_mesh(surface, flags);
}

```

The mode simply gives a flag telling whether the surface motion is simple, isotropic, directional, or general. The flags data structure contains variables controlling whether deloop will be performed and how the mesh is to be modified, etc. The facet motion algorithm is contained in the first three functions above, and the mesh control, surface-cell hybrid control and deloop operations are contained in the last three functions.

Each of the above functions is made up of even more basic operations. Also, each function must check the various relevant data structures and flags to ensure that the proper operation is performed. This point is also where time saving checks are made, for example querying the rate before integrating over the visibility. Any new function must check that the input data is valid, and that it updates the correct output data. This operation may involve fairly comprehensive internal parameter evaluation. Additionally, several variations on a single operation may be necessary for optimum performance, depending on the status of existing input data. It is precisely because of this complex data manipulation, that an implementation using C++ should be investigated for the next phase of SAMPLE-3D.

## 8.6. SUMMARY AND CONCLUSION

SAMPLE-3D provides a platform for easily including new models and new algorithms to support multiple applications. The current implementation may be viewed as a hierarchy of operations for connecting process models to the evolution of a surface topography. From a theoretical point of view, SAMPLE-3D mirrors the ambitions of the CAD Framework Initiative TCAD Semiconductor Wafer Representation. SAMPLE-3D is smaller in scope and ambition, but does include a basic data representation, functions for performing deformation, visibility, and surface diffusion operations. The organization allows new models to be included, building on the existing framework. SAMPLE-3D also has a basic user interface capability, including an X-Windows based graphical user interface linking CIF layout files, and image and exposure simulation with 3D etching and deposition. The user interface can also be expanded as the program evolves.

As it stands, changes to the models in SAMPLE-3D require modifications to the software and recompilation. It is not possible to build new models by sitting at the user interface and putting equations together. Although such a system would be possible, current workstation capability does not yet make such a goal practical, since the run times for such a general system would be quite slow. For the moment, it is better to code the models directly, and compile a 3D simulator optimized for those models. SAMPLE-3D as currently implemented is applicable to a variety of problems in integrated circuit fabrication, some of which are demonstrated in the next chapter.

## REFERENCES

1. K.K.H. Toh, "Algorithms for Three-Dimensional Simulation of Photoresist Development," Memo. No. UCB/ERL M90/123, Ph.D. Dissertation, University of California, Berkeley, December 14, 1990.
2. R. Rubenstein and H. Hersh, *The Human Factor - Designing Computer Systems for People*, Digital Press, Burlington, MA, 1984.
3. B. Schneiderman, *Designing the User Interface: Strategies for Effective Human-Computer Interaction*, Addison-Wesley, Reading, Mass., 1986.
4. J.D. Foley, A. van Dam, S.K. Feiner, and J.F. Hughes, *Computer Graphics: Principles and Practice*, p. 403, Addison-Wesley, Reading, Mass., 2nd edition, 1990.
5. H.C. Wu, A.S. Wong, Y.L. Koh, E.W. Scheckler, and A.R. Neureuther, "Simulated Profiles from the Layout, Design Interface in X (SIMPL-DIX)," *IEDM Technical Digest*, San Francisco, CA, December 1988.
6. A.S. Wong, "An Integrated Graphical Environment for Operating IC Process Simulators," ERL Memorandum M89/67, University of California, May 1989.
7. S.M. Chiu, *PIX: Program Interface in X, A Graphical User Interface Software User's Guide*, Electronics Research Laboratory, University of California, March 1991.
8. H.C. Wu, "SIMPL-DIX (Simulated Profiles from the Layout - Design Interface in X)," ERL Memorandum UCB/ERL M88/13, University of California, January 1988.
9. C.A. Mead and L.A. Conway, *Introduction to VLSI Systems*, Addison-Wesley, Reading, Mass., 1980.
10. R.W. Scheifler and J. Gettys, "The X Window System," *ACM Transactions on Graphics*, vol. 5, no. 2, pp. 79-109, April 1986.

11. E.W. Scheckler, A.S. Wong, R.H. Wang, G. Chin, J.R. Camagna, K.K.H. Toh, K.H. Tadros, R.A. Ferguson, A.R. Neureuther, and R.W. Dutton, "A Utility-Based Integrated Process Simulation System," *1990 Symposium on VLSI Technology, Digest of Technical Papers*, pp. 97-98, Honolulu, Hawaii, June 4-7, 1990.
12. E.W. Scheckler, "Extraction of Topography Dependent Electrical Characteristics from Process Simulation using SIMPL, with Application to Planarization and Dense Interconnect Technologies," ERL Memorandum UCB/ERL M89/27, University of California, March 1989.
13. S. Sechrest, *An Introductory 4.3BSD Interprocess Communication Tutorial*, Computer Science Research Group, EECS Department, University of California, July 1985.
14. T. O'Reilly, *X Toolkit Intrinsics Reference Manual*, 5, O'Reilly & Associates, Inc., Sebastopol, CA, 1990.
15. T.L. Luan, "Manufacturing-Based IC Process and Device Simulation," ERL Memorandum UCB/ERL M91/55, University of California, May 1991.
16. M.E. Law, C.S. Rafferty, and R.W. Dutton, *SUPREM-IV User's Manual*, Stanford University, 1988.
17. P. Sutardja and W.G. Oldham, "Modeling of Stress-Effects in Silicon Oxidation Including Viscosity Oxide," *IEDM Technical Digest*, pp. 264-267, December 1987.
18. C.H. Corbex, A.F. Gerodolle, S.P. Martin, and A.R. Poncet, "Data Structuring for Process and Device Simulations," *IEEE Transactions on Computer-Aided Design*, vol. 7, no. 4, pp. 489-500, April 1988.
19. K. Kato, N. Shigyo, T. Wada, S. Onga, M. Konaka, and K. Taniguchi, "A Supervised Simulation System for Process and Device Designs Based on a Geometrical Data interface," *IEEE Transactions on Electron Devices*, vol. ED-34, no. 10, pp. 2049-2058,

October 1987.

20. E. van Schie, *Trendy: An Integrated Program for IC Process and Device Simulation*, Ph.D. Dissertation, Universiteit Twente, The Netherlands, 1990.
21. Y. Akiyama, Y. Hatanaka, M. Asou, Y. Tamegaya, H. Ikeuchi, and H. Kuge, "Development of a unified process and device simulation environment," *1991 International Workshop on VLSI Process and Device Modeling*, pp. 92-93, Oiso, Japan, May 1991.
22. P.A. Gough, M.K. Johnson, P. Walker, and H. Hermans, "An Integrated Device Design Environment for Semiconductors," *IEEE Transactions on Computer-Aided Design*, vol. CAD-10, no. 6, pp. 808-821, June 1991.
23. S.G. Duvall, "An Interchange Format for Process and Device Simulation," *IEEE Transactions on Computer-Aided Design*, vol. CAD-7, no. 7, pp. 741-754, July 1988.
24. E.W. Scheckler, A.S. Wong, R.H. Wang, G. Chin, J.R. Camagna, A.R. Neureuther, and R.W. Dutton, "A Utility-Based Integrated System for Process Simulation," *to appear IEEE Transactions on Computer-Aided Design*, May 1992.
25. R.H. Wang, "The Berkeley Topography Utilities," M.S. Project Report, University of California, August 1991.
26. A.S. Wong and A.R. Neureuther, "The Intertool Profile Interchange Format: A Technology CAD Environment Approach," *IEEE Transactions on Computer-Aided Design*, vol. CAD-10, no. 9, pp. 1157-1162, September 1991.
27. D.S. Boning, M.L. Heytens, and A.S. Wong, "The Intertool Profile Interchange Format: An Object-Oriented Approach," *IEEE Transactions on Computer-Aided Design*, vol. CAD-10, no. 9, pp. 1150-1156, September 1991.
28. S. Selberherr, F. Fasching, C. Fischer, S. Halama, H. Pimingstorfer, H. Read, H. Stippel, P. Verhas, and K. Wimmer, "The Viennese TCAD System," *1991 International*

*Workshop on VLSI Process and Device Modeling*, pp. 32-35, Oiso, Japan, May 1991.

29. A. Wong, W. Dietrich, and M. Karasick, "The SWR Architecture Document Version 0.2," CAD Framework Initiative #162, Austin, Texas, March 1991.

## CHAPTER 9

### APPLIED SIMULATION

#### 9.1. INTRODUCTION

General three-dimensional topography simulation capability is applicable to many interesting issues in advanced IC process technology. Successful simulation, however, requires adequately characterized process models. Such models do not exist in general. Nevertheless, plausible physical models may be proposed for individual cases. This chapter demonstrates how the models and algorithms of SAMPLE-3D may be applied to specific technology examples. It was not possible in this work to provide rigorous comparisons with experimental results; however, the examples should prove in agreement with similar published results familiar to IC process technologists.

A collection of simulation examples were chosen to demonstrate many of the unique features of SAMPLE-3D. A pattern transfer example describes how deep-UV lithography simulation is coupled with plasma etching. A photoresist silylation example demonstrates plasma etching of a material with an inhomogeneous directional etch rate. Ion milling of a rotating wafer shows how the complex interaction of a moving shadow and changing angles of incidence create features suitable for continuous phase transitions on advanced phase shift masks. Phase-shift lithography with standing waves demonstrates the use of the surface-cell hybrid based deloop function. Deposition followed by etching with density variation shows how inhomogeneous material properties may be considered in a multistep process simulation. A simulation involving wet etching, dry etching, and metal deposition demonstrates a multistep example which may be compared with a similar simulation presented by Fujinaga *et. al.*

using 3D-MULSS.<sup>1</sup> Finally, a simulation comparing surface advancement development algorithms with the cell-removal algorithm of chapter 3 is presented to address some of the pros and cons of the alternate approaches.

## 9.2. DEEP-UV LITHOGRAPHY AND PATTERN TRANSFER

Fig. 9.1 is the result of deep-UV lithography simulation of a chemical amplification photoresist followed by a dry etch pattern transfer step into a silicon substrate.

The aerial image simulation with SPLAT<sup>2</sup> assumed a light wavelength of  $\lambda=248\text{nm}$ , a numerical aperture  $\text{N.A.}=0.4$ , a partial coherence  $\sigma=0.5$ , and  $0.4\mu\text{m}$  clear opaque elbows on a clear field. The exposure simulation used a model for Shipley SNR-248 negative tone, chemical amplification photoresist. The Dill model bleaching parameters<sup>3</sup> in the simulation were  $A=-0.712$ ,  $B=1.157$ , and  $C=0.00229$ . The refractive index was  $1.79 - 0.0088i$  for the photoresist and  $1.70 - 3.38i$  for the silicon. The exposure dose was  $25.2 \text{ mJ/cm}^2$ . The post exposure bake simulation used an analytic model derived from 2D solutions of the reaction kinetics<sup>4</sup> and assumed a bake of  $140^\circ\text{C}$  for 60 seconds. The development model relating the etch rate to the exposure matrix  $M(x,y,z)$  was:

$$R(x,y,z)=R_0\left[1-\frac{C_e}{C_0}\right]^\alpha \quad (9.1)$$

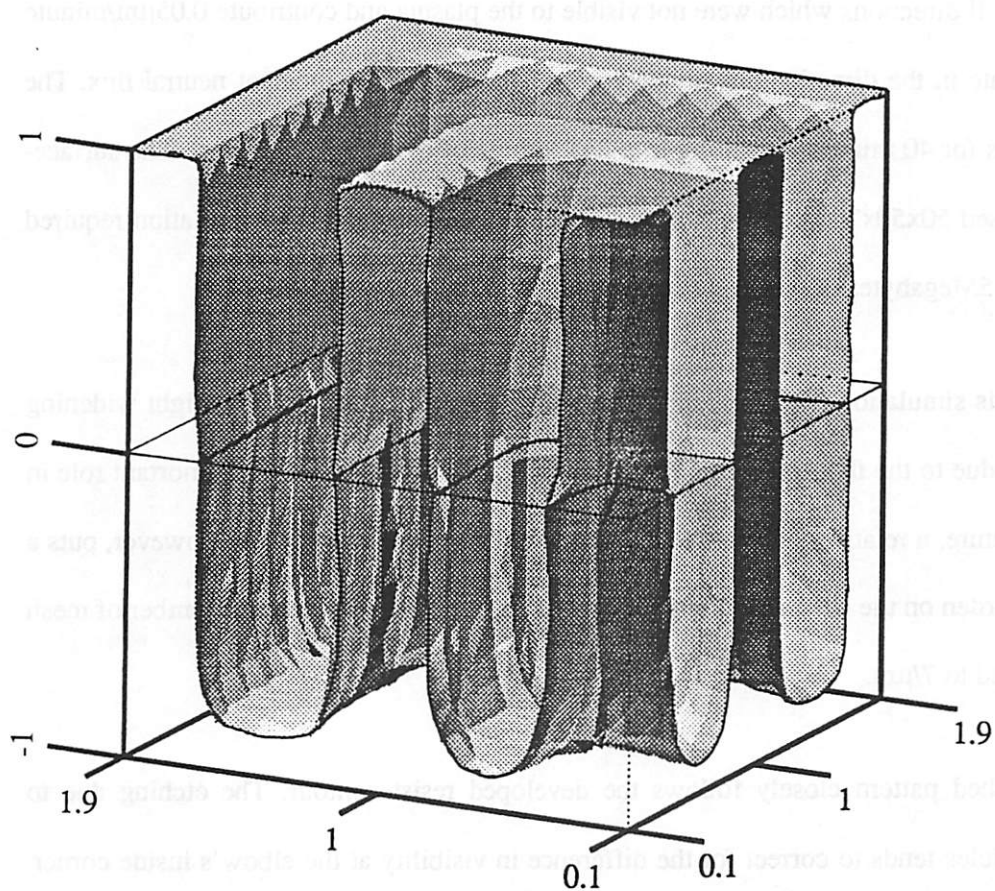
with  $R_0 = 6.5\mu\text{m/s}$ ,  $C_0 = 0.035$ ,  $\alpha = 6.3$ , and  $C_e$  given by:

$$C_e = M^6 - 6M^5 + 15M^4 - 20M^3 + 15M^2 \quad (9.2)$$

The etch rate is zero if  $C_e$  exceeds  $C_0$ . The development proceeded for 40 seconds.

After the development simulation, parameters were set to define the photoresist as a mask. The plasma etch model parameters do not describe an experimental process, but nevertheless approximate a possible etch mechanism. The photoresist was defined as a



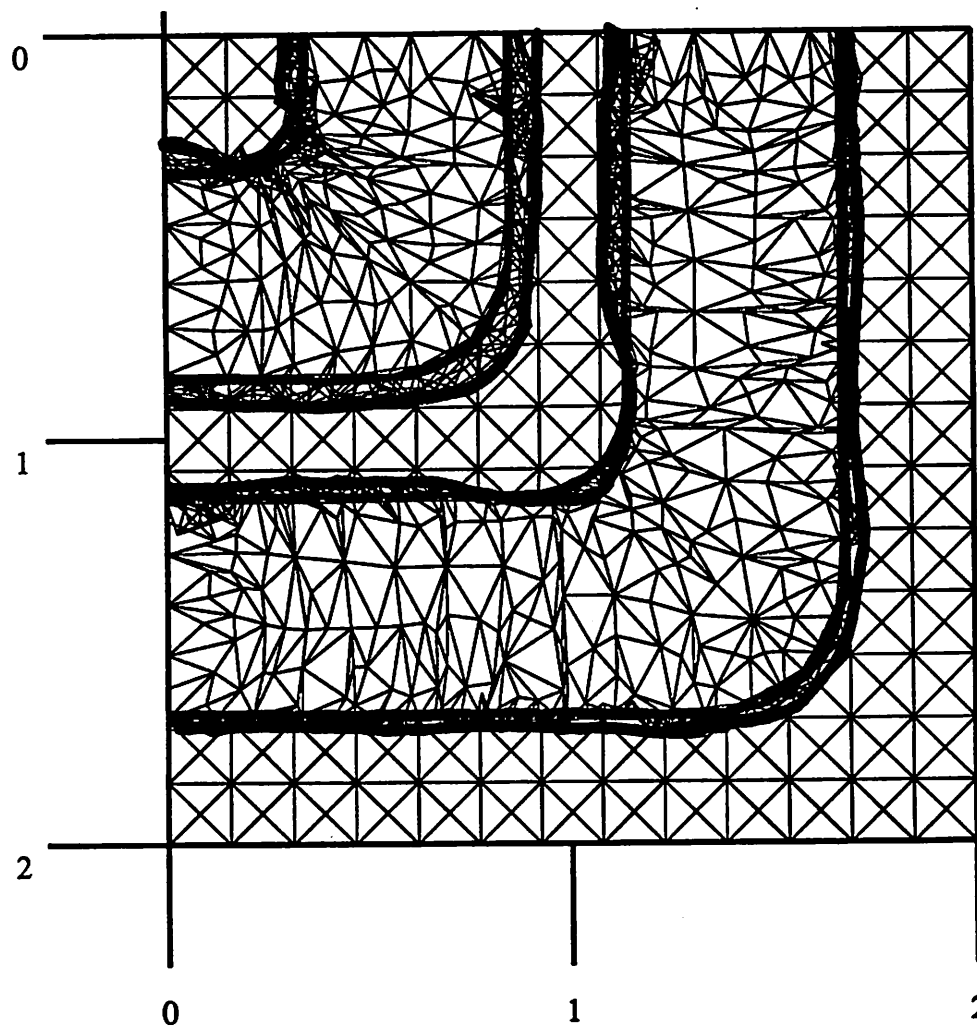


**Fig. 9.1:** Deep-UV lithography of 0.4μm elbows in SNR-248 resist followed by plasma-etching of underlying material.

material with zero etch rate for all plasma etch components (*i.e.* a perfect mask.) The silicon assumed a neutral flux distribution of  $F_{dn} = \cos\phi$ , and a delta function for directly incident ions. The etch rate on a flat wafer was defined as  $0.1\mu\text{m}/\text{minute}$  for the neutrals alone, and  $0.4\mu\text{m}/\text{minute}$  in the presence of ions. Reflected ions and neutrals were assumed to arrive uniformly from all directions which were not visible to the plasma and contribute  $0.05\mu\text{m}/\text{minute}$  to the etch rate in the direction already determined by the directly incident neutral flux. The etch time was for 40 minutes, resulting in a final profile of over 6000 triangles. The surface-cell hybrid used  $50 \times 50 \times 200$  cells. Using  $45 \times 90$  hemisphere elements, the simulation required 3 hours and 15Megabytes of physical RAM, on an IBM Powerstation 530.

With this simulation 3D issues in pattern transfer may be studied. The slight widening of the trench due to the flux distribution is apparent. Since visibility plays an important role in the final structure, a relatively dense  $45 \times 90$  patch hemisphere was used. This, however, puts a significant burden on the CPU speed requirements. To reduce this burden, the number of mesh nodes was held to  $7/\mu\text{m}$ .

The etched pattern closely follows the developed resist contour. The etching due to reflected particles tends to correct for the difference in visibility at the elbow's inside corner. A similar simulation, with the reflected etch rate component reduced to  $0.001\mu\text{m}/\text{min}$ , is shown from the bottom in Fig. 9.2. In this case, the etched pattern deviates from the developed resist pattern due to the change in visibility at the inside elbow. The above examples show that investigation of the relationship of the etched pattern to changes in the relative strengths of the various etch components requires full three-dimensional simulation for inherently three-dimensional structures.



**Fig. 9.2:** Deep-UV pattern transfer revealing 3D effects at inside elbow. Plot shows view from below the trenches. Original resist pattern, and widest part of trench are highlighted to better illustrate the effect.

### 9.3. SILYLATION

Silylation of photoresists is a surface-imaging dry development technology for fine patterning.<sup>5, 6, 7</sup> The process consists of three parts. First, the photoresist is exposed. Second, silicon is incorporated into the exposed regions. Third, the silylated photoresist is etched in an oxygen plasma. The etch rate is greatly influenced by the Si concentration.

Silylation of resists consisting of styrene and para (t-BOC) styrene copolymers, with tri-phenyl sulfonium hexa-fluoro-arsenate as a photosensitizer, and with hexa-methyl-disilazane (HMDS) as the silylating agent has been studied by Spence *et. al.*<sup>8</sup> The amount of silicon in the resist is directly related to the number of sites available for silicon uptake. In the tertiary-butyloxycarbonyl (t-BOC) resist, available sites are generated when a photo-produced acid catalyzes the removal of a t-BOC protecting group from the resin.<sup>9</sup>

Combining an exposure dependent Si uptake model with etch rate vs. Si content data yields a quantitative t-BOC silylation model. First, exposure in the resist is simulated using Dill's model.<sup>10</sup> The normalized concentration of chemical sites for Si uptake  $C_{as}$  are related to the  $M$  values in the resist and the post exposure bake, by solving the reaction kinetics and fitting to experimental data. Ferguson has derived the following expression to model this step:<sup>4</sup>

$$C_{as} = 1 - \exp \left[ -(1-M)^m \left( \frac{k_1}{mk_2} \right) (1 - e^{-mk_2 t}) \right] \quad (9.3)$$

where  $t$  is the bake time, and  $k_1$ , and  $k_2$  are rate coefficients for the deprotection and acid loss reactions in the photoresist. The rate coefficients are related to an activation energy  $E_{ai}$ , the bake temperature  $T$  and the Boltzman constant  $k_B$  by

$$k_i = k_{io} \exp \left( \frac{E_{ai}}{-k_B T} \right) \quad (9.4)$$

The parameter  $m$  indicates the order of the deprotection and acid-loss reactions. Experimental measurements were used to derive  $k_1=23.4s^{-1}$  and  $k_2=0.034s^{-1}$  for post exposure bake at  $90^\circ$  for 60 seconds. The  $m$  parameter was 1.00 indicating first order reaction kinetics.

Experimental evidence of the diffusion of Si in this photoresist system support the hypothesis that all available chemical sites become filled with Si. A proposed mechanism for the reaction is shown in Fig. 9.3.<sup>8</sup> This mechanism relates the weight percent Si to the concentration of available sites  $C_{as}$ :

$$W\%_{Si} = \frac{C_{as} W_{Si}}{(1-C_{as})W_{nonsilylated} + C_{as} W_{silylated}} \quad (9.5)$$

where  $W_{Si}=28a.u.$ ,  $W_{silylated}=192a.u.$ ,  $W_{nonsilylated}=120a.u.$  The directional etch rate in an oxygen plasma is linearly related to the weight percent Si for large Si concentrations.<sup>11</sup> However, for Si concentrations less than a few weight percent, the etch rate increases rapidly.<sup>12</sup> A simple two-piece linear model can be used to fit experimental data for the two regions. Fig. 9.4 shows the etch rate in the vertical direction vs. weight percent using this approach. The simple model assumes etching in the vertical direction only.

A simulation for  $0.4\mu m$  elbows is shown in Fig 9.5. The simulation assumed deep-UV optical lithography at  $\lambda=248nm$ ,  $N.A.=0.42$ , and  $\sigma=0.5$ . The exposure used Dill's model<sup>10</sup> and vertical propagation with  $A=-0.24$ ,  $B=3.77$ , and  $C=0.0011$ . The relatively high B value indicates that the resist absorbs most of the incident energy, giving a surface-imaging process. The exposure dose was  $3.5mJ/cm^2$ . Experimental data for top loss in dry development in an RIE system at 100 watts, 50 mTorr yielded  $p1=10\%$ ,  $p2=3\%$ ,  $r1=0.7$  and  $r2=4.2$ .<sup>5</sup> The etch rates are in normalized units. The conversion to rates in micrometers per second depends on the nominal etch rate for unsilylated resist.

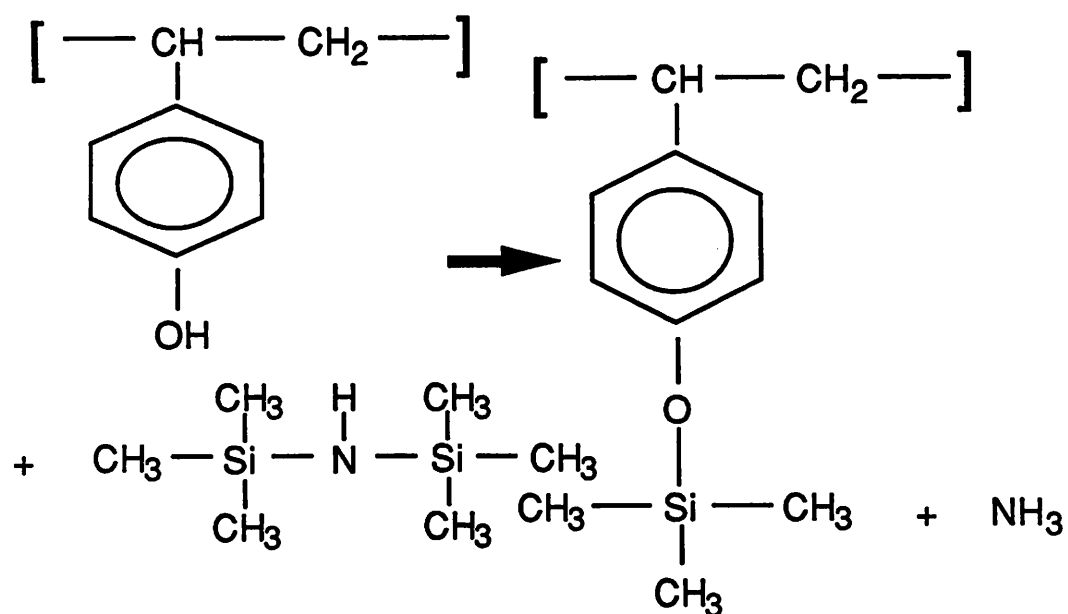


Fig 9.3: Possible silylation reaction according to C.A. Spence (ref. 8)

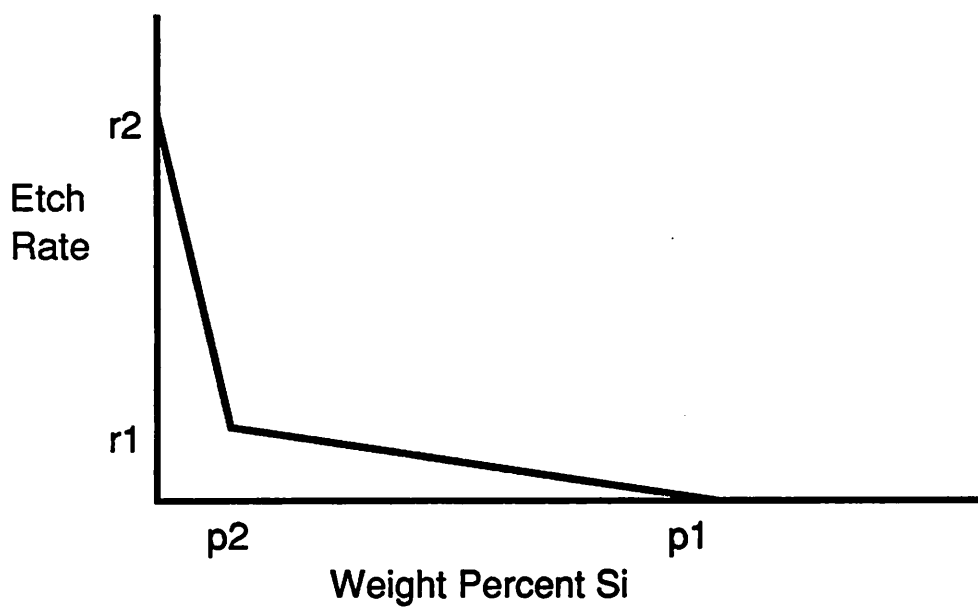


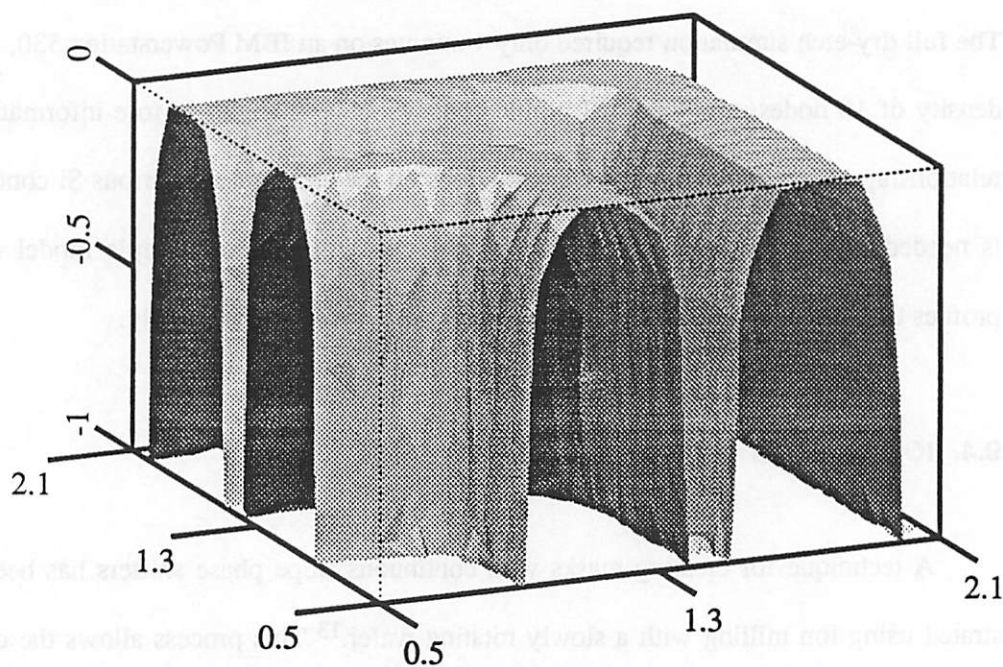
Fig 9.4: Simple rate model to relate weight percent Si in silylated resist to vertical etch rate.

The simple model does not use a full calculation of the flux distribution over the visibility solid angle, and thus is very fast in comparison with the example of the previous section. The full dry-etch simulation required only 4 minutes on an IBM Powerstation 530, for a mesh density of 10 nodes/ $\mu\text{m}$ . The final profile contains 4117 triangles. More information on the relationship of ion and neutral flux distributions on the etch rate for various Si concentrations is needed to develop a more complete model. Nevertheless, the simple model does yield profiles that show many features similar to those observed experimentally.

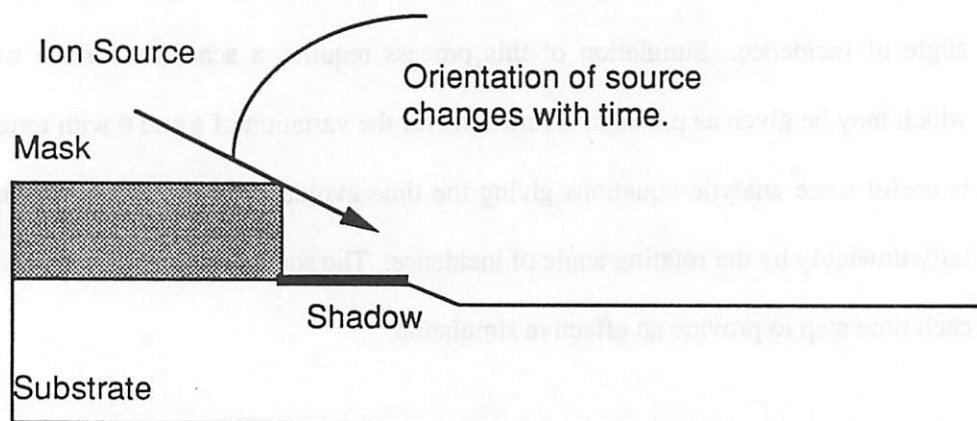
#### 9.4. ION MILLING WITH A ROTATING WAFER

A technique for creating masks with continuous slope phase shifters has been demonstrated using ion milling with a slowly rotating wafer.<sup>13</sup> This process allows the creation of smooth transitions between adjacent  $0^\circ$  and  $180^\circ$  phase-shift regions in masks created on quartz wafers. The process proceeds as shown in Fig. 9.6. A layer of CVD oxide is covered with patterned photoresist. As the wafer is rotated, the shadow cast by the photoresist moves along the wafer. Shadowed regions do not etch, and unshadowed regions etch according to the angle of incidence. Simulation of this process requires a schedule for the wafer rotation, which may be given as parametric functions for the variation of  $\phi$  and  $\theta$  with time. Simulation is useful since analytic equations giving the time evolution of the surface are made particularly unwieldy by the rotating angle of incidence. The source orientation may be updated with each time step to provide an effective simulation.

Simulation of ion milling of a CVD silicon dioxide layer using a rotating source is shown in Fig 9.7. The  $\text{SiO}_2$  layer is covered by 2.0  $\mu\text{m}$  of photoresist which is patterned to reveal a strip of oxide,  $4.0\mu\text{m}$  wide and  $8.0\mu\text{m}$  long. The sputter yield curve for oxide assumes  $S(\alpha) = 10.7\cos\alpha - 11.7\cos^2\alpha + 2.0\cos^4\alpha$ , and the etch rate at normal incidence is  $0.00027\mu\text{m/s}$ .



**Fig 9.5:** Simulation of dry-etched silylated t-BOC resist.



**Fig 9.6:** Schematic of rotating source used to create a sloped taper on a substrate.



There is significant mask erosion since the resist etch rate is nearly equal to the oxide etch rate, but to emphasize the effect of the rotation on the end result, the simulation assumes no resist loss and no material redeposition. For an oxide etched to a depth of less than about  $0.3\mu\text{m}$ , the resist loss has only a small effect on the final oxide slope. The ion beam starts at normal incidence, and rotates so that  $\phi$  decreases to  $-90^\circ$  after 40 minutes. Fig. 9.7a shows a 3D view and Fig. 9.7b shows the cross sections at 10 minute intervals. Fig. 9.7c shows the cross sections that result when  $\phi$  varies in the opposite direction, from  $-90^\circ$  to  $0^\circ$ . A simulation with  $\phi$  and  $\theta$  both varying from  $0^\circ$  to  $-90^\circ$  over a 40 minute interval is shown in Fig. 9.7d. The simulations required approximately 1 minute each on an IBM Powerstation 530.

## 9.5. PHASE-SHIFT LITHOGRAPHY WITH SIMPLE DELOOP

SAMPLE-3D has a simple deloop routine which identifies non-physical parts of the surface as all those mesh vertexes which have entered non-material cells. The loop is removed by collapsing all the triangles which contain three vertexes in non-material cells. A lithography development simulation with standing waves demonstrates the application of the simple deloop function. Fig. 9.8a shows an aerial image simulation with SPLAT of a phase shift mask. The bottom half of the mask is opaque. The top half has, from left to right, a  $180^\circ$  clear feature, a  $90^\circ$  clear feature and a  $0^\circ$  clear feature. The aerial image simulation assumes a wavelength  $\lambda=436\text{nm}$ , partial coherence  $\sigma=0.7$ , and numerical aperture 0.5. The exposure dose in  $0.713\mu\text{m}$  thick KTI 820 photoresist was  $180\text{ mJ/cm}^2$ . The resist development used Dill's Model<sup>10</sup> with  $E_1 = 5.63$ ,  $E_2 = 7.43$ , and  $E_3 = -12.6$ .

The development simulation after 6 and 12 seconds is shown in Figs. 9.8a and 9.8b. The simulation used a ray-trace method, which would normally give large loops at the standing wave edges in the resist profile. Instead, these loops were removed as they occurred. The loop

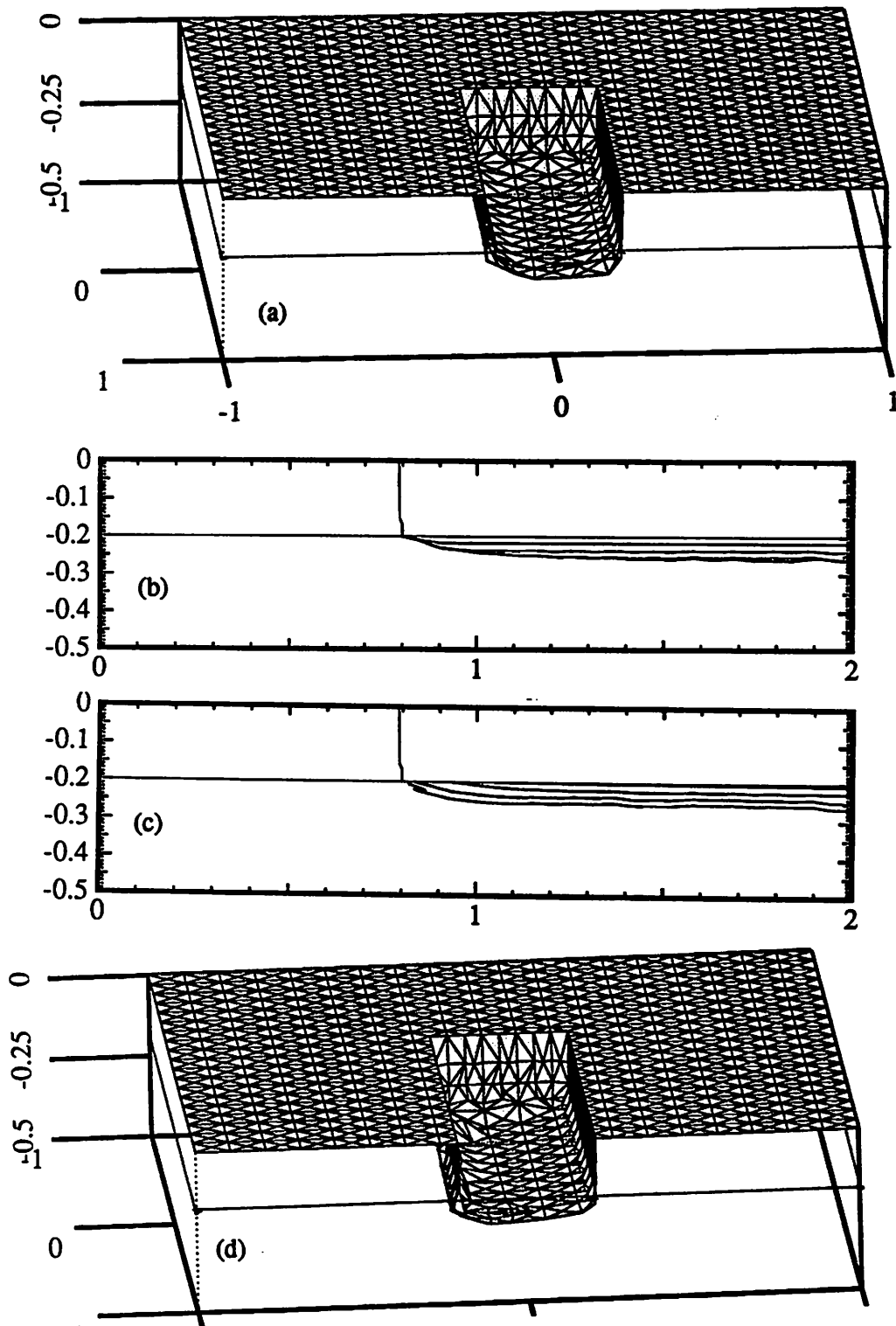


Fig. 9.7: Ion milling with a rotating source. (a) 3D view with decreasing angle. (b) Cross-section for decreasing angle. (c) Cross-section for increasing angle. (d) 3D view for both  $\phi$  and  $\theta$  decreasing.

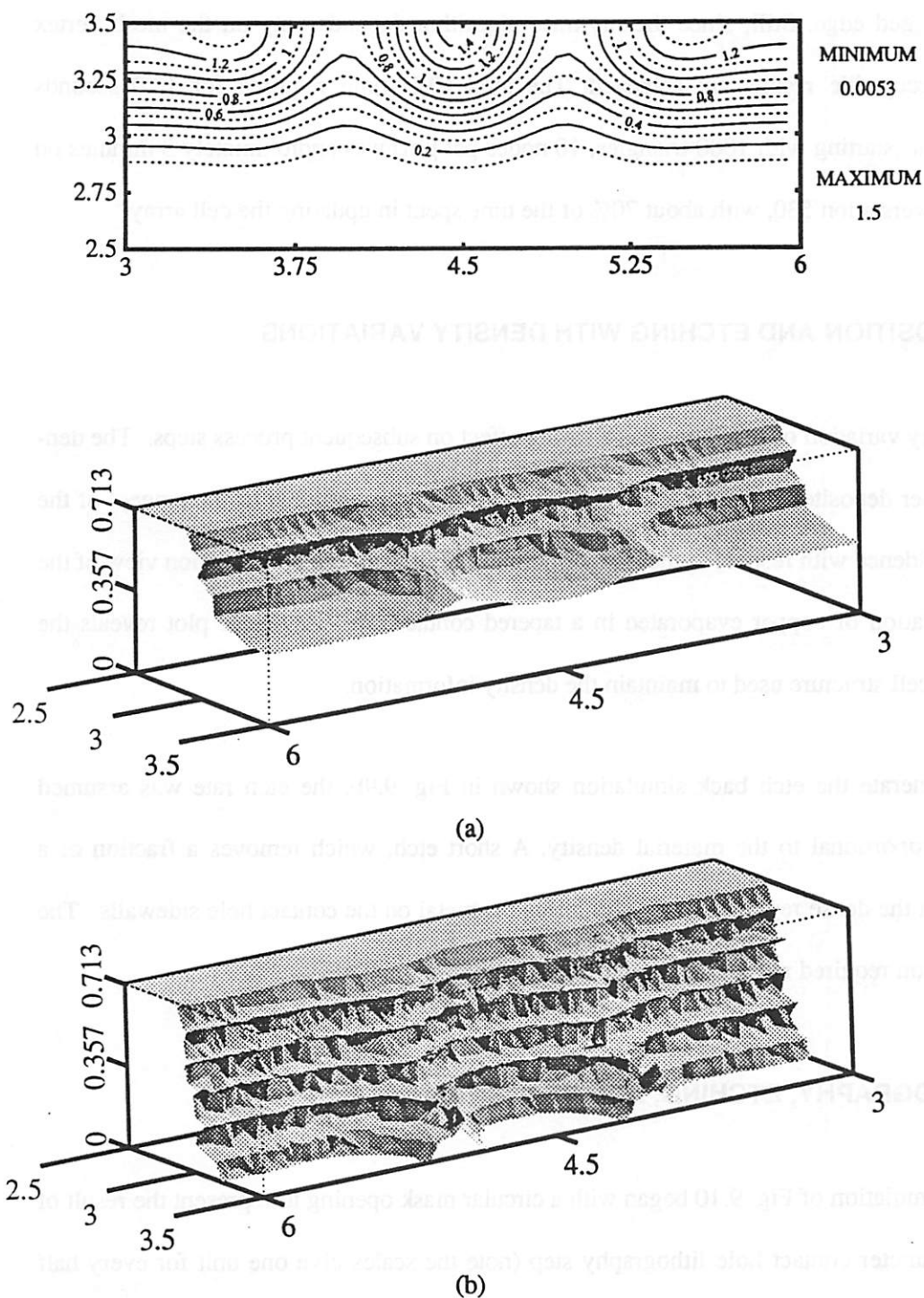


Fig. 9.8: KTI-820 resist in g-line lithography with a phase-shift mask. Standing waves resulted in loops which were removed using surface-cell hybrid.  
 (a) 6 seconds development, (b) 12 seconds development.

removal does not calculate the exact triangle intersections at the start of the loop, but instead leaves a ragged edge. Still, since the ray-trace algorithm depends only on the mesh vertex position, acceptable results are achieved. The total CPU time required for 12.0 seconds development (starting with 1000 triangles, 10 nodes per  $\mu\text{m}$ ) was approximately 3 minutes on an IBM Powerstation 530, with about 70% of the time spent in updating the cell array.

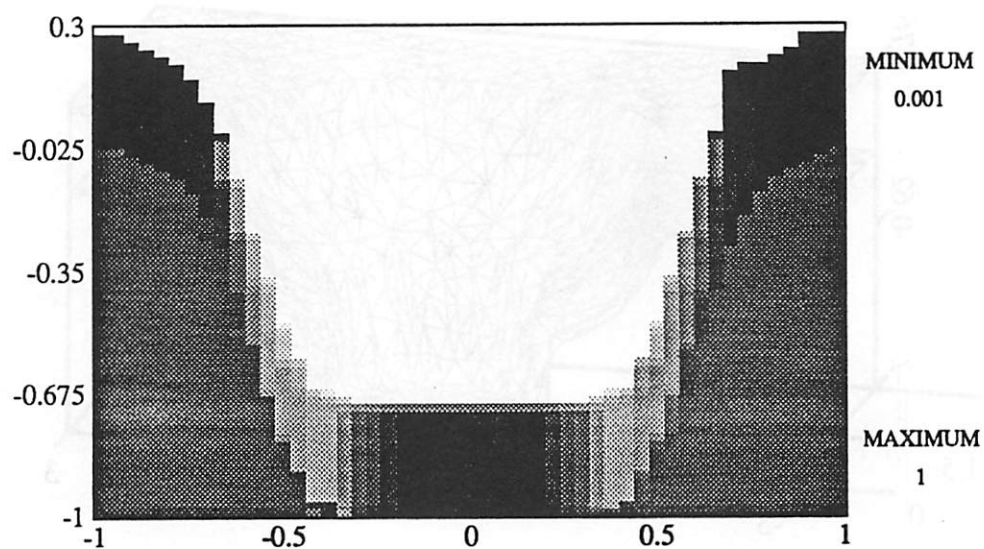
## 9.6. DEPOSITION AND ETCHING WITH DENSITY VARIATIONS

Density variation of thin films has a strong effect on subsequent process steps. The density of copper deposited on a sloped sidewall was assumed proportional to the tangent of the angle of incidence with respect to the surface normal.<sup>14</sup> Fig. 9.9a is a cross-section view of the density variation of copper evaporated in a tapered contact. The grey scale plot reveals the underlying cell structure used to maintain the density information.

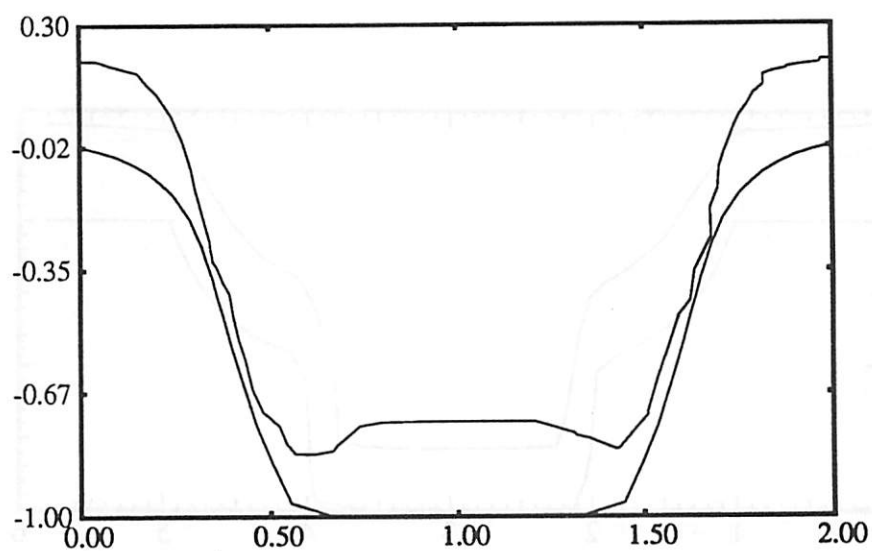
To generate the etch back simulation shown in Fig. 9.9b, the etch rate was assumed inversely proportional to the material density. A short etch, which removes a fraction of a micron from the dense regions, nearly eliminates the metal on the contact hole sidewalls. The full simulation required about one minute of CPU time.

## 9.7. LITHOGRAPHY, ETCHING, AND DEPOSITION

The simulation of Fig. 9.10 began with a circular mask opening to represent the result of a  $1.0\mu\text{m}$  diameter contact hole lithography step (note the scales give one unit for every half micron.) A wet etch step attacked the substrate isotropically to a depth of  $0.5\mu\text{m}$ . This was followed by a simple directional etch simulation for  $0.5\mu\text{m}$  additional material removal. Finally, 0.4 of metal was deposited assuming surface migration with  $\sigma=0.5\mu\text{m}$ , and a cosine

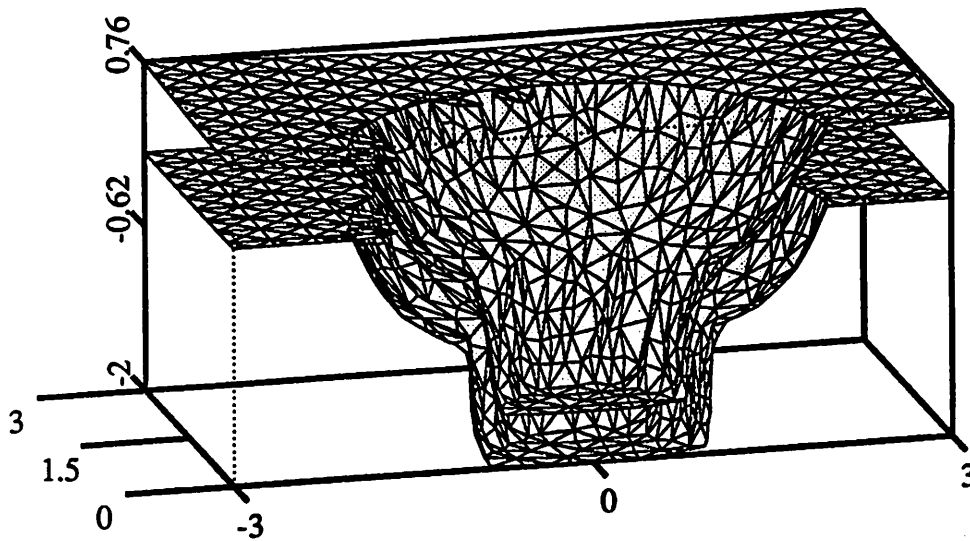


(a)

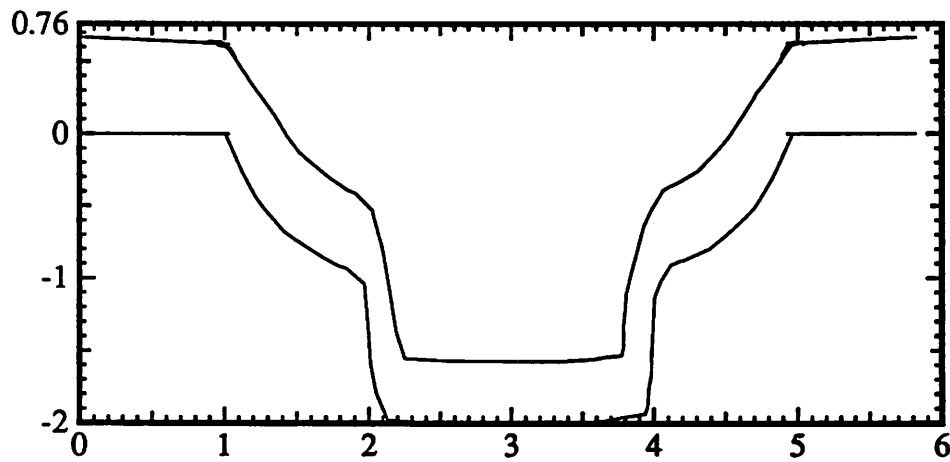


(b)

Fig. 9.9: Evaporation of copper with orientation dependent density variation.  
 (a) Cross-section of deposited film. (b) Line edge profile after wet etch.



(a)



(b)

**Fig. 9.10:** Multistep example: Wet etch, dry etch, sputter deposition. (a) Cutaway of 3D simulation. (b) 2D cross-section view.

distribution of the incoming metal flux.

The parameters and dimensions are similar to those used by Fujinaga *et. al.* an published at IEDM 1990 for a multistep processes simulated with 3D-MULSS<sup>15</sup> There are several similarities but also some differences. First, this simulation assumes a round mask, whereas Fujinaga's result is slightly square shaped (due to his more complete lithography simulation.) Also, there is a difference in the deposition. Fujinaga's result shows a thinner film at the bottom of the contact hole, and more rounded and protruding surfaces at the corners of the deposited film. This difference is likely due to different flux distribution and surface diffusion models.<sup>16</sup> Fujinaga's result using a diffusion equation method, required 50 minutes CPU time on a vector processor supercomputer (capable of 1.5GFlops peak).<sup>16</sup> The result here, using 45x90 hemisphere elements, required about 62 minutes on an IBM Powerstation 530, with about 60 minutes spent performing the deposition simulation. This indicates that the algorithms in SAMPLE-3D appear to be an order of magnitude faster than those using the diffusion equation method in 3D-MULSS. However, more work and experimental verification is needed to understand the differences between the programs and to evaluate the correctness of the algorithms and the dominant physical models. Also, the above example does not accurately compare the diffusion method for surface advancement with SAMPLE-3D, for cases which do not have complex flux distribution and visibility calculations.

## 9.8. LITHOGRAPHY SIMULATION WITH RAYS AND CELLS

To understand some of the issues involved in comparing surface-advancement and cell-removal algorithms, the development of the same rate matrix was performed using different algorithms. the rate matrix was generated assuming an exposure wavelength of  $\lambda=365\text{nm}$ , a numerical aperture  $\text{N.A.}=0.32$ , a partial coherence  $\sigma=0.5$ , a dose of  $150\text{mJ}/\text{cm}^2$  and a mask

with 0.5 $\mu\text{m}$  minimum features and 0°, 90°, and 180° phase transitions. The resist model assumed Olin Hunt HiPR 6512 with Dill parameters  $A=1.05$ ,  $B=0.11$ , and  $C=0.012$ . The post exposure bake of 120°C for 60 seconds was modeled by assuming a diffusion with a characteristic length of 0.08 $\mu\text{m}$ . The rate is related to the amount of photoactive compound (PAC) remaining  $M$  by the Kim model:<sup>17</sup>

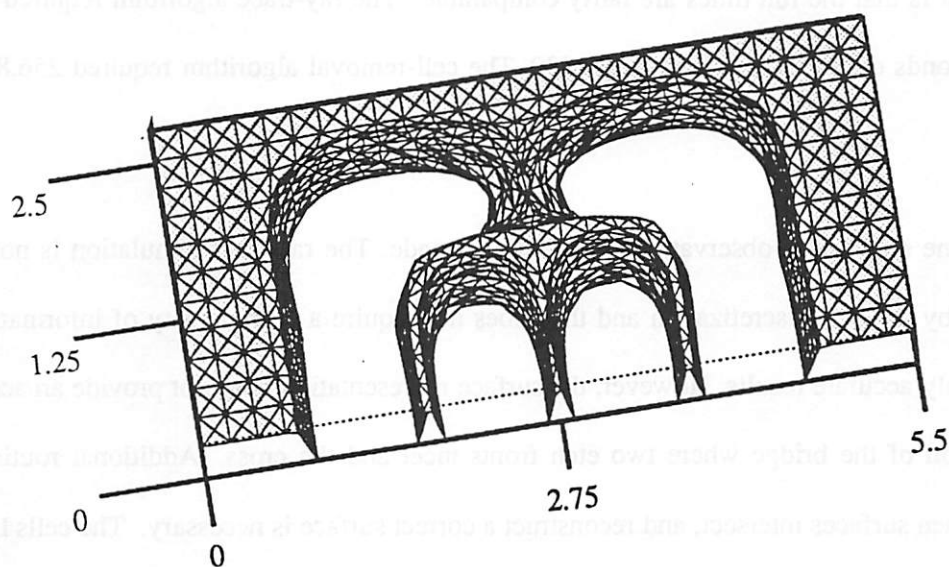
$$\text{rate}(x,y,z) = \frac{1}{\frac{1 - M(x,y,z) \cdot \exp[-R_3(1 - M(x,y,z))]}{R_1} + \frac{M(x,y,z) \cdot \exp[-R_3(1 - M(x,y,z))]}{R_2}} \quad (9.6)$$

A fit to experimental data yielded  $R_1=0.0759\mu\text{m}/\text{sec}$ ,  $R_2=0.001\mu\text{m}/\text{sec}$ , and  $R_3=7.46$ .†

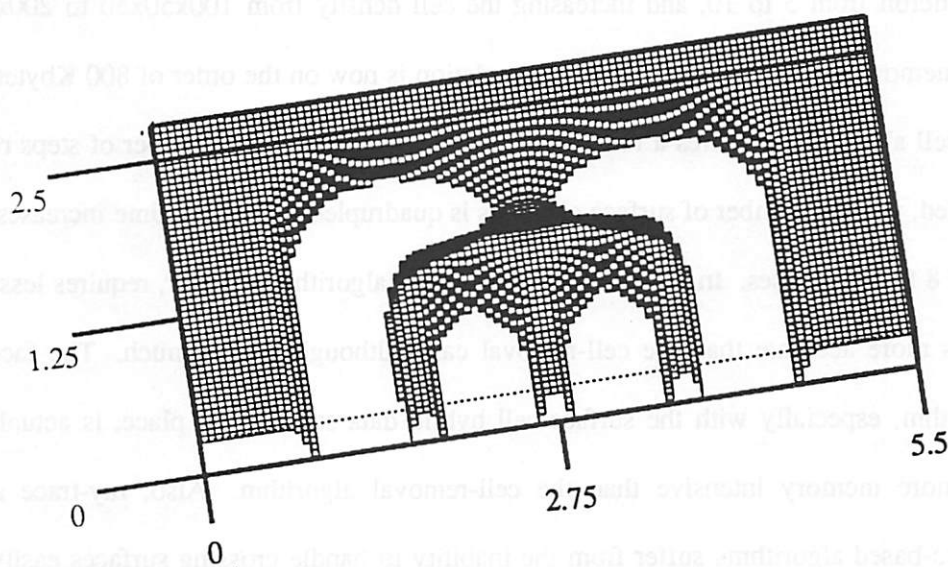
Fig. 9.11a shows the result of a ray-trace simulation, and Fig. 9.11b, gives the result of a cell removal simulation. In both cases, the resist broke through to the substrate after 12 seconds and the development was completed at 60 seconds. In both cases, the rate matrix required 1.84Mbytes of memory. The ray-trace simulation began with 5 nodes/micron corresponding to 1144 initial triangles, 1753 initial nodes, and 610 initial nodes requiring approximately 200 Kbytes of memory. (A scaled-down version of SAMPLE-3D was used for this simulation, requiring less memory than the general SAMPLE-3D data structure.) The cell removal simulation used 100x50x50 cells, with 5000 initial surface cells requiring 1.1 Mbytes. On completion, the ray-trace simulation resulted in a surface with 2957 triangles, 4596 segments, and 1640 nodes requiring approximately 400 Kbytes. The cell-removal case resulted in a final surface with 20222 cells, requiring 1.4 Mbytes to represent the topography. In both cases the time step was determined automatically, but the ray-trace simulation allowed larger step sizes than the cell removal algorithm since the minimum segment length was larger than the minimum cell edge. However, the calculation of the ray equation at each node

† Measurements performed with a Perkin-Elmer development rate monitor, for exposure in a GCA 6200 i-line stepper. A.K Pfau, and K. Chan assisted in measuring the resist development.





(a)



(b)

**Fig. 9.11:** Comparison of ray-trace and cell-removal algorithms for phase-shift lithography at i-line in Olin Hunt HiPR 6512. (a) ray-trace result  
(b) cell-removal result.

requires more operations than the determination of the volume removed from each cell. The end result is that the run times are fairly comparable. The ray-trace algorithm required 202.8 CPU seconds on an IBM Powerstation 530. The cell-removal algorithm required 256.8 CPU seconds.

Some qualitative observations may also be made. The ray-trace simulation is not constrained by the cell discretization and thus does not require a high density of information to give highly accurate results. However, the surface representation does not provide an accurate description of the bridge where two etch fronts meet and the cross. Additional routines to check when surfaces intersect, and reconstruct a correct surface is necessary. The cells handle this properly without additional computation.

Another interesting exercise is to increase the accuracy by doubling the number of nodes per micron from 5 to 10, and increasing the cell density from  $100 \times 50 \times 50$  to  $200 \times 100 \times 100$ . The memory required for the ray-trace simulation is now on the order of 800 Kbytes initially. The cell algorithm, requires a full 8.4 Mbytes. In both cases, the number of steps required is doubled, and the number of surface elements is quadrupled, so the run time increases by a factor of 8 for both cases. In conclusion, the ray-trace algorithm is faster, requires less memory, and is more accurate than the cell-removal case, although not by much. The facet-motion algorithm, especially with the surface-cell hybrid data structure in place, is actually slower and more memory intensive than the cell-removal algorithm. Also, ray-trace and other surface-based algorithms suffer from the inability to handle crossing surfaces easily, thus an effective deloop algorithm is of critical importance.

## 9.9. SUMMARY AND CONCLUSION

The collection of simulation examples presented here demonstrate the use of a wide range of topography simulation models. In many cases it was possible to combine multiple process steps into one simulation run. The required CPU time varied from tens of seconds for simple isotropic etching, to several hours for full calculation of solid angle visibility for surface meshes with thousands of vertexes. The memory requirements likewise varied from 500Kbytes for simple surfaces to 20Mbytes for multiple layers and full use of the cell array. The CPU and memory requirements are consistent with the theoretical estimates which could be derived by applying the analysis of sections 8.2.6 and 8.2.7.

The simulations also reveal several cases where three-dimensional simulation and the unique capabilities of SAMPLE-3D are useful. The pattern transfer example reveals how the interrelationship of visibility and the relative strength of various etch components can affect the final feature shape. The silicon silylation example demonstrates the need to consider inhomogeneous material properties (in this case Si concentration) in a dry-etch process. The shadow effects in ion milling with a rotating wafer also underscore the importance of 3D geometries. Phase shift lithography with standing waves demonstrates the need for deloop algorithms as well as the need for 3D simulation when the final resist profile does not correspond exactly to a constant image intensity contour. Etching and deposition with density variations again reveal the need to consider inhomogeneous material properties, which are introduced in one process step and have an effect on a later step. A multistep example was useful for comparison with a different reported 3D topography simulator. Finally, a comparison of ray-trace and cell-removal algorithms for lithography development showed that the ray-trace method is superior in terms of memory use and accuracy, but has difficulty handling the case where etch fronts cross, and is only slightly faster.

This chapter also points to the need for continued characterization of processes and additional evaluation of the algorithms in SAMPLE-3D. Although it was beyond the scope of this work, it would be desirable to provide comparisons of simulation and experiment for specific cases. Such comparisons could be used to provide additional insight into the correctness of both physical models and simulation algorithms.

## REFERENCES

1. M. Fujinaga, N. Kotani, T. Kunikiyo, H. Oda, M. Shirahata, and Y. Akasaka, "Three-Dimensional Topography Simulation Model : Etching and Lithography," *IEEE Transactions on Electron Devices*, vol. ED-37, no. 10, pp. 2183-2192, October 1990.
2. K.K.H. Toh, "Two-Dimensional Images with Effects of Lens Aberrations in Optical Lithography," *M.S. Thesis, Memorandum No. UCB/ERL M88/30*, University of California, Berkeley, May 20, 1988.
3. F.H. Dill, W.P. Homberger, P.S. Hauge, and J.M. Shaw, "Characterization of Positive Photoresist," *IEEE Transactions on Electron Devices*, vol. ED-22, no. 7, pp. 445-452, July 1975.
4. R.A. Ferguson, *Modeling and Simulation of Reaction Kinetics in Advanced Resist Processes for Optical Lithography*, pp. 133-134, Ph.D. Dissertation, University of California, Berkeley, May 1991.
5. S.A. MacDonald, H. Ito, H. Hiraoka, and C.G. Wilson, "A New Oxygen Plasma-Developable UV-Sensitive Resist," *SPE Proceedings, Tech. Conf. Photopolymers - Principles, Processes and Materials*, p. 177ff, 1985.
6. B. Roland, R. Lombaerts, C. Jakus, and F. Coopmans, "The Mechanism of the DESIRE Process," *Proceedings SPIE*, vol. 771, p. 111f, 1987.
7. R.-J. Visser, J.P.W. Schellekens, M.-E. Reuhman-Hisken, and L.J. van IJzendoorn, "Mechanisms and Kinetics of Silylation of Resist Layers from the Gas Phase," *Proceedings SPIE*, vol. 771, p. 111ff, 1987.
8. C.A. Spence, S.A. MacDonald, and H. Schlosser, "Silylation of poly (t-BOC) styrene resists: Performance and Mechanisms," *Proceedings SPIE: Advances in Resist Technol-*

*ogy and Processing VII*, vol. 1262, pp. 344-357, March 1990.

9. R. Tarascon, E. Reichmanis, F. Houlihan, A. Shugard, and L. Thompson, *Polymer Engineering and Science*, vol. 29, no. 13, pp. 850-858, July 1989.
10. F.H. Dill, A.R. Neureuther, J.A. Tuttle, and E.J. Walker, "Modeling Projection Printing of Positive Photoresists," *IEEE Transactions on Electron Devices*, vol. ED-22, no. 7, pp. 456-464, July 1975.
11. F. Watanabe and Y. Ohnishi, *Journal of Vacuum Science and Technology B*, vol. 4, no. 1, p. 422, 1986.
12. G.N. Taylor, M.Y. Hellman, T.M. Wolf, and J.M. Zeigler, *Proceedings SPIE*, vol. 920, p. 290, 1988.
13. A.K. Pfau, E.W. Scheckler, D.M. Newmark, and A.R. Neureuther, "Continuous Slope Phase-Shift Masks," *11th BACUS Symposium on Photomask Technology*, SPIE, Sunnyvale, CA, September 26, 1991.
14. R.N. Tait, S.K. Dew, T. Smy, and M.J. Brett, "Density Variation of Tungsten Films Sputtered Over Topography," *Journal of Applied Physics (submitted)*, 1991.
15. M. Fujinaga, T. Kunikiyo, T. Uchida, N. Kotani, A. Osaki, and Y. Akasaka, "New Topography Expression Model and 3D-Topography Simulation of Al-Sputter Deposition, Etching, and Photolithography," *IEDM Technical Digest*, pp. 905-908, San Francisco, December 9-12, 1990.
16. M. Fujinaga, *Personal Communication*, June 1991.
17. D.J. Kim, W.G. Oldham, and A.R. Neureuther, "Development of Positive Photoresist," *IEEE Transactions on Electron Devices*, vol. ED-31, no. 12, pp. 1730-1735, December 1984.

## CHAPTER 10

### CONCLUSION

*The first lesson of philosophy is  
that we may all be mistaken*  
Will Durant

#### 10.1. LOOKING BACK AND LOOKING AHEAD

Successful three-dimensional topography simulation requires the effective integration of many fields of study, including computational geometry, algorithm design, numerical analysis, transport physics, chemistry, the mathematics of diffusion, IC fabrication technology, and the psychology of human computer interaction. This dissertation has brought many of these fields of knowledge to bear on the problem of simulating integrated circuit etching and deposition processes. Algorithms for surface advancement, and efficient visibility calculation drew from computational geometry, computer graphics, and algorithm theory. Models for physical processes drew from the literature and applied hypotheses as to the physics and chemistry underlying IC fabrication technology. Additionally, human factors research was discussed in the context of user interface design. Much has been learned in the course of this work and this chapter will summarize the main contributions of this research. Naturally, given the scope of the problem, there is also still much to be said, and this chapter will attempt to peer into those as yet uncharted realms.

A main contribution of this work has been the demonstration of a practical 3D topography simulator for wide range of physical processes. The system integrates lithography simulation with etching and deposition simulation, and has a wide range of supporting graphics capability. The software runs on an engineering workstation and can handle surfaces described by in excess of 10,000 triangular facets. The simulator includes models for wet

chemical etching, plasma etching, ion milling, thin film evaporation, sputter deposition, and to a lesser extent, chemical vapor deposition. Secondary effects such as damage enhanced plasma etching, surface migration, thin film column orientation, and density variation are also included. The algorithms for surface advancement, visibility, and surface migration are among the most widely applicable and efficient yet reported for topography simulation. The surface advancement and visibility algorithms require  $O(N)$  time. Reflection, surface diffusion, and surface migration have been identified as necessary for many process models, but also computationally difficult - requiring worst case  $O(N^2)$  time in the current implementation. The combination of models and algorithms in SAMPLE-3D provides significant simulation capability, which not only provides superior CPU and memory efficiency, but also brings more models together in one place than any other reported topography simulator.

It is now possible to explore problems in pattern transfer technology which could not be addressed by simulation before. Regions of several square micrometers may be subjected to the simulation of multiple process steps to better understand the interaction of patterns, processes and materials. The 3D capability allows continued model development, since many proposed physical mechanisms may be implemented and tested. The relative importance of reflection and surface diffusion models can then be sorted out by comparing 3D simulations with 3D test structures, providing additional information not available to 2D simulation. Also, the inherently three-dimensional nature of current technologies for 64 Mbit DRAMs and beyond, requires 3D simulation capability to fully explore the trade-offs among various device designs.

The flexible framework for models and applications supports diverse approaches to 3D simulation. Models may be turned on and off to balance the need for generality with the need for efficiency. Software for new algorithms and models may be included without excessive



investment in programmer time. This is a critical issue for the continued applicability of 3D topography simulation, since new processes and new physical models are proposed at a rapid rate. To be useful, simulation must keep up with the state of the art in processes, models, and algorithms. As a research framework, SAMPLE-3D is also useful in the research environment as a foundation for further innovative work by others, in all aspects of 3D topography simulation. SAMPLE-3D has already supported the implementation of a wide variety of models, and has served as a basis for comparing such diverse algorithms as ray-trace, facet-motion, and cell-removal for topography simulation.

A general facet-motion algorithm was presented for isotropic, directional and highly orientation dependent processes, which also considers material boundaries, mask borders, and intersection ambiguities. The algorithm uses the motion of the individual facets in one time step to construct a new surface representation, based on facet intersections and the fastest and slowest directions within the solid angle swept out by the facets at each node. The analysis of the algorithm addressed several issues of accuracy, memory usage, and efficiency. The facet-motion algorithm showed best results when the mesh was made most dense where the spatial variation in etch rates was greatest. Also, the time step must be kept small so that the maximum distance traveled by any node is less than 10-20% of the minimum segment length. The time step must be smallest for geometries with sharp corners. Although the facet-motion algorithm is less prone to loop formation than ray-trace algorithms, loops can still occur at very sharp corners. When loops occur they rapidly lead to unstable surfaces, indicating the critical need for an efficient, robust and accurate deloop routine.

The new surface-cell hybrid data structure provided the basis for implementing efficient shadow, solid-angle visibility, and loop identification routines. Fast algorithms were presented for updating the cell array as the surface advances through it. Empirical results show that for

a surface mesh with 10 nodes/micron initially, 20-25 cells/micron is sufficient to preserve accuracy. Once the cells are defined, very fast algorithms for shadowing, visibility, and loop identification are possible, since the cell array allows rapid determination of material properties given the location of any point in the simulation region. Algorithms for reflection are more time consuming, but may also use the cells for improved efficiency.

The analysis of the algorithms and data structures allows the estimation of CPU and memory requirements for specific models. Benchmark estimates for specific examples provide a basis for estimating the actual run time and memory needed for a specific simulation case. The estimation rules are supported by the the examples presented throughout the dissertation.

Other reported approaches to 3D topography simulation were also carefully reviewed. Volume removal algorithms appear to offer results sufficient for practical development simulation, especially when etch fronts cross or tunnels appear, although they are not as accurate and memory efficient as ray-trace algorithms. This work presented a new cell removal algorithm which is both fast and very computationally efficient (requiring 5-10 minutes and less than 10 Mbytes for typical problems.) The algorithm achieves memory efficiency by storing information only where it is needed, in a dynamically allocated surface cell list. The speed is achieved using simple volume removal calculations and a simple spillover technique to allow a constant time step over an entire surface.

Some accuracy problems with cell removal algorithms have been identified, but practical and robust lithography simulation is nevertheless possible with such approaches. Unfortunately, cell removal algorithms do not appear well suited to the general etching and deposition problem when oblique surface orientations are a factor. Cell algorithms may be applied if the individual cells are used to store information about atomic or molecular level material

interactions. One variation on the cell-removal technique, called a "network" method shows promise but appears to be computationally intensive and shows some local error at convex corners. Nevertheless, the combination of surface and cell information in the network method is similar in many respects to the surface-cell hybrid approach, thus allowing the best features of both approaches to be brought to topography simulation. Another approach using a diffusion equation method, is also robust and applicable to many mass-transport models. However, it too is very computationally intensive and does not appear well suited to problems with a strong dependence on surface orientation.

Ultimately, this work has shown that there is no one best way to address all of the problems in 3D topography simulation. Instead, an approach which allows a variety of models and algorithms to be brought together in one place is more flexible, more general, and more likely to be useful in the long run. There continue to be competing algorithms (cell-removal, ray-trace, facet-motion etc.) and competing models (surface diffusion, reflection based, ballistic transport, etc.). Among the capabilities of simulation is the ability to test hypotheses and sort out the relative importance of proposed physical mechanisms. Thus, continued model development is enhanced by the existence of a software platform for exploring various approaches. This work supports the philosophy underlying the CAD Framework Initiative TCAD Framework Group's efforts to promote tool integration and model development by providing functional access to a semiconductor wafer representation database.

The original goals of this work was to develop a practical system for 3D topography simulation and provide a comprehensive perspective on the overall problem as a foundation for continued research. There are many areas which still must be explored further, if 3D topography simulation is to be useful into the next century. These issues are discussed in the next four sections on data representations, algorithms, process models and software

organization. All of these areas should prove fertile ground for continued research.

## **10.2. IMPROVING THE DATA REPRESENTATION**

The current implementation of SAMPLE-3D supports multiple planar layers and up to five non intersecting surface meshes to describe the overall topography. The surface mesh data structure, especially the node data type, contains many variable specific to certain operations. This is a result of the evolutionary development of SAMPLE-3D. Thus, the data structures have not been fully optimized to save space, but still allow efficiency. Additionally, the hemisphere, and surface-cell hybrid cell arrays should be dynamically allocated to allow different size array limits to be entered at run time.

There is still a need for more general topography descriptions which handle the following cases in a more complete way: material voids, regions of material which become detached from the rest of the surface, islands of material completely surrounded by other materials, multiple non-planar layers. Some of these topographies can be addressed through minor modification of the existing data structures, especially since the surface-cell hybrid can be applied to the problem of voids, tunnels, and non-planar material representations. However, there is still room for significant additional work in general 3D wafer representations, with topography rich constituent elements.

## **10.3. IMPROVING THE ALGORITHMS**

SAMPLE-3D has been designed with the self-imposed constraint of practicality. Thus SAMPLE-3D was developed on and for engineering workstations with up to 32 megabytes of physical memory, and capable of up to about 12 MFlops average performance. Algorithms

were designed to take advantage of the available memory to provide computational efficiency. The trade-offs of memory, speed, and accuracy always remained a concern. However, computer capabilities are increasing at a rather significant rate. In just 4 years, the computer power available to the technology CAD research group at the University of California, has improved nearly 100 fold. There is every reason to expect that this trend will continue, so many of the limits imposed by current engineering workstations will be reduced over time. Nevertheless, there is still room for work in many areas related to algorithms and computation.

*Algorithms:* There is still a pressing need for continued development of accurate and efficient algorithms. The surface-cell data structure, while very efficient, still loses some accuracy by discretizing the geometry one level beyond that imposed by the surface mesh. There is also room for better deloop algorithms. The surface-cell based method is good at locating loops, but an effective scheme for removing the loops and reconstructing a stable surface mesh has not been fully implemented in SAMPLE-3D. Efficient algorithms for surface reflection calculation are still needed. This is among the most difficult visibility problems encountered in topography simulation, and is computationally expensive even in 2D simulation. A surface diffusion equation solver would prove useful, since it may be applied to models for both etching and deposition. There has also been much work in solid modeling, surface representations and computational geometry which may prove useful to 3D topography simulation.

*Advanced Hardware:* There is high degree of similarity between computer graphics and many of the geometric operations needed for 3D topography simulation. This naturally suggests that graphics workstations might be brought to bear on the surface evolution and visibility calculation problems. Graphics hardware already provides many of the vector operations required in topography simulation. Another advance in hardware is massively parallel

computation. Many algorithms in process simulation are inherently parallel. For example, all operations required at every surface point at every time step (*e.g.* visibility, surface migration, advancement) may be performed simultaneously. The difficult question is how to allocate the necessary information to the individual processors without sacrificing efficiency.

#### 10.4. IMPROVING THE MODELS

There is also much to do in the realm of physical process modeling. Many of the basic surface reactions are not well understood. In the meantime, a well-chosen set of parameters may be used to describe the results of experiments to characterize certain processes, or even individual pieces of equipment. The true test is whether the simulation results match experimental data. Such comparisons have not been fully undertaken for 3D topography simulation.

There is a need to continue developing models to describe the transport of particles from a source or a plasma, to the wafer surface. Many of the models developed for both etching and deposition require a well-characterized flux distribution. Continued characterization of particle flux distributions as Gaussians or hypercosines, will allow this information to be presented in a convenient form to SAMPLE-3D.

IN SAMPLE-3D the process models are primarily macroscopic in nature. They do not fully consider underlying atomic or molecular level interactions. As integrated circuit features progress into the nanometer regime, it will be increasingly necessary to consider atomic level interactions in more detail. Macroscopic models will still be useful for many years, but further investigation of microscopic models should prove useful. Additionally, microscopic level investigations may provide insight into new and better macroscopic models, in much the way ballistic simulations have produced results relevant to the deposition simulations developed in

chapter 7. Eventually it may be possible to derive sputter yield curves from simulations of ion trajectories through solids, to predict plasma etch rates from models for surface reactions, to estimate surface roughness from atomic level simulations, and to simulate a host of other important phenomena using fundamental physical models.

## 10.5. SOFTWARE REORGANIZATION

SAMPLE-3D as currently implemented is a piece of development software which has undergone many changes, and has had many elements added, removed, and otherwise altered. As a prototype tool for exploring algorithms and approaches to 3D topography simulation, it has served its purpose well. However, it is not yet an industrial grade program, free of errors. The time is ripe for a major rewrite of the software, building on the experiences which have gone in to its development.

To reorganize SAMPLE-3D, first many redundant routines must be replaced by a set of consistent and concise functions. For example, there are currently two different dot product routines, which should be replaced by one. Second, the hierarchical functional organization should be reflected more clearly in the software. Ultimately, functions from vector operations to complex visibility should be organized as a library, with consistent and well documented arguments and operations. This will make the concept of using SAMPLE-3D as a software platform for continued model development a practical reality. Third, the data structures should be streamlined to remove redundant or unused data elements. The global variables should also be reorganized. Many of the above ideas naturally suggest an object-oriented approach to programming, perhaps with a language such as C++. Fourth, many of the process models are quite modest in scope. As new information becomes available, the models should be improved. Fifth, it would also be useful to organize the models so they are easily reducible

to the 2D case. Many problems in process technology may be addressed by 2D simulation, with considerable savings in memory and CPU requirements. A fully general object oriented approach would be applicable to 2D data objects as well as 3D data objects. Addressing all of these areas would lead to the development of a robust and powerful 3D topography simulation tool based on the lessons learned in the development of the prototype SAMPLE-3D.

There is also room for significant progress in the field of 3D TCAD tool integration. The basic data representation in SAMPLE-3D is a starting point for more general 3D topography representations. There is also a need for communication between surface based and cell based topography representations, including atomic level data representations. All of this should then be made compatible with 3D impurity simulations and device modeling.

## **10.6. THE LAST WORD**

It is said that good science raises as many questions as it answers. Certainly, work on SAMPLE-3D has raised many issues both practical and theoretical. The intersection of the wide number of fields necessary to develop 3D process simulation has been a rich source for innovative ideas and new research. Work on this project has provided many opportunities for creating new approaches to the 3D topography simulation problem and has required numerous judgements to make the trade-offs necessary in building a practical system. It is hoped that this research will prove as useful as it has been interesting. The goal of this work has been to advance the state of the art in practical 3D topography simulation. It is the role of the reader and the simulation user to judge whether this has been achieved.



## BIBLIOGRAPHY

- W.L. Engl, *Process and Device Modeling*, Elsevier Science Publishers, B.V., Amsterdam, 1986.
- S.M. Sze, *VLSI Technology*, McGraw-Hill Book Co., New York, 1988.
- D.M. Manos and D.L. Flamm, *Plasma Etching, an Introduction*, Academic Press, Boston, 1989.
- "DEPICT-2," *Technology Modeling Associates*, 1990.
- SAMPLE 1.7a User's Guide*, Electronics Research Laboratory, University of California, Berkeley, 1990.
- SAMPLE 1.8 User's Guide*, Electronics Research Laboratory, University of California, Berkeley, 1991.
- Y. Akiyama, Y. Hatanaka, M. Asou, Y. Tamegaya, H. Ikeuchi, and H. Kuge, "Development of a unified process and device simulation environment," *1991 International Workshop on VLSI Process and Device Modeling*, pp. 92-93, Oiso, Japan, May 1991.
- H.H. Andersen and H.L. Bay, "Sputtering Yield Measurements," in *Sputtering by Particle Bombardment I*, ed. R. Behrisch, vol. 47, pp. 201-202, Springer-Verlag, Berlin, 1981.
- G. Arfken, *Mathematical Methods for Physicists*, Academic Press, Orlando, Florida, 3rd edition, 1985.
- H. Bach, "Messung der Zerstauebungsraten an Nichtleitern und Bestimmung der Bindungsenthalpie von Kieselglas," *Naturwissenschaften*, vol. 9, pp. 429-430, 1968.
- H. Bach, *Journal of Non-Crystalline Solids*, vol. 3, p. 1, 1970.
- E. Barouch, B. Bradie, and S.V. Babu, "Calculation of Developed Resist Profiles by Least Action Principle," *Interface'88 : Proceedings of KTI Microelectronics Seminar*, pp. 187-196, November 1988.
- E. Barouch, B. Bradie, H. Fowler, and S.V. Babu, "Three-Dimensional Modeling of Optical Lithography for Positive Photoresists," *Interface'89 : Proceedings of KTI Microelectronics Seminar*, pp. 123-136, November 1989.
- J. Bauer, "Modelle fuer den fotolithografischen Prozess," *Feingeraetetechnik*, vol. 29, p. 127ff, 1980.

- J. Bauer, W. Mehr, and U. Glaubitz, "Simulation and Experimental Results in 0.6  $\mu\text{m}$  Lithography using an I-Line Stepper," *Proceedings of SPIE: Optical/Laser Microlithography III*, vol. 1264, pp. 431-445, March 1990.
- K.E. Bean, "Anisotropic Etching of Si," *IEEE Transactions on Electron Devices*, vol. ED-25, p. 1185, 1978.
- E.B. Becker, G.F. Carey, and J.T. Oden, *Finite Elements, An Introduction*, I, p. 246, Prentice-Hall, Englewood Cliffs, New Jersey, 1981.
- R. Behrisch, *Sputtering by Particle Bombardment I*, 47, Springer-Verlag, Berlin, 1981.
- D.A. Bernard, "Simulation of Focus Effects in Photolithography," *IEEE Transactions on Semiconductor Manufacturing*, vol. 1, no. 3, pp. 85-97, August 1988.
- P.H. Berning, "Theory and Calculations of Optical Thin Films," in *Physics of Thin Films*, ed. George Hass, vol. I, pp. 69-121, Academic Press, New York, 1963.
- I.A. Blech, "Evaporated Film Profiles Over Steps in Substrates," *Thin Solid Films*, vol. 6, pp. 113-118, Elsevier Sequoia S.A., 1970.
- I.A. Blech and H.A. VanderPlas, "Step coverage simulation and measurement in a dc planar magnetron sputtering system," *Journal of Applied Physics*, vol. 54, p. 3489ff, 1983.
- I.A. Blech, "Step Coverage of Vapor Deposited Thin Aluminum Films," *Solid State Technology*, pp. 123-125, December 1983.
- J. Bloem and L.J. Gilling, "Epitaxial Growth by Chemical Vapor Deposition," in *VLSI Electronics*, ed. N. G. Einspruch and H. Huffs, vol. 12, p. 89, Academic Press, Orlando, 1985.
- D.S. Boning, M.L. Heytens, and A.S. Wong, "The Intertool Profile Interchange Format: An Object-Oriented Approach," *IEEE Transactions on Computer-Aided Design*, vol. CAD-10, no. 9, pp. 1150-1156, September 1991.
- M. Born and E. Wolf, *Principles of Optics, Sixth Edition*, Pergamon Press, London 1980.
- L. Borucki, H.H. Hansen, and K. Varahramyan, "FEDDS - A 2D semiconductor fabrication process simulator," *IBM Journal of Research and Development*, vol. 29, no. 3, pp. 263-276, May 1985.
- M.J. Brett, "Structural transitions in aggregation simulation of thin film growth," *Journal of Vacuum Science Technology A*, vol. 6, p. 1749, 1988.

- M.J. Brett, K.L. Westra, and T. Smy, *Proceedings International Electron Device Meeting*, pp. 336-339, San Francisco, CA, December 1988.
- M.J. Brett, "Simulation of structural transitions in thin films," *Journal of Materials Science*, vol. 24, p. 623, 1989.
- I. Brodie and J.J. Muray, *The Physics of Microfabrication*, Plenum Press, New York, 1982.
- R.H. Bruce and A.R. Reinberg, *Journal of the Electrochemical Society*, vol. 129, p. 393, 1982.
- P. Burggraaf, "Wet Etching Today," *Semiconductor International*, p. 48, February 1983.
- T.S. Cale, T.H. Gandy, M.K. Jain, M. Ramaswami, and G.B. Raupp, "A General Model for PVD Deposition," *Proceedings Eight International VLSI Multilevel Interconnection Conference*, pp. 350-352, Santa Clara, June 11-12, 1991.
- L.B. Carll, *A Treatise on the Calculus of Variations*, pp. 335-344, John Wiley & Sons, New York, 1881. Chapter 1, Section X.
- C. Catana, J.S. Colligon, and G. Carter, *Journal of Materials Science*, vol. 7, p. 467, 1972.
- B.A. Chen, *Plasma Processes for VLSI Pattern Transfer*, Ph.D. Dissertation, Yale University, May 1990.
- J.Y. Chen and R. Henderson, "Photo-CVD for VLSI," *Journal of the Electrochemical Society*, vol. 131, p. 2147, September 1984.
- S.M. Chiu, *PIX: Program Interface in X, A Graphical User Interface Software User's Guide*, Electronics Research Laboratory, University of California, March 1991.
- P.G. Ciarlet, "Conforming Finite Element Methods for Shell Problems," in *Mathematics of Finite Elements and Applications II*, ed. J. Whitemann, Academic Press, 1977.
- R.W. Clough and C.P. Johnson, "A Finite Element Approximation for the Analysis of Thin Shells," *International Journal of Solids Structures*, vol. 4, pp. 43-60, 1968.
- J.W. Coburn and H.F. Winters, "Ion and electron assisted gas-surface chemistry - an important effect," *Journal of Applied Physics*, vol. 50, no. 5, pp. 3189-3196, May 1979.
- J.W. Coburn, *Plasma Etching and Reactive Ion Etching*, p. 15, American Vacuum Society, New York, 1982.
- D.C. Cole, E.M. Buturla, S.S. Furkay, K. Varahramyan, J. Slinkman, J.A. Mandelman, D.P. Foty, O. Bula, A.W. Strong, J.W. Park, T.D. Linton Jr., J.B. Johnson, M.V. Fischetti, S.E. Laux, P.E. Cottrell, H.G. Lustig, F. Pileggi, and D. Katcoff, "The Use of

Simulation in Semiconductor Technology Development," *Solid-State Electronics*, vol. 33, no. 6, pp. 591-623, 1990.

C.H. Corbex, A.F. Gerodolle, S.P. Martin, and A.R. Poncet, "Data Structuring for Process and Device Simulations," *IEEE Transactions on Computer-Aided Design*, vol. 7, no. 4, pp. 489-500, April 1988.

J. Crank, *The Mathematics of Diffusion*, p. 12, Clarendon Press, Oxford, UK, 1956.

S.K. Dew, T. Smy, and M.J. Brett, "Thin Film Microstructure Simulation of RF Bias Planarized Metal Interconnects using a Ballistic Deposition Model," *Proceedings Eighth International VLSI Multilevel Interconnection Conference*, pp. 353-355, Santa Clara, CA, June 11-12, 1991.

F.H. Dill, W.P. Hornberger, P.S. Hauge, and J.M. Shaw, "Characterization of Positive Photoresist," *IEEE Transactions on Electron Devices*, vol. ED-22, no. 7, pp. 445-452, July 1975.

F.H. Dill, A.R. Neureuther, J.A. Tuttle, and E.J. Walker, "Modeling Projection Printing of Positive Photoresists," *IEEE Transactions on Electron Devices*, vol. ED-22, no. 7, pp. 456-464, July 1975.

F.H. Dill, "Optical Lithography," *IEEE Transactions on Electron Devices*, vol. ED-22, no. 7, pp. 440-444, July 1975.

A.G. Dirks and H.J. Leamy, "Columnar Microstructure in Vapor-Deposited Thin Films," *Thin Solid Films*, vol. 47, pp. 219-233, 1977.

J.P. Ducommun, M. Cantagrel, and M. Marchal, "Development of a general surface contour by ion erosion. Theory and computer simulation," *Journal of Materials Science*, vol. 9, pp. 725-736, 1974.

J.P. Ducommun, M. Cantagrel, and M. Moulin, "Evolution of well-defined surface contour submitted to ion bombardment: computer simulation and experimental investigation," *Journal of Materials Science*, vol. 10, pp. 52-62, 1975.

S.G. Duvall, "An Interchange Format for Process and Device Simulation," *IEEE Transactions on Computer-Aided Design*, vol. CAD-7, no. 7, pp. 741-754, July 1988.

D.J. Eliot, *Integrated Circuit Fabrication Technology*, McGraw-Hill, New York, 1982.

R.C. Evans, G. Koppelman, and V.T. Rajan, "Shaping geometric objects by cumulative translational sweeps," *IBM Journal of Research and Development*, vol. 31, no. 3, pp. 343-360, May 1987.

- G. Farin, *Curves and Surfaces for Computer Aided Geometric Design*, Academic Press, Boston, 1990.
- R. Ferguson, J.M. Hutchinson, C.A. Spence, and A.R. Neureuther, "Modeling and Simulation of a Deep-UV Acid Hardening Resist," *Electron, Ion and Photon Beam Sci. and Technol.*, 1990.
- R.A. Ferguson, *Modeling and Simulation of Reaction Kinetics in Advanced Resist Processes for Optical Lithography*, pp. 133-134, Ph.D. Dissertation, University of California, Berkeley, May 1991.
- W. Fichtner, "Physics of VLSI Processing and Process Simulation," in *Silicon Integrated Circuits Part C*, ed. Dawon Kahng, Academic Press, Orlando, Florida, 1985.
- D.L. Flamm, "Basic Chemistry and Mechanisms of Plasma Etching," *Journal of Vacuum Science Technology B1*, pp. 23-30, 1983.
- P.D. Flanner-III, S. Subramanian, and A.R. Neureuther, "Two-Dimensional Optical Proximity Effects," *Proceedings of SPIE: Optical Microlithography V*, vol. 633, pp. 239-244, March 1986.
- J.D. Foley, A. van Dam, S.K. Feiner, and J.F. Hughes, *Computer Graphics: Principles and Practice*, Addison-Wesley, Reading, Mass., 2nd edition, 1990.
- W.F. Foote, "Simulation of Anisotropic Crystal Etching," *M.S. Project Report*, University of California, Berkeley, September 12, 1990.
- F.C. Frank, "On the Kinematic Theory of Crystal Growth and Dissolution Processes," in *Growth and Perfection of Crystals*, ed. R.H. Doremus, B.W. Roberts, D. Turnbull, pp. 411-419, Wiley and Sons, New York, 1958.
- M. Fujinaga, N. Kotani, H. Oda, M. Shirahata, H. Genjo, T. Katayama, T. Ogawa, and Y. Akasaka, "Three Dimensional Topography Simulation model Using Diffusion Equation," *IEDM Technical Digest*, pp. 332-335, San Francisco, CA, December 11-14, 1988.
- M. Fujinaga, T. Kunikiyo, T. Uchida, N. Kotani, A. Osaki, and Y. Akasaka, "New Topography Expression Model and 3D-Topography Simulation of Al-Sputter Deposition, Etching, and Photolithography," *IEDM Technical Digest*, pp. 905-908, San Francisco, December 9-12, 1990.
- M. Fujinaga, N. Kotani, T. Kunikiyo, H. Oda, M. Shirahata, and Y. Akasaka, "Three-Dimensional Topography Simulation Model : Etching and Lithography," *IEEE Transactions on Electron Devices*, vol. ED-37, no. 10, pp. 2183-2192, October 1990.
- M. Fujinaga, "3D-Topography Simulator for Sequential Processes," *1991 International Workshop on VLSI Process and Device Modeling*, pp. 80-83, Oiso, Japan, May 26-27,

1991.

- J. Gamelin, R. Guerrieri, and A.R. Neureuther, "Exploration of Scattering from Topography with Massively Parallel Computers," *J. Vac. Sci. Technol. B.*, 1989.
- I. Garganti, T. Walsh, and O. Wu, "Viewing Transformations of Voxel-Based Objects via Linear Octrees," *IEEE Computer Graphics and Applications*, vol. 6, no. 10, pp. 12-21, October 1986.
- A. Gerodolle, "2D Aspects of Ion-Enhanced Reactive Etching of Si with SF<sub>6</sub>," *Proceedings 19th ESSDERC*, pp. 206-207, 1989.
- A. Gerodolle, C. Corbex, A. Poncet, T. Pedron, and S. Martin, "TITAN 5: a two-dimensional process and device simulator," in *Software Tools for Process, Device and Circuit Modelling*, pp. 56-67, Boole Press, Dublin, Ireland, 1989.
- A. Gerodolle, A. Poncet, C. Corbex, and S. Martin, "From Process Simulation to Device Optimization: How To Satisfy the Requirements of Both research and Industry," *1991 International Workshop on VLSI Process and Device Modeling*, pp. 86-89, Oiso, Japan, May 26-27, 1991.
- A. Gerodolle and J. Pelletier, "Two-Dimensional Implications of a Purely Reactive Model for Plasma Etching," *IEEE Transactions on Electron Devices*, vol. 38, no. 9, pp. 2025-2032, September 1991.
- R. Glang, "Vacuum Evaporation," in *Handbook of Thin Film Technology*, ed. L. Maissel and R. Glang, pp. 1-130, McGraw-Hill, New York, 1970.
- T. Gocho, Y. Morita, and J. Sato, "Trench Isolation Technology for 0.35 $\mu$ m Device by Bias ECR CVD," *1991 Symposium on VLSI Technology, Digest of Technical Papers*, pp. 87-88, Oiso, Japan, May 28-30, 1991.
- B. Gorowitz, T.B. Gorczyca, and R.J. Saia, "Applications of PECVD in VLSI," *Solid State Technology*, p. 197, June 1985.
- P.A. Gough, M.K. Johnson, P. Walker, and H. Hermans, "An Integrated Device Design Environment for Semiconductors," *IEEE Transactions on Computer-Aided Design*, vol. CAD-10, no. 6, pp. 808-821, June 1991.
- W.R. Grove, *Philosophical Transactions of the Royal Society*, p. 87, London, 1852.
- P.I. Hagouel, *X-ray Lithographic Fabrication of Blazed Diffraction Gratings*, Ph.D. Dissertation, University of California, Berkeley, 1976.
- E. Haines, "Essential Ray Tracing Algorithms," in *An Introduction to Ray Tracing*, ed. A.S. Glassner, pp. 33-77, Academic Press, London, 1989.

- I. Hanyu, S. Asai, K. Kosemura, H. Ito, M. Nunokawa, and M. Abe, "New Phase-Shifting Mask with Highly Transparent  $\text{SiO}_2$  Phase Shifters," *Proceedings of SPIE: Optical/Laser Microlithography III*, vol. 1264, pp. 167-177, 1990.
- L. Hartsough, "Resistivity of Bias-Sputtered Ti-W Films," *Thin Solid Films*, vol. 64, p. 17, 1979.
- W. Henke, D. Mewes, M. Weiss, G. Czech, and R. Schiessl-Hoyler, "Simulation of Defects in 3-Dimensional Resist Profiles in Optical Lithography," *Microelectronic Engineering*, vol. 13, pp. 497-501, 1991.
- Y. Hirai, M. Sasago, M. Endo, K. Ikeda, S. Tomida, and S. Hayama, "Three Dimensional Process Simulation for Photo and Electron Beam Lithography and Estimations of Proximity Effects," *Symposium on VLSI Technology, Digest of Technical Papers*, p. 15, 1987.
- Y. Hirai, S. Tomida, K. Ikeda, M. Sasago, M. Endo, S. Hayama, and N. Nomura, "Three-Dimensional Resist Process Simulator PEACE (Photo and Electron Beam Lithography Analyzing Computer Engineering System)," *IEEE Transactions on Computer-Aided Design of Integrated Circuits*, vol. CAD-10, no. 6, pp. 802-807, June 1991.
- C.A.R. Hoare, "Quicksort," *Computer Journal*, vol. 5, pp. 10-15, 1962.
- V. Hoffman, "High Rate Magnetron Sputtering for Metallizing Semiconductor Devices," *Solid State Technology*, pp. 57-61, December 1976.
- H.H. Hopkins, "On the Diffraction Theory of Optical Images," *Proc. Royal Soc. Series A.*, vol. 217, no. 1131, pp. 408-432, 1953.
- K.H. Huebner, *The Finite Element Method for Engineers*, John Wiley & Sons, New York, 1975.
- G.M. Hunter, "Efficient Computation and Data Structures for Graphics," Ph.D. Thesis, Department of Electrical Engineering and Computer Science, Princeton University, Princeton, NJ, 1978.
- R.L. Huston and C.E. Passerello, *Finite Element Methods: An Introduction*, Marcel Dekker, Inc., New York, 1984.
- B. Huynh, K.K.H. Toh, W.E. Haller, and A.R. Neureuther, "Optical Printability of Defects in Two-Dimensional Patterns," *J. Vac. Sci. Technol. B* 6(6), pp. 2207-2212, Nov/Dec 1988.
- T. Ishizuka, "Simulation of Three-Dimensional Negative Photoresist Images Using Phase-Shifting Mask," *Proceedings of the International Conference on Computer Applications to Materials Science and Engineering - CAMSE '90*, Tokyo, Japan, August 28-31, 1990.

- T. Ishizuka, "Three-Dimensional Simulation in Photoresist Development," *IEICE (In Japanese)*, vol. J73-C-II, no. 11, pp. 775-785, November 1990.
- T. Ito, K. Kadota, H. Fukui, M. Nagao, A. Sugimoto, M. Nozaki, and T. Kato, "Submicrometer Pattern Correction for Optical Lithography," *Proceedings of SPIE: Optical/Laser Microlithography*, vol. 922, pp. 9-17, March 1988.
- K. Itoh, "Trends in Megabit DRAM Circuit Design," *IEEE Journal of Solid-State Circuits*, vol. 25, no. 6, pp. 778-789, June 1990.
- R.E. Jewett, P.I. Hagouel, A.R. Neureuther, and T. Van Duzer, "Line-Profile Resist Development Simulation Techniques," *Polymer Eng. Sci.*, vol. 17, no. 6, pp. 381-384, June 1977.
- L. Jia, W. Jian-kun, and W. Shao-jun, "Three-Dimensional Development of Electron Beam Exposed Resist Patterns Simulated by Using Ray Tracing Model," *Microelectronic Engineering*, vol. 6, pp. 147-151, 1987.
- R.C. Johnson, "Microsystems and Software," *Electronics*, pp. 150-163, October 23, 1980.
- F. Jones and J. Paraszczak, "RD3D (Computer Simulation of Resist Development in Three Dimensions)," *IEEE Transactions on Electron Devices*, vol. ED-28, no. 12, pp. 1544-1552, December 1981.
- C.W. Jurgensen and E.S.G. Shaqfeh, "Application of the Kinetic Theory of Bombardment Induced Interface Evolution to the Pattern Transfer Step in Multi-Layer Lithography," *Proceedings SPIE: 1185 Dry Processing for Submicrometer Lithography I*.
- C.W. Jurgensen and E.S.G. Shaqfeh, "Kinetic Theory of Bombardment Induced Interface Evolution," *Journal of Vacuum Science Technology B*, vol. 7, no. 6, pp. 1488-1492, Nov/Dec 1989.
- K. Kadota, T. Ito, H. Fukui, M. Nagao, A. Sugimoto, M. Nozaki, and T. Kato, "Resist Pattern Analysis for Submicron Feature Size using 3-D Photolithography Simulator," *Proceedings of SPIE: Optical/Laser Microlithography II*, vol. 1088, pp. 94-105, 1989.
- K. Kato, N. Shigyo, T. Wada, S. Onga, M. Konaka, and K. Taniguchi, "A Supervised Simulation System for Process and Device Designs Based on a Geometrical Data interface," *IEEE Transactions on Electron Devices*, vol. ED-34, no. 10, pp. 2049-2058, October 1987.
- W. Kern and V.S. Ban, "Chemical Vapor Deposition of Inorganic Thin Films," in *Thin Film Processes*, ed. J. L. Vossen and W. Kern, pp. 257-331, New York, 1978.
- W. Kern, "Deposited Dielectrics for VLSI," *Semiconductor International*, vol. 8, no. 7, p. 122, July 1985.



- B.W. Kernighan and D.M. Ritchie, *The C Programming Language*, Prentice-Hall, Englewood Cliffs, New Jersey, 1978.
- K. Kikuta, T. Kikkawa, and H. Aoki, "0.25  $\mu$ m Contact Hole Filling by Al-Ge Reflow Sputtering," *1991 Symposium on VLSI Technology, Digest of Technical Papers*, pp. 35-36, Oiso, Japan, May 28-30, 1991.
- D.J. Kim, W.G. Oldham, and A.R. Neureuther, "Development of Positive Photoresist," *IEEE Transactions on Electron Devices*, vol. ED-31, no. 12, pp. 1730-1735, December 1984.
- G.M. Koppelman and M.A. Wesley, "Oyster: A Study of Integrated Circuits as Three-Dimensional Structures," *IBM Journal of Research and Development*, vol. 27, p. 149, 1983.
- N. Kumar and et. al., "Fabrication of RF Reactively Sputtered TiN Thin Films," *Semiconductor International*, pp. 100-104, April 1987.
- R.B. Laibowitz, A.N. Broers, J.T. C. Yeh, and J.M. Viggiano, *Applied Physics Letters*, vol. 35, p. 891, 1979.
- H. Lamb, *Hydrodynamics*, p. 7, Dover, New York, 1945.
- M.E. Law, C.S. Rafferty, and R.W. Dutton, *SUPREM-IV User's Manual*, Stanford University, 1988.
- K.Y. Lee, Y.H. Kim, and C.G. Hwang, "New Three-Dimensional Simulator for Electron Beam Lithography," *1991 International Workshop on VLSI Process and Device Modeling*, pp. 44-45, Oiso, Japan, May 26-27, 1991.
- M.D. Levenson, N.S. Viswanathan, and R.A. Simpson, "Improving Resolution in Photolithography with a Phase-Shifting Mask," *IEEE Transactions on Electron Devices*, vol. ED-29, no. 12, pp. 1812-1846, December 1982.
- R.A. Levy and M.L. Green, "Low Pressure Chemical Vapor Deposition of Tungsten and Aluminum for VLSI Applications," *Journal of the Electrochemical Society*, vol. 134, pp. 37C-49C, 1987.
- H.M. Liaw, "Surface Morphology of Selective Epitaxial Silicon," in *Semiconductor Silicon*, Electrochemical Society, Pennington, New Jersey, 1986.
- K.-K. Lin and C.J. Spanos, "Statistical Equipment Modeling for VLSI Manufacturing: An Application for LPCVD," *IEEE Transactions on Semiconductor Manufacturing*, vol. 3, no. 4, pp. 216-229, November 1990.
- J. Lindhard, "Influence of Crystal Lattice on Motion of Energetic Charged Particles," *Matematisk-Fysiske Meddelelser*, vol. 34, no. 14, 1965.

- P. Lloyd, H.K. Dirks, E.J. Prendergast, and K. Singhal, "Technology CAD for Competitive Products," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. CAD-9, no. 11, pp. 1209-1216, November 1990.
- J. Lorenz, J. Pelka, H. Ryssel, A. Sachs, A. Seidl, and M. Svoboda, "COMPOSITE - A Complete Modeling Program of Silicon Technology," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. CAD-4, no. 4, pp. 421-430, April 1985.
- T.L. Luan, G.S. May, H.-C. Liu, and C.J. Spanos, "Using Equipment Models within a Technology CAD Framework," *Proceedings ICCAD*, 1991.
- T.L. Luan, "Manufacturing-Based IC Process and Device Simulation," ERL Memorandum UCB/ERL M91/55, University of California, May 1991.
- D. MacArthur, "Chemical Etching of Metals," in *Etching for Pattern Definition*, p. 76, Electrochemical Society, 1976.
- S.A. MacDonald, H. Ito, H. Hiraoka, and C.G. Wilson, "A New Oxygen Plasma-Developable UV-Sensitive Resist," *SPE Proceedings, Tech. Conf. Photopolymers - Principles, Processes and Materials*, p. 177ff, 1985.
- C.A. Mack, "PROLITH: A Comprehensive Optical Lithography Model," *Proceedings of SPIE: Optical Microlithography IV*, vol. 538, pp. 207-220, March 1985.
- C.A. Mack, "Understanding Focus Effects in Submicron Optical Lithography," *Proceedings of SPIE: Optical/Laser Microlithography*, vol. 922, pp. 135-148, March 1988.
- V. Mastromarco, A.R. Neureuther, and K.K.H. Toh, "Printability of Defects in Optical Lithography : Polarity and Critical Location Effects," *J. Vac. Sci. Technol. B* 6(1), pp. 224-229, Jan/Feb 1988.
- S. Matsuo, "An Analytical Treatment on the Pattern Formation Process By Sputter Etching," *Japanese Journal of Applied Physics*, vol. 15, no. 7, pp. 1253-1262, July 1976.
- S. Matsuo and Y. Adachi, "Reactive Ion Beam Etching Using a Broad Beam ECR Ion Source," *Japan Journal of Applied Physics*, vol. 21, p. L4, 1982.
- T. Matsuzawa, T. Ito, and H. Sunami, "Three-dimensional Photoresist Image Simulation on Flat Surfaces," *IEEE Transactions on Electron Devices*, vol. ED-32, no. 9, pp. 1781-1783, September 1985.
- T. Matsuzawa, A. Moniwa, N. Hasegawa, and H. Sunami, "Two-Dimensional Simulation of Photolithography on Reflective Stepped Substrate," *IEEE Transactions on Computer Aided Design of Integrated Circuits*, vol. CAD-6, no. 3, pp. 446-451, May 1987.

- J.S. Mayo, "Technology Requirements of the Information Age," *Bell. Lab. Rec.*, vol. 60, p. 55, 1982.
- J. McVittie, J. Rey, L.-Y. Cheng, A. Bariya, S. Ravi, and K. Saraswat, "SPEEDIE: A Profile Simulator for Etching and Deposition," *TECHCON '90, Extended Abstract Volume*, pp. 16-19, Semiconductor Research Corporation, San Jose, California, October 16-18, 1990.
- J.P. McVittie, J.C. Rey, L.Y. Cheng, M.M. IslamRaja, and K.C. Saraswat, "LPCVD Profile Simulation Using a Re-Emission Model," *IEDM Technical Digest*, pp. 917-920, San Francisco, CA, December 1990.
- C.A. Mead and L.A. Conway, *Introduction to VLSI Systems*, Addison-Wesley, Reading, Mass., 1980.
- P. Meakin, P. Ramanial, L.M. Sander, and R.C. Ball, "Ballistic deposition on surfaces," *Physics Review A*, vol. 34, p. 5091, 1986.
- A. Moniwa, T. Matsuzawa, T. Ito, and H. Sunami, "A Three-Dimensional Photoresist Imaging Process Simulator for Strong Standing-Wave Effect Environment," *IEEE Transactions on Computer Aided Design of Integrated Circuits*, vol. CAD-6, no. 3, pp. 431-437, May 1987.
- R.A. Morgan, *Plasma Etching in Semiconductor Fabrication*, Elsevier Science Publishers, B. V., New York, 1985.
- D.J. Barber, F.C. Frank, M. Moss, J.W. Steeds, I.S.T. Tsong, "Prediction of Ion-bombarded Surface Topographies Using Franks's Kinematic Theory of Crystal Dissolution," *Journal of Materials Science*, vol. 8, pp. 1030-1040, 1973.
- K.P. Mueller and J. Pelka, "Redeposition in ion milling," *Microelectronic Engineering*, vol. 7, pp. 91-101, North-Holland, 1987.
- C. Murray, "Wet Etching Update," *Semiconductor International*, pp. 80-85, May 1986.
- Y. Nakagome, H. Tanaka, K. Takeuchi, E. Kume, Y. Watanbe, T. Kaga, Y. Kawamoto, F. Murai, R. Izawa, D. Hisamoto, T. Kisu, T. Nishida, E. Takeda, and K. Itoh, "An Experimental 1.5-V 64-Mb DRAM," *IEEE Journal of Solid-State Circuits*, vol. 26, no. 4, pp. 465-472, April 1991.
- N.G. Nakhodkin and A.I. Shaldervan, "Effect of vapour incidence angles on profile and properties of condensed films," *Thin Solid Films*, vol. 10, p. 109, 1972.
- M.A. Narasimham and F.H. Dill, *Projection Printer Photolithographic Images in Positive Photoresists Under Polychromatic Exposure*. Presented at the SPSE 30th Annual Conference, North Hollywood, CA, May 1977.

- M.A. Narasimham and J.H. Carter, "Effects of Defocus on Photolithographic Images Made With Projection Printing Systems," *Proceedings of SPIE: Semiconductor Microlithography III*, 1978.
- S. Nassif, A.J. Strojwas, and S.W. Director, "FABRICS II: A Statistically Based IC Fabrication Process Simulator," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. CAD-3, pp. 20-46, January 1984.
- A.R. Neureuther, C.Y. Liu, and C.H. Ting, "Modeling Ion Milling," *Journal of Vacuum Science Technology*, vol. 16, no. 6, pp. 1767-1771, November/December 1979.
- A.R. Neureuther, C.H. Ting, and C.-Y. Liu, "Application of Line-Edge Profile Simulation to Thin-Film Deposition Processes," *IEEE Transactions on Electron Devices*, vol. ED-27, no. 8, pp. 1449-1455, August 1980.
- A.R. Neureuther, "IC Process Modeling and Topography Design," *IEEE Proceedings*, vol. 71, no. 1, pp. 121-128, January 1983.
- A.R. Neureuther and W.G. Oldham, "Simulation of Optical Lithography," in *Advances in CAD for VLSI Process and Devices Simulation*, ed. W. L. Engl, Elsevier Science Publishers, B.V., Amsterdam, 1985.
- A.R. Neureuther, "Algorithms for Wafer Topography Simulation," in *NASECODE IV: Proceedings of the Fourth International Conference on the Numerical Analysis of Semiconductor Devices and Integrated Circuits*, ed. J. J. H. Miller, pp. 58-69, Boole Press, Trinity College, Dublin, Ireland, June 19-21, 1985.
- A.R. Neureuther, "Basic Models and Algorithms for Wafer Topography Simulation," in *New Problems and New Solutions for Device and Process Modelling*, ed. J.J.H. Miller, pp. 99-109, Boole Press, Trinity College, Dublin, Ireland, June 17-18, 1985.
- A.R. Neureuther, "Topography Simulation Tools," *Solid State Technology*, pp. 71-75, March 1986.
- A.R. Neureuther, P. Flanner III, and S. Shen, "Coherence of Defect Interactions with Features in Optical Imaging," *J. Vac. Sci. Technol. B.*, pp. 308-312, Jan/Feb 1988.
- A.R. Neureuther, W.G. Oldham, B. Huynh, K. Toh, R. Ferguson, W.E. Haller, and D. Sutija, "Test Patterns and Simulation Software for Characterization of Optical Projection Printing," *Interface'88 : Proceedings of KTI Microelectronics Seminar*, pp. 113-122, November 1988.
- A.R. Neureuther, K.K.H. Toh, J.E. Fleishman, D. Yu, G. Misium, B. Huynh, B. Uathavikul, and W.G. Oldham, "Exploratory Test Structures for Image Evaluation in Optical Projection Printing," *Proceedings of SPIE: Optical/Laser Microlithography II*, vol. 1088, pp. 83-91, March 1989.

- M.E. Newell, R.G. Newell, and T.L. Sancha, "A solution to the Hidden Surface Problem," *Proceedings of the ACM National Conference 1972*, pp. 443-450, 1972.
- J.M. Nieuwenhuizen and H.B. Haanstra, "Microfractography of thin films," *Philips Technical Review*, vol. 27, no. 3/4, pp. 87-91, 1966.
- A. Nitayama, T. Sato, K. Hashimoto, F. Shigemitsu, and M. Nakase, "New Phase-Shifting Mask with Self-Aligned Phase-Shifters for a Quarter Micron Photolithography," *Proceedings of IEDM*, pp. 57-60, 1989.
- R.S. Nowicki, "The RF Diode Co-Deposition of Refractory Metal Silicides," *Solid State Technology*, p. 95, November 1980.
- S. Odanaka, H. Umimoto, M. Wakabayashi, and H. Esaki, "SMART-P: Rigorous Three-Dimensional Process Simulator on a Supercomputer," *IEEE Transactions on Computer-Aided Design of Integrated Circuits*, vol. CAD-7, no. 6, pp. 675-683, June 1988.
- H.K. Oertel, M. Weiss, J. Chlebek, H.L. Huber, R. Dammel, C.R. Lindley, J. Lingnau, and J. Theis, "Resist Modeling near Resolution and Sensitivity Limits in X-Ray Lithography," *Proceedings SPIE*, vol. 1089, p. 283ff., 1989.
- W.G. Oldham, S.N. Nandgaonkar, A.R. Neureuther, and M.M. O'Toole, "A General Simulator for VLSI Lithography and Etching Processes: Part I - Application to Projection Lithography," *IEEE Transactions on Electron Devices*, vol. ED-26, no. 4, pp. 717-722, April 1979.
- W.G. Oldham, A.R. Neureuther, C. Sung, J.L. Reynolds, and S.N. Nandgaonkar, "A General Simulator for VLSI Lithography and Etching Processes: Part II - Application to Deposition and Etching," *IEEE Transactions on Electron Devices*, vol. ED-27, no. 8, pp. 1455-1459, August 1980.
- P.-L. Pai and C. Ting, *Proceedings IEEE VLSI Multilevel Interconnection Conference*, pp. 258-354, Santa Clara, California, June 1990.
- Y.H. Park, A.H. Chung, and M.A. Ward, "Step Coverage Evaluation of Copper Films Prepared in Magnetron Sputtering," *Proceedings Eighth International VLSI Multilevel Interconnection Conference*, pp. 295-297, Santa Clara, CA, June 11-12, 1991.
- J. Pelka, K.P. Mueller, and H. Mader, "Simulation of Dry Etch Processes by COMPOSITE," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. CAD-7, pp. 154-159, February 1988.
- J. Pelka, H.-C. Scheer, P. Hoffmann, W. Hoppe, and C. Huth, *Microelectronic Engineering*, vol. 9, p. 491, North-Holland, 1989.

- J. Pelka, M. Weiss, W. Hoppe, and D. Mewes, "Influence of ion scattering on dry etch profiles," *Journal of Vacuum Science and Technology B*, vol. 7, no. 6, pp. 1483-1487, November/December 1989.
- J. Pelka, "SOLID : Comprehensive Three Dimensional Simulation Program for Optical Microlithography," *Information Brochure, Fraunhofer-Institut fur Mikrostrukturtechnik*, May 1990.
- J. Pelka, "Simulation of Ion-Enhanced Dry-etch Processes," *Microelectronic Engineering*, vol. 13, pp. 487-491, 1991.
- B.R. Penumalli, "A Comprehensive Two-Dimensional VLSI Process Simulation Program, BICEPS," *IEEE Transactions on Electron Devices*, vol. ED-30, September 1983.
- A.K. Pfau, E.W. Scheckler, D.M. Newmark, and A.R. Neureuther, "Continuous Slope Phase-Shift Masks," *11th BACUS Symposium on Photomask Technology*, SPIE, Sunnyvale, CA, September 26, 1991.
- Bui-Tuong Phong, "Illumination for Computer Generated Pictures," *Communications of the ACM*, vol. 18, no. 6, pp. 311-317, June 1975.
- W. Pilz, J. Pelka, and P. Banks, "Profile Evolution in the Multilevel Technique," *Microelectronic Engineering*, vol. 11, pp. 521-525, Elsevier Science Publishers B.V., Amsterdam, 1990.
- J.G. Posa, "Memories," *Electronics*, pp. 132-145, October 23, 1980.
- F.P. Preparata and M.I. Shamos, *Computational Geometry: An Introduction*, Springer-Verlag, New York, 1985.
- M.D. Prouty and A.R. Neureuther, "Optical Imaging with Phase-Shift Masks," *Proceedings of SPIE*, vol. 470, pp. 228-232, March 1984.
- J.L. Reynolds, *Characterization of Plasma Etched Structures in IC Processing*, Ph.D. Dissertation, University of California, Berkeley, 1983.
- B. Roland, R. Lombaerts, C. Jakus, and F. Coopmans, "The Mechanism of the DESIRE Process," *Proceedings SPIE*, vol. 771, p. 111f, 1987.
- R. Rubenstein and H. Hersh, *The Human Factor - Designing Computer Systems for People*, Digital Press, Burlington, MA, 1984.
- S.M. Rubin and T. Whitted, "A 3-Dimensional Representation for Fast Rendering of Complex Scenes," *Proceedings of SIGGRAPH '80, in Computer Graphics*, vol. 14, no. 3, pp. 110-116, July 1980.

- W.R. Runyan and K.E. Bean, *Semiconductor Integrated Circuit Processing Technology*, Addison-Wesley, Reading, Massachusetts, 1990.
- H. Sawin, D. Gray, T. Dalton, and J. Arnold, "Mechanisms of Pattern Dependencies in Plasma Etching Processes," *Proceedings of Sematech SCOE Coordination Meeting*, pp. 167-179, Austin, Texas, April 30, 1991 to May 1, 1991 .
- E.W. Scheckler, "Extraction of Topography Dependent Electrical Characteristics from Process Simulation using SIMPL, with Application to Planarization and Dense Interconnect Technologies," ERL Memorandum UCB/ERL M89/27, University of California, March 1989.
- E.W. Scheckler, A.S. Wong, R.H. Wang, G. Chin, J.R. Camagna, K.K.H. Toh, K.H. Tadros, R.A. Ferguson, A.R. Neureuther, and R.W. Dutton, "A Utility-Based Integrated Process Simulation System," *1990 Symposium on VLSI Technology, Digest of Technical Papers*, pp. 97-98, Honolulu, Hawaii, June 4-7, 1990.
- E.W. Scheckler, K.K.H. Toh, D.M. Hoffstetter, and A.R. Neureuther, "3D Lithography, Etching, and Deposition Simulation (SAMPLE-3D)," *1991 Symposium on VLSI Technology, Digest of Technical Papers*, pp. 97-98, Oiso, Japan, May 28-30, 1991.
- E.W. Scheckler, A.S. Wong, R.H. Wang, G. Chin, J.R. Camagna, A.R. Neureuther, and R.W. Dutton, "A Utility-Based Integrated System for Process Simulation," *to appear IEEE Transactions on Computer-Aided Design*, May 1992.
- R.W. Scheifler and J. Gettys, "The X Window System," *ACM Transactions on Graphics*, vol. 5, no. 2, pp. 79-109, April 1986.
- E. van Schie, *Trendy: An Integrated Program for IC Process and Device Simulation*, Ph.D. Dissertation, Universiteit Twente, The Netherlands, 1990.
- B. Schneiderman, *Designing the User Interface: Strategies for Effective Human-Computer Interaction*, Addison-Wesley, Reading, Mass., 1986.
- S. Sechrest, *An Introductory 4.3BSD Interprocess Communication Tutorial*, Computer Science Research Group, EECS Department, University of California, July 1985.
- R. Sedgewick, *Algorithms*, Addison-Wesley, Reading, Massachusetts, 1983.
- S. Selberherr, F. Fasching, C. Fischer, S. Halama, H. Pimingstorfer, H. Read, H. Stippel, P. Verhas, and K. Wimmer, "The Viennese TCAD System," *1991 International Workshop on VLSI Process and Device Modeling*, pp. 32-35, Oiso, Japan, May 1991.
- C.H. Sequin, "Microfabrication on the Macintosh," *USENIX Computer Graphics Workshop*, Monterey, California, November 1989.

- P. Singer, "Techniques of Low Pressure CVD," *Semiconductor International*, p. 72, May 1984.
- J.G. Skinner, "Some Relative Merits of Contact, Near-Contact and Projection Printing," *Proceedings Kodak Interface*, vol. 73, p. 53, 1973.
- R. Smith, G. Carter, and M.J. Nobes, "The theory of surface erosion by ion bombardment," *Proceedings of the Royal Society of London A*, vol. 407, pp. 405-433, 1986.
- R. Smith, S.J. Wilde, G. Carter, I.V. Katardjiev, and M.J. Nobes, "The simulation of two-dimensional surface erosion and deposition processes," *Journal of Vacuum Science and Technology B*, vol. 5, no. 2, pp. 579-585, March/April 1987.
- T. Smy, R.N. Tait, K.L. Westra, and M.J. Brett, "Simulation of Density Variation and Step Coverage for Via Metallization," *Proceedings Sixth International VLSI Multilevel Interconnection Conference*, pp. 292-298, Santa Clara, CA, June 12-13, 1989.
- T. Smy, K.L. Westra, and M.J. Brett, "Simulation of Density Variation and Step Coverage for a Variety of Via/Contact Geometries Using SIMBAD," *IEEE Transactions on Electron Devices*, vol. ED-37, no. 3, pp. 591-598, March 1990.
- T. Smy, R.N. Tait, and M.J. Brett, "Ballistic Deposition Simulation of Via Metallization Using a Quasi-Three-Dimensional Model," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. CAD-10, no. 1, pp. 130-135, January 1991.
- C.A. Spence, S.A. MacDonald, and H. Schlosser, "Silylation of poly (t-BOC) styrene resists: Performance and Mechanisms," *Proceedings SPIE: Advances in Resist Technology and Processing VII*, vol. 1262, pp. 344-357, March 1990.
- A.D.G. Stewart and M.W. Thompson, "Microtopography of Surfaces Eroded by Ion-Bombardment," *Journal of Materials Science*, vol. 4, pp. 56-60, 1969.
- T. Strahl, "Flash Evaporation, An Alternative to Magnetron Sputtering in the Production of High-Quality Aluminum Alloy Films," *Solid State Technology*, pp. 78-82, December 1978.
- P. Sutardja and W.G. Oldham, "Modeling of Stress-Effects in Silicon Oxidation Including Viscosity Oxide," *IEDM Technical Digest*, pp. 264-267, December 1987.
- R.N. Tait, T. Smy, and M.J. Brett, "Simulation and Measurement of Density Variations in Mo films Sputter Deposited Over Oxide Steps," *Journal Vacuum Science and Technology B*.
- R.N. Tait, T. Smy, and M.J. Brett, "A Ballistic Deposition Model for Films Evaporated Over Topography," *Thin Solid Films*, vol. 187, Elsevier/Sequoia, The Netherlands, 1990.



- R.N. Tait, S.K. Dew, T. Smy, and M.J. Brett, "Density Variation of Tungsten Films Sputtered Over Topography," *Journal of Applied Physics (submitted)*, 1991.
- R. Tarascon, E. Reichmanis, F. Houlihan, A. Shugard, and L. Thompson, *Polymer Engineering and Science*, vol. 29, no. 13, pp. 850-858, July 1989.
- G.N. Taylor, M.Y. Hellman, T.M. Wolf, and J.M. Zeigler, *Proceedings SPIE*, vol. 920, p. 290, 1988.
- S. Tazawa, S. Matsuo, and K. Saito, "A General Characterization and Simulation Method for Deposition and etching Technology," *to appear: IEEE Transactions on Semiconductor Manufacturing*.
- S. Tazawa, S. Matsuo, and K. Saito, "Unified Topography Simulator for Complex Reaction Including Both Deposition and Etching," *1989 Symposium on VLSI Technology, Digest of Technical Papers*, pp. 45-46, May 1989.
- T. Terasawa, N. Hasegawa, T. Kurosaki, and T. Tanaka, "0.3-micron Optical Lithography using a Phase-Shifting Mask," *Proceedings of SPIE: Optical/Laser Microlithography II*, vol. 1088, pp. 25-33, 1989.
- P. Thoren, I.W. Rangelow, R. Kassing, and P. Kuechner, "Three-dimensional simulation of sputter deposition processes for sub-um technology," *Microelectronic Engineering*, vol. 8, North-Holland, 1988.
- T. Thurgate, "Segment Based Etch Algorithm and Modeling," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. CAD-10, no. 9, pp. 1101-1109, September 1991.
- K.K.H. Toh, G. Dao, R. Singh, and H. Gaw, *Chromeless Phase-Shifted Masks : A New Approach to Phase-Shifting Masks*. Presented at the BACUS 10th Annual Symposium on Microlithography, September 1990.
- K.K.H. Toh and A.R. Neureuther, "Identifying and Monitoring Effects of Lens Aberrations in Projection Printing," *Proceedings of SPIE: Optical Microlithography VI*, vol. 772, pp. 202-209, March 1987.
- K.K.H. Toh, C.C. Fu, K.L. Zollinger, A.R. Neureuther, and R.F.W. Pease, "Characterization of Voting Suppression of Optical Defects Through Simulation," *Proceedings of SPIE: Optical/Laser Microlithography*, vol. 922, pp. 194-202, March 1988.
- K.K.H. Toh, "Two-Dimensional Images with Effects of Lens Aberrations in Optical Lithography," *M.S. Thesis, Memorandum No. UCB/ERL M88/30*, University of California, Berkeley, May 20, 1988.

- K.K.H. Toh and A.R. Neureuther, "Mapping Image Quality in Optical Lithography with Visual Test Patterns," *Interface'89 : Proceedings of KTI Microelectronics Seminar*, pp. 155-177, November 1989.
- K.K.H. Toh, "Algorithms for Three-Dimensional Simulation of Photoresist Development," Memo. No. UCB/ERL M90/123, Ph.D. Dissertation, University of California, Berkeley, December 14, 1990.
- K.K.H. Toh and A.R. Neureuther, "Three-Dimensional Simulation of Optical Lithography," *Proceedings of SPIE: Optical/Laser Microlithography IV*, March 1991.
- P. Tong and J.N. Rossettos, *Finite-Element Method: Basic Technique and Implementation*, MIT Press, Cambridge, Massachusetts, 1977.
- J.I. Ullacia-Fresnedo, "Theoretical and Experimental Considerations Necessary to Build a Dry-etching Process Simulator," Technical report No. G833-1, Integrated Circuits Laboratory, Stanford University, 1988.
- J.I. Ullacia-Fresnedo and J.P. McVittie, "etching using Monte Carlo techniques," *Journal of Applied Physics*, vol. 65, no. (4), pp. 1484-1491, February 15, 1989.
- H. Uemimoto, S. Odanaka, I. Nakao, and H. Esaki, "Numerical Modeling of Nonplanar Oxidation Coupled with Stress Effects," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. CAD-8, no. 6, pp. 599-607, June 1989.
- H. Uemimoto and S. Odanaka, "Three-Dimensional Numerical Simulation of Local Oxidation of Silicon," *IEEE Transactions on Electron Devices*, vol. ED-38, no. 3, pp. 505-511, March 1991.
- H.P. Urbach and D.A. Bernard, "Modeling Latent Image Formation in Photolithography using the Helmholtz Equation," *Proceedings of SPIE: Optical/Laser Microlithography III*, vol. 1264, pp. 278-293, March 1990.
- T. Ushirogouchi, Y. Onishi, and T. Tada, "Resist profile simulation for photoresist composition optimization," *Journal of Vacuum Science Technology B*, vol. 8, no. 6, pp. 1418-1422, November/December 1990.
- R.-J. Visser, J.P.W. Schellekens, M.-E. Reuhman-Hisken, and L.J. van IJzendoorn, "Mechanisms and Kinetics of Silylation of Resist Layers from the Gas Phase," *Proceedings SPIE*, vol. 771, p. 111ff, 1987.
- J.L. Vossen and J.J. Cuomo, "Glow Discharge Sputter Deposition," in *Thin Film Processes*, ed. J.L. Vossen and W. Kern, pp. Chapter II-1, Academic Press, New York, 1978.
- E.J. Walker, "Reduction of Photoresist Standing-Wave Effects by Post-Exposure Bake," *IEEE Transactions on Electron Devices*, vol. ED-22, no. 7, pp. 464-466, July 1975.

- R.H. Wang, "The Berkeley Topography Utilities," M.S. Project Report, University of California, August 1991.
- F. Watanabe and Y. Ohnishi, *Journal of Vacuum Science and Technology B*, vol. 4, no. 1, p. 422, 1986.
- G.F. Watson, "Solid State," *IEEE Spectrum*, pp. 52-53, January 1991.
- G.F. Watson, "The Main Event," *IEEE Spectrum*, p. 30, January 1991.
- K.L. Westra, T. Smy, and M.J. Brett, *IEEE Electron Device Letters*, vol. EDL-10, p. 198, 1989.
- R.H. Wilson, R.W. Stoll, and M.A. Calacone, *Tungsten and Other Refractory Metals for VLSI Applications*, 1, p. 35, 1986.
- F.W. Wise, "Lens Aberrations and Nonuniform Illumination in Projection Lithography," M.S. Thesis, University of California, Berkeley, December 1981.
- R.N. Wolf, M.A. Wesley, J.C. Kyle Jr., F. Gracer, and W.J. Fitzgerald, "Solid Modeling for Production Design," *IBM Journal of Research and Development*, vol. 31, no. 3, pp. 277-295, May 1987.
- S. Wolf and R.N. Tauber, *Silicon Processing for the VLSI Era*, 1 - Process Technology, Lattice Press, Sunset Beach, California, 1986, reprint June, 1987.
- S. Wolf, *Silicon Processing for the VLSI Era*, 2 - Process Integration, Chapter 9, Lattice Press, Sunset Beach, California, 1990.
- S. Wolfram, *Mathematica, A System for Doing Mathematics by Computer*, Addison-Wesley, Redwood City, California, 1988.
- A. Wong, W. Dietrich, and M. Karasick, "The SWR Architecture Document Version 0.2," CAD Framework Initiative #162, Austin, Texas, March 1991.
- A.S. Wong, "An Integrated Graphical Environment for Operating IC Process Simulators," ERL Memorandum M89/67, University of California, May 1989.
- A.S. Wong and A.R. Neureuther, "The Intertool Profile Interchange Format: A Technology CAD Environment Approach," *IEEE Transactions on Computer-Aided Design*, vol. CAD-10, no. 9, pp. 1157-1162, September 1991.
- H.C. Wu, A.S. Wong, Y.L. Koh, E.W. Scheckler, and A.R. Neureuther, "Simulated Profiles from the Layout, Design Interface in X (SIMPL-DIX)," *IEDM Technical Digest*, San Francisco, CA, December 1988.

- H.C. Wu, "SIMPL-DIX (Simulated Profiles from the Layout - Design Interface in X),," ERL Memorandum UCB/ERL M88/13, University of California, January 1988.
- S. Yamaguchi, "Three-Dimensional Simulation Technology," *Semiconductor World (In Japanese)*, pp. 149-153, January 1990.
- S. Yamamoto, T. Kume, M. Ohgo, T. Matsuzawa, S. Tachi, and H. Sunami, "A Two-Dimensional Etching Profile Simulator: ESPRIT," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. CAD-6, no. 3, pp. 417-421, March 1987.
- A. Yamano, K. Harafuji, M. Kubota, and N. Nomura, "Simulation Study of Ion Energy Distribution in a Plasma Sheath Including Charging Effects," *submitted for publication*, May 1991.
- N. Yamauchi, T. Yashi, and T. Wada, "A pattern edge profile simulation for oblique ion milling," *Journal of Vacuum Science Technology A*, vol. 2, no. 4, pp. 1552-1557, October-December 1984.
- M.S. Yeung, "Modeling High Numerical Aperture Optical Lithography," *Proceedings of SPIE: Optical/Laser Microlithography*, vol. 922, pp. 149-167, March 1988.
- O.C. Zienkiewicz and R.L Taylor, "The Finite Element Method," 3rd edition, McGraw-Hill, New York, 1983.

## **APPENDIX**

### **USER GUIDES**

**crater** - a cell removal algorithm for 3D etching of resists.  
**netch** - a new etch program, the basic element of SAMPLE-3D.  
**sample3D-gui** - a graphical user interface for SAMPLE-3D  
**ptom** - for converting pdraw format data files to Mathematica  
**cutaway** - for making cutaway views of pdraw format data files  
**pdraw update** - the latest version of pdraw  
**contour update** - the latest version of contour

## NAME

**crater** - Cell-Removal Algorithm for 3D Etching of Resists

## SYNOPSIS

**crater** [-bf *binary rate file*] [-af *ascii rate file*] [-et *etch time*] [-pt *plot time*] [-if *instruction file*] [-s] [-[no]pr] [-[no]h] [-noplot]

## DESCRIPTION

**CRATER** is a 3D cell-based volume removal program for simulation of photoresist dissolution. It is compact efficient and sufficiently accurate for useful simulation. The program reads a dissolution rate file describing the etch rates in a 3D simulation region. If the -et option is not specified, the program then reads user commands with an interactive command line interpreter or from an external instruction file. After simulating the resist development for a given time, the program provides 2D and 3D plot files in a convenient format for external plotting. The program can also make system calls to the **drawplot** and **pdraw** X-Window plotters.

The rate information may be supplied in either binary or ascii format. The rate file is arranged as follows:

## Rate File Format

```
xmin xmax ymin ymax zmin zmax
nx
ny
nz
rate(0,0,0)
rate(1,0,0)
rate(2,0,0)
..
rate(0,1,0)
..
..
```

In the above, *xmin*, *xmax*, *ymin*, *ymax*, *zmin*, and *zmax*, are the bounds of the simulation region in micrometers. Typically *zmin*=0.0, and *zmax* is the resist thickness. *nx*, *ny*, and *nz* give the number of elements in the x,y, and z direction. *rate(i,j,k)* is the etch rate in microns/sec at the inside corner of the cell (*i,j,k*). The etch rate at any point is determined by interpolating the eight rates in the cell of interest. The rate at (*nx,ny,nz*) is the rate at the location (*xmax,ymax,zmax*). There are actually *nx*-1 cells for *nx* elements. The cells for the rate information are dynamically allocated in the program and are independent of the cells used for the dissolution simulation.

Instructions are run in the order which they are entered. A typical simulation session begins by entering commands to initialize the simulation region and rates. Then the etch model, etch time and time step information is entered. The command "runcells" executes the simulation. Finally, plots of the result may be made.

## OPTIONS

- bf *binary-ratefile***  
reads in a binary format dissolution rate file created by an external program (cf. **bleach**).
- af *ascii-ratefile***  
reads in an ascii format dissolution rate file created by an external program (cf. **bleach**).
- et *etchtime***  
sets the etch time in seconds, and causes the program to commence execution.
- pt *plotime***  
sets the time interval in seconds between plots. if this option is specified, plotting will be automatic.
- if *instruction\_file***  
reads in an external file containing simulator instructions. If the last line of the instruction file does not terminate the program, additional commands may be entered via the terminal command interpreter.
- s** suppresses most output to the screen, and calls to external plotting programs.
- [no]pr**  
forces or suppresses the creation of PostScript files by external plotting programs **drawplot**, and **pdraw**.
- [no]h**  
forces or suppresses the invocation of hidden line removal by the external 3D plotter **pdraw**.
- noplot**  
suppresses plotting on the screen. Plotfiles are still created.

## SIMULATOR COMMANDS

Command lines begin with a keyword, continue with a list of variables, and end with a semicolon ";". A pound sign "#" may be used to force the program to ignore everything up to the next semicolon.

*initialization commands:*

- sim\_region *xmin xmax ymin ymax zmin***  
Forces the simulation region to conform to the given parameters.
- analytic\_rate *rate\_type***  
Several analytic etch rate functions are encoded in the program. They are not intended for general use.
- celldensity *nx ny nz***  
Specifies the number of cells in the x,y and z directions. The dimensions of the simulation region must be already entered in order to calculate the cell size.
- etchlayers *numlay thick0 rate0 (drate1)* .**  
Specifies etch rates in  $\mu\text{m}/\text{sec}$  for several layers. The layers begin from the bottom of the simulation region and each have a thickness specified in microns. If the etch model is directional both isotropic and directional rates for each layer must be given.
- showsets**  
Shows the current simulation settings.

**loadrate ftype [filename]**

Causes the program to read an external rate file. If ftype is 0 the file is binary, if it is 1 the file is ascii. The file name may be given. The default file is "rval.3D.binary".

**etch commands:****etchmodel mtype [parameter list]**

Several models may be entered. The parameters that must be included depend on the model type mtype. Isotropic etching is mtype=0. Simple directional etching is mtype=1 and requires the etch source angle phi and theta. Silylation etching is mtype=2 and requires the etch source phi and theta, the silylation rate model parameters p1,p2,r1, and r2, and the isotropic etch rate. In this model, the rate matrix is assumed to hold the normalized concentration of Si in the resist.

**etchtime time [dt]**

The etch time and time step are given in seconds. If dt is not given it will be determined automatically by the program. Successful simulation will result if the time step is small enough so that at least 4 time steps are needed to etch one cell.

**ratefact value**

The rate matrix value will be scaled by the rate factor given here. The default is value=1.0.

**mask commands:****clearmask**

clears any mask information.

**maskpiece type coords**

Mask features may be added to the internal mask data structure with this command. Mask type 0 indicates the mask feature is rectangular with position (x1,y1), width x1, and height y1. Mask type 1 indicates a triangular mask element with vertexes at (x1,y1),(x2,y2), and (x3,y3).

**setmask type**

This sets the mask entered with previous maskpiece commands. If the type is 0 the mask features are clear and the background is opaque. If type=1 the mask features are opaque and the background is open.

**execution and plotting commands:****runcells [excessflag]**

Runs the simulation. If excessflag=0 no excess overflow calculation is made. If excessflag=1, then the overflow calculation is made. Without the overflow calculation, the time step must be very small. The default is excessflag=1.

**plotcells ptype [index] [filename]**

Creates a plot file and invokes an external plot program. If ptype=0, a slice along a constant x plane is made with the particular plane equal to x-array index index. Similar plots are made with ptype=1 for constant y, and ptype=2 for constant z. Giving ptype=3 creates a 3D plot file. The plot file name may be given, otherwise the file will be written to the default plotfile "2D.cells" or "3D.cells".

**miscellaneous commands:****system commands**



runs a unix command via a system call. This is convenient for manipulating or compressing files during a long batch run.

**savestruct,loadstruct**  
not yet implemented

**end,quit,exit**  
terminates the program.

## **AUTHOR**

Edward W. Scheckler

## **FILES**

*rval.3D.binary, 3D.cells, 2D.cells*

## **BUGS**

Basically, crater is a very robust program. The maximum problem size is installation dependent, since some of the internal matrices are statically allocated. Some default values may not be initialized properly, so it is best to set them explicitly with the command interpreter. There also needs to be more work to investigate the overall accuracy of cell-based development simulation as compared to other methods.

## **SEE ALSO**

**splat(L), bleach(L), etch(L), drawplot(L), pdraw(L)**

## NAME

**netch** - a new 3D etch and deposition simulator for SAMPLE-3D

## SYNOPSIS

**netch** [-b *binary\_ratefile*] [-af *ascii\_ratefile*] [-if *instruction\_file*] [-ip] [-in] [-mf *mask\_file*] [-et *etch-time*] [-pt *plottime*] [-s] [-[no]v] [-[no]pr] [-[no]h] [-noplot] [-[no]arc] [-[no]c]

## DESCRIPTION

**NETCH** is a major extension to the etch program in **SAMPLE-3D**, allowing 3D simulation of wet and dry etching, ion-milling, and deposition in integrated circuit processing. The program has several command line options for compatibility with the earlier lithography dissolution program **ETCH**. Generally, however, **NETCH** interprets commands line by line from a command file or from standard input.

The program can read inhomogeneous material rate information from a file in either binary or ascii format. It produces output in a format convenient for external plotting and can also make Unix system calls to the X-window plot programs **drawplot** and **pdraw**.

External rate information may be supplied in either binary or ascii format. The rate file is arranged as follows:

## Rate File Format

```
xmin xmax ymin ymax zmin zmax
nx
ny
nz
rate(0,0,0)
rate(1,0,0)
rate(2,0,0)
..
rate(0,1,0)
..
..
```

In the above, *xmin*, *xmax*, *ymin*, *ymax*, *zmin*, and *zmax*, are the bounds of the simulation region in micrometers. Typically *zmin*=0.0, and *zmax* is the resist thickness. *nx*, *ny*, and *nz* give the number of elements in the x,y, and z direction. *rate(i,j,k)* is the etch rate in microns/sec at the inside corner of the cell (*i,j,k*). The etch rate at any point is determined by interpolating the eight rates in the cell of interest. The rate at (*nx,ny,nz*) is the rate at the location (*xmax,ymax,zmax*).

Instructions are run in the order which they are entered. A typical simulation session begins by entering commands to initialize the simulation region and rates. Then the etch model, etch time and time step information are entered. The command "run" executes the simulation. Finally, plots of the result may be made.

## OPTIONS

- bf *binary\_ratefile***  
reads in a binary format dissolution rate file created by an external program (cf. **bleach**).
- af *ascii\_ratefile***  
reads in an ascii format dissolution rate file created by an external program (cf. **bleach**).
- if *instruction\_file***  
reads in an external instruction file and executes each line in sequence. See next section for commands. If the last command in the command file does not terminate the program, additional commands may be entered via the terminal command interpreter.
- ip** invokes the command interpreter. A prompt appears, and waits for a command. Commands terminate with a semicolon (;). See next section.
- in** runs a rudimentary interactive command menu. Many options are not accesible with this command.
- mf** reads in an external mask file. See next section for file format.
- et *etchtime***  
sets the default etch time in seconds
- pt *plottime***  
sets the default time interval in seconds between plots.
- s** suppresses most output to the screen, and calls to external plotting programs. This option is used when the program is invoked by an external X-window user-interface.
- [no]v**  
forces or suppresses verbose mode.
- [no]pr**  
forces or supresses the creation of PostScript files by external plotting programs **drawplot**, and **pdraw**.
- [no]h**  
forces or supresses the invocation of hidden line removal by the external 3D plotter **pdraw**.
- noplot**  
supresses plotting on the screen. Plotfiles are still created.
- [n]arc**  
forces or supresses arc interpolation used in the mesh modification algorithm.

## SIMULATOR COMMANDS

Command lines begin with a keyword, continue with a list of variables, and end with a semicolon (;). A pound sign (#) may be used to force the program to ignore everything up to the next semicolon.

*initialization commands:*

**sim\_region** *xmin xmax ymin ymax zmin*

Forces the simulation region to conform to the given parameters. Parameter *zsurf* gives the initial surface *z* value.

**adv\_method** *flag*

sets the surface advancement method. Setting *flag=0* forces the original ray-trace algorithm, *flag=1* forces a stramlined ray-trace algorithm, *flag=1* sets a facet motion algorithm (with a fixed time step) and *flag=3* sets the internal advancement method flag to FACET\_ADAPTIVE which sets the time step automatically at each iteration.

**meshdensity *N***

specifies the initial number of nodes per micron on the surface edge. The total number of nodes for a square surface is  $N \times N + (N-1) \times (N-1)$ .

**advance\_accur *flag***

if *flag* = 1, sets an internal mesh advancement accuracy flag to HIGH to better estimate the distance a point travels in an inhomogeneous rate field. If *lag* = 0, this accuracy is set to LOW, which is sufficient for nearly all simulations.

**modify\_method *flag***

if *flag* = 1, sets segment based mesh refinement. If *lag* = 0, sets triangle based mesh refinement (default).

**adaptive\_grid *flag***

if *flag* = 1, allows a finer mesh in regions where etch rate varies rapidly. If *lag* = 0, mesh is refined based on segment length only (default).

**cell\_update *flag***

if *flag* = 1 the surface/cell hybrid data structure is updated as the surface passes through it. This is a necessary command for efficient visibility calculations. If *flag* = 0, the surface/cell hybrid data structure is not used.

**mesh\_accur adv mod grid deloop**

is a command for setting the previous four flags. The *deloop* flag is slightly different: if *deloop* = 1, a deloop operation will be performed. If *deloop* = 2, the surface/cell hybrid data structure will be set but no deloop will be performed.

**noclip *flag***

if *flag* = 1, mesh clipping is supressed, otherwise clipping is allowed (default).

**deloop method**

turns the internal deloop flag on and sets the deloop method. This has not been fully implemented as of 10/29/91.

**arc** same as the sommand line option -arc.

**rockbottom *flag***

if *flag* = 1, anything below *zmin* is assumed to have an etch rate of 0.0. If *flag* = 0, the etch rate at any point below *zmin* is assumed to be that of the same (x,y) in the lowest layer.

**celldensity *nx ny nz***

Specifies the number of cells in the x,y and z directions used in the surface/cell hybrid. The dimensions of the simulation region must have been already entered in order to calculate the cell size.

**hemisize *nphi ntheta***

sets the number of elements in the hemisphere above the wafer used for visibility calculations. The maximum values are installation dependent. *nphi* = 45, and *ntheta* = 10 are sufficient for most simulations.

**mask commands:****clearmask**

clears any mask information.

**maskpiece type coords**

Mask features may be added to the internal mask data structure with this command. Mask type 0 indicates the mask feature is rectangular with position (x1,y1), width x1, and height y1. Mask type 1 indicates a triangular mask element with vertexes at (x1,y1), (x2,y2), and (x3,y3).

**setmask type**

This sets the mask entered with previous maskpiece commands. If the type is 0 the mask features are clear and the background is opaque. If type=1 the mask features are opaque and the background is open.

**structure manipulation commands:****transferpattern shiftvalue**

takes the current pattern and shifts it up by *shiftvalue*. The profile may now be used as an etch mask for pattern transfer simulation.

**setnewlayer**

creates a new surface layer which is the exact copy of the current surface layer. This is a necessary step before performing deposition simulation. It can also be used to store intermediate layers.

**stripresist**

not yet implemented.

**clipnow x1 y1 x2 y2**

clips away everything from the current surface outside of the rectangle with corners (x1,y1), (x2,y2).

**etch commands:****etchmodel mtype [parameter list]**

Several models may be entered. The parameters that must be included depend on the model type mtype. Types and parameter lists for the various models are given below:

**isotropic etching: 0**

etch rates are entered with the *etchlayers* command

**directional etching: 1 phi theta**

etch rates are entered with the *etchlayers* command. The etch source angles *phi* (from z-axis) and *theta* (from x-axis in x-y plane) are in degrees.

**plasma etching: 2 phi theta ionflux ionsigma neutral\_m**

etch rates are entered with the *etchlayer* command. The etch source angles *phi* (from z-axis) and *theta* (from x-axis in x-y plane) are in degrees. The source ion flux is given in mA/cm<sup>2</sup>. The standard deviation *sigma* on the ion spread is given in degrees, for a beam assumed to be a gaussian with dependence on *phi*. The *neutral\_m* gives the power of the neutral particle flux cosine: flux = cos<sup>m</sup>(*phi*). Additional plasma etch effects are set with the *passivation*, *damage*, *charging*, and *ionmill* commands.

**crystal etching: 3**

rates are specified along 6 key directions with the *crystalrates* command.

**ionmilling: 4 *phi theta ionflux***

etch rates are entered with the *etchlayer* command. The etch source angles *phi* (from z-axis) and *theta* (from x-axis in x-y plane) are in degrees. The source ion flux is given in mA/cm<sup>2</sup>. This assumes a very narrow ion beam. The sputter yield is set with the *ionmill* command.

**silylation: 5 *phi theta p1 p2 r1 r2 isorate***

sets a simple two piece silylation etch rate model. This command assumes that the etch matrix contains the weight % Si in a resist. For Si weight % = 0, the etch rate is *r1*. At Si weight % = *p1*, the etch rate is *r2*. For Si weight % greater than *p2*, the etch rate is 0. Rates are given in  $\mu\text{m/s}$ . Intermediate values are linearly interpolated.

**loadrate *ftype [filename]***

Causes the program to read an external rate file. If *ftype* is 0 the file is binary, if it is 1 the file is ascii. The file name may be given. The default file is "rval.3D.binary". The rate file contains the simulation region and overrides any previous internal simulation region.

**etchlayers *numlay thick0 rate0 (drate1)***

Specifies etch rates in  $\mu\text{m/sec}$  for several layers. The layers begin from the bottom of the simulation region and each have a thickness specified in microns. If the etch model is directional both isotropic and directional rates for each layer must be given. This command is only used with etch model 0 or 1.

**etchnumlay *nlayers***

sets the number of layers using the *etchlayer* command for use with etch model 2 or higher.

**etchlayer *numlay thick ris rdn rii***

used with *etchnumlay*. Layers begin with *numlay* = 0 measured from the bottom of the simulation region up. The layer thickness *thick* is given in micrometers. The rate value *ris* is the isotropic etch rate in  $\mu\text{m/s}$ , *rdn* is the etch rate due to directly incident neutral particles in  $\mu\text{m/s}$ , and *rii* is the rate due to indirectly incident (reflected) particles in  $\mu\text{m/s}$ .

**passivation *numlay rndep alpha lambda***

sets passivation parameters for layer *numlay*. The redeposition rate out of the plasma is *rndep*, the ion to neutral flux ratio scale factor is *alpha*. The characteristic passivating layer thickness is *lambda*. See section 6.3.1 of E.W. Scheckler's Ph.D thesis for details.

**damage *numlay drate***

sets the etch rate enhancement due to ion-bombardment induced damage in plasma etching for layer *numlay*. The parameter *drate* is the etch rate on a flat wafer with damage enhanced etching is  $\text{rate} = \text{rdn}[1 + \text{drate}]$ .

**charging**

not implemented

**silylation *numlay p1 p2 r1 r2***

allows silylation parameters to be set for etch model 2. The parameters have the same meaning as the parameters in etch model 5.

**ionmill *numlay s0 a1 a2 a3***

sets the sputter yield for layer *numlay*. The etch rate on a flat wafer is  $(s0 * \text{ionflux})/\text{dens}$ . The sputter yield curve is given by powers of cosines:  $S(\phi) =$

$a1\cos(\phi_i) + a2\cos^2(\phi_i) + a3\cos^4(\phi_i)$ . The peak sputter yield angle (in degrees) and ratio of maximum sputter yield to the value at  $\phi_i=0$  should also be given.

**crystalrates numlay thick r001 r011 r111**

sets the thickness (in  $\mu\text{m}$ ) and etch rates along the listed directions (in  $\mu\text{m/s}$ ) for layer numlay.

**densitymodel flag**

if *flag* = 1, sets a density model, else turns it off (default). The density model sets etch and deposition rates inversely proportional to the local material density. In deposition, a record of the density of deposited material is maintained.

**rotatesource phiinc thetainc**

allows simulation of ion milling with a rotating wafer. The parameters *phiinc* and *thetainc* are the angle rotation rates in degrees per second.

**etchtime time [dt]**

The etch time and time step are given in seconds. If *dt* is not given it will be determined automatically by the program. Successful simulation will result if the time step is small enough so that at least 4 time steps are needed to etch one cell.

**deposition commands:**

**depomodel mtype rate parameter-list**

Several models may be entered. The parameters that must be included depend on the model type *mtype*. The rate is the deposition rate on a flat wafer (regardless of source orientation) in  $\mu\text{m/s}$  and must be a negative number for deposition. Types and parameter lists for the various models are given below:

**unidirectional: 1 rate phi theta**

the etch source direction is given in degrees for evaporation from one direction.

**dual: 2 rate phi1 theta1 phi2 theta2**

two different etch sources may be specified for dual source evaporation.

**hemispherical: 3 rate n A phi theta**

sets parameters for sputter distribution. The flux distribution is given by  $\text{flux} = \cos^2(A \times \phi_i)$  and is 0 for  $A \times \phi_i > 90^\circ$ . The angles *phi* and *theta* are the wafer tilt in degrees.

**deposurfmigration sigma**

invokes surface migration by surface convolution. The standard deviation in  $\mu\text{m}$  of the random walk is given by *sigma*.

**depocolumn mtype [A B C]**

sets modified-tangent rule column growth for evaporation. If *mtype* = 1, the *A,B,C* parameters are read, and applied to alter column orientation based on arrival flux with respect to surface orientation. See chapter 7 of E.W.Scheckler's Ph.D. dissertation.

**deptime time val**

sets the deposition time. If the advance method is FACET\_ADAPTIVE, then *val* is the adaptive factor. Otherwise, it is the time step in seconds.

**execution and plotting commands:**

**run** runs the surface advancement simulation.

**runcells [excessflag]**

Runs the simulation. If `excessflag=0` no excess overflow calculation is made. If `excessflag=1`, then the overflow calculation is made. Without the overflow calculation, the time step must be very small. The default is `excessflag=1`.

**plot2D *x1 y1 x2 y2 [filename]***

locates the profile of the surface along the outline from (*x1,y1*) to (*x2,y2*) and saves the result in *filename*. Invokes `drawplot` is possible. The default file is "2Dslices.f77"

**plot3D *[filename]***

saves 3D plot data of the current surface in *filename* and invokes `pdraw` if possible. The default file is "surface.plot"

**plotdensity *pctype index [filename]***

creates a file of density values along a given plane. If *pctype*=0, a slice along a constant x plane is made with the particular plane equal to x-array index *index*. Similar plots are made with *pctype*=1 for constant y, and *pctype*=2 for constant z. The results may be plotted with the contour program `greyscale` option. The default file is "2Ddens.cont".

**plotshadow *flag***

creates a file called "shadow.int" which can be read by certain versions of `pdraw`. If the flag is 0, every shadowed triangle is set dark. If the flag is 1, every triangle is shaded according to the average visibility at its 3 vertexes.

**plotcells *pctype [index] [filename]***

If *pctype*=0, a slice along a constant x plane is made with the particular plane equal to x-array index *index*. Similar plots are made with *pctype*=1 for constant y, and *pctype*=2 for constant z. Giving *pctype*=3 creates a 3D plot file. The plot file name may be given, otherwise the file will be written to the default plotfile "2D.cells" or "3D.cells".

**miscellaneous commands:**

**meshcheck**

checks the mesh data structure self-consistency.

**system commands**

runs a unix command via a system call. This is convenient for manipulating or compressing files during a long batch run.

**savestruct,loadstruct**

not yet implemented

**end,quit,exit**

terminates the program.

**AUTHOR**

Edward W. Scheckler

**FILES**

*rval.3D.binary, 3D.cells, 2D.cells, surface.plot, 2Dslices.f77, 2Ddens.cont,*



**BUGS**

Lots of bugs - let the user beware! Best results occur for evaporation, sputtering, and plasma etching. Deloop has some problems, and has not been fully implemented. Isotropic etching with a mask is also error prone.

**SEE ALSO**

splat(L), bleach(L), etch(L), drawplot(L), pdraw(L), contour(L)

## NAME

sample3D-gui v1.1

## SYNOPSIS

sample3D-gui [ options ] [ CIF files ]

## DESCRIPTION

Sample3D-gui is an X -window based interactive design interface for running 3D topography simulation. Currently, **SPLAT** is invoked for aerial image simulation, **BLEACH** for photoresist exposure, and **ETCH & NETCH** for 3D development, etching and deposition simulation.

sample3D-gui has a number of internal tools to assist the designer in running process simulation. Several 2D and 3D plotting capabilities are available. A pattern editor is included which allows the user to add or delete patterns, to modify formats of patterns, and to update pattern specifications. **HUNCH** is implemented to allow the designer to use operations between masks or sets of masks to highlight locations where topographical problems are anticipated. **WORST** is also included for layout worst-case misalignment.

Version v1.1 is similar to the lithography simulation version v1.1, but allows etching and deposition simulation as well. Version v1.1 also supports IPC communication with **NETCH** so that the internal 3D data structure may be kept active while other menu options are chosen.

## OPTIONS

When **sample3D-gui** is invoked, optional command flags can be used to reset the default parameters. These options also override those set in the ".Xdefaults" file (see the X DEFAULTS section). To restore the default value, a flag can begin with a "+" instead of a "-".

The available options are:

- ar** Enable the auto-raise mode, which automatically raises the **sample3D-gui** window when the mouse cursor enters it.
- b** Enable the bell mode, which signals when an error occurs.
- bd color** On color displays, determine the border color of the window. The default is *black*.
- bg color** On color displays, determine the background color of the window. The default is *white*.
- bw pixels** Set the width of the window border in pixels. The default is 2.
- fg color** On color displays, determine the window foreground (text) color. The default is *black*.
- fm font** Specify the font to be used for the command menu. The default is *vtbold*. The list of available fonts can be found in the directory "/usr/new/lib/X/font".
- fn font** Specify the font for the standard text display. The default is *vtsingle*.

- ft font** Specify the font to be used for the title bar. The default is *vtwidth*.
- hl color** On color displays, determine the color to be used for highlighting. The default is *black*.
- ms color** On color displays, determine the color of the mouse cursor. The default is *black*.
- n name** Specify the name of the window to be used by the window manager. This *name* is also displayed in the title bar.
- pf file** Specify the pattern file to be loaded, rather than the default *sample3D\_pattern*.
- r** Reverse definitions of the foreground and background colors.
- rv** Same as **-r**.
- sf file** Specify that a SPLAT format input file is to be loaded as the layout. Only SPLAT boxes (trial 7) are recognized.
- tb** Enable the title bar with the window name being displayed.
- = geometry** The *sample3D-gui* window is created with the specified size and location determined by the supplied *geometry* specification. See X(1) for details of this specification.
- host:** Normally, *sample3D-gui* gets the host and display number to use from the environment variable "DISPLAY". One can, however, specify them explicitly. The *host* specifies which machine to create the *sampel3D-gui* window on, and the *display* argument specifies the display number. Either value can be defaulted by omission, but ":" is necessary to specify one or both.

## MENU COMMANDS

A menu command is invoked by clicking the mouse at the appropriate command box in the menu. To escape from a command, click at the selected command box again. The menu is set up with a hierarchy of commands, the commands are :

### AERIAL IMAGE

prompts the user for mask information then displays a form for optical stepper parameters. Operation proceeds by calling *SPLAT* for aerial image simulation, and plots the image intensity contours superimposed on the masks in the layout viewport.

### EXPOSE 3D RESIST

checks for a valid rate file and then presents a form for photoresist exposure information. Operation proceeds by calling *BLEACH* for calculating the etch rates in the photoresist region.

### DEVELOP 3D RESIST

asks for an etch time and then prompts for development simulation with *ETCH* or *CRATER*. When complete, the 3D result is displayed in the lower viewport.

### RUN 3D LITH

prompts for all the mask, aerial image, exposure, and development input parameters and then runs *SPLAT*, *BLEACH*, and *ETCH* in succession. The result is displayed in the lower viewport.

**RUN 2D SAMPLE**

allows *SAMPLE* 1.8a to be run for 2D lithography simulation. It prompts for all necessary information and displays the result in the lower viewport.

**RUN 3D ETCH**

if not already active, initializes an IPC link to *NETCH* and asks for mask information and proceeds to the next menu

**DEVELOP**

makes a system call to *ETCH* or *CRATER* for development simulation.

**ISOTROPIC ETCH**

prompts for parameters to run *NETCH* isotropic etch simulation.

**DIRECTIONAL ETCH**

prompts for parameters to run *NETCH* simple directional etch simulation.

**PLASMA ETCH**

prompts for parameters to run *NETCH* plasma etch simulation simulation. Requires neutral and ion flux distribution parameters, etch rates for various materials, etc.

**ION MILL**

prompts for parameters to run *NETCH* ion milling simulation. Requires ion flux, material density and sputter yield, etc.

**CHANGE SAMPLE-3D DEFAULTS**

prompts for flags to reset *NETCH* defaults.

**QUIT SAMPLE-3D**

exits *NETCH* and closes the IPC socket.

**RUN 3D DEPO**

if not already active, initializes an IPC link to *NETCH* and proceeds to the next menu

**ISOTROPIC DEPOSITION**

prompts for parameters to run *NETCH* isotropic deposition simulation.

**UNIDIR. SOURCE DEPOSITION**

prompts for parameters to run *NETCH* point source evaporation simulation.

**HEMISPHERE SOURCE DEPOSITION**

prompts for parameters to run *NETCH* hemispherical deposition simulation, *e.g.* sputtering.

**PLOTS**

goes to the plotting menu.

**LOOK AT 3D PLOT**

if not already active, loads plot data and proceeds to the next menu:

**LEFT**

moves viewpoint to the left (decrement theta)

**RIGHT**

moves viewpoint to the right (increment theta)

**UP**

moves viewpoint up (decrement phi)

**DOWN**

moves viewpoint down (increment phi)

**TOP VIEW**

move viewpoint to birdseye view

**ORIGINAL VIEW**

return to original viewpoint

**HIDDEN LINE ON/OFF**

perform hidden line sort and show illumination as grey scale.

**SET VIEW POINT**

set viewpoint by phi and theta offset from original view position.

**SET LIGHT SOURCE**

set light source phi and theta

**GET HARD COPY**

create PostScript file of output and send to a printer.

**NEW 3D PLOT**

load a new plot file.

**LOOK AT F77PLOT**

display a *drawplot* format (SAMPLE f77punch7 format) data file in lower viewport

**LOOK AT CONTOUR**

display a *contour* format data file in upper viewport.

**CALL PDRAW**

make a system call to *pdraw*.

**CALL CONTOUR**

make a system call to *contour*.

**CALL DRAWPLOT**

make a system call to *drawplot*.

**CALL DRAWMASK**

make a system call to *drawmask*.

**OPTIONS****EDIT PATTERN**

The following commands are for editing the stipple patterns displayed on the two sides of the window:

**ADD PATTERN**

Add a pattern with the specified name to the current list of patterns. The color and the stipple patterns are randomly generated and they can be changed by the following command.

**CHANGE PATTERN**

Select a pattern to be changed by clicking at the pattern in either one of the pattern maps displayed on both sides of the SIMPL-DIX windows. Three color bars for mixing RGB can be used to change the color of the selected pattern by holding down a mouse button and moving the mouse across the bars. The stipple patterns can be changed by modifying the current pattern on the bitmap displayed.

**MOVE PATTERN**

Move a selected pattern to another location in the color maps as specified by clicking the mouse.

**REMOVE PATTERN**

Remove a pattern from the current pattern maps.

**SAVE PATTERN**

Save a modified pattern definition list in a specified file.

**EDIT LAYOUT**

The following commands perform operations in the layout viewport.

**NEW LAYOUT**

loads a new CIF layout file.

**ZOOM**

zoom in a region of the layout. Escape (ESC) returns to the original unzoomed view.

**FRAME ON**

overlay a scale fram on the current layout.

**SET A CUTLINE**

draw a cutline on the layout and save for later use.

**UNIX SYSTEM CALL**

send a Unix command to the system.

**SPLAT TO CIF**

prompts for a SPLAT input file and creates a CIF file *splatfile.cif*.

**WORST**

Worst allows masks to be realigned. The updated masks can be saved in a CIF file.

**X TRANS**

translate a given mask along an x-direction. Positive x is to the right.

**Y TRANS**

translate a given mask along a y-direction. Positive y is up.

**UPDATE CURRENT LAYOUT**

use layout displayed in lower viewport as new layout.

**SAVE CURRENT LAYOUT**

save layout to a CIF file.

**HUNCH**

Enter the *HUNCH* geometric manipulation mode and the HUNCH menu is displayed. Hunch definitions specifying geometric operations between various mask layers can be specified in a hunch file and loaded in by the *LOAD HUNCH DEF* command or entered interactively by using the *EDIT HUNCH DEF* command. The syntax for the layer description making up a hunch definition is :

*<Layer Name> : <Operation Specification>*

The operations that can be performed and the associated specifications are :

Geometrical Bloat	<i>BLOAT (IntFactor) Layer</i>
Geometrical Scale	<i>SCALE (FloatFactor) Layer</i>
Geometrical Translation	<i>XTRANS (IntFactor) Layer</i>
Geometrical Translation	<i>YTRANS (IntFactor) Layer</i>
Logical Complement	<i>NOT Layer</i>
Logical Addition	<i>Layer OR Layer</i>
Logical Intersection	<i>Layer AND Layer</i>
Logical Subtraction	<i>Layer SUB Layer</i>
Logical Exclusive-or	<i>Layer XOR Layer</i>
Logical Inverse-and	<i>Layer NAND Layer</i>
Logical Inverse-or	<i>Layer NOR Layer</i>
Topological Boundary	<i>BOUND Layer</i>
Topological Inside	<i>Layer IN Layer</i>
Topological Outside	<i>Layer OUT Layer</i>

The HUNCH menu has the following commands :

#### **LOAD HUNCH DEF**

Load a file containing hunch definitions for the current layout and display the geometric view of the layer description. These loaded hunch definitions are known as the current hunch list.

#### **LIST/DISP HUNCH DEF**

Toggle between showing the layer description statements in a hunch definition file and displaying the geometric view of the layer descriptions.

#### **EDIT HUNCH DEF**

Edit the current hunch layer descriptions. All modifications to the current hunch list will not affect the original file containing the hunch definitions unless they are explicitly saved by the *SAVE HUNCH DEF* command. The following are hunch editing commands :

#### **ADD DEF**

Add a new layer description to the current hunch list.

#### **CHANGE DEF**

Change a layer description in the current hunch list. A description is selected for modification by clicking the mouse at the layer name in the appropriate statement displayed.

#### **REMOVE DEF**

Remove a current layer description.

#### **SET DEF**

Select a layer description to be displayed geometrically. The default is all current layer descriptions in the hunch lists are selected. A selected layer description will be highlighted in the Reverse Video color.

#### **SAVE HUNCH DEF**

All modifications to the current hunch list is saved in a file by this command.

#### **SAVE HUNCH LAYER**

Save the hunch layers defined by the current hunch list in a file. The hunch layers is written in CIF.

#### **RETURN**

Return to the parent of the current menu in the menu hierarchy.

## QUIT SESSION

Leave *sample3D-gui* completely. If any of the files modified by *sample3D-gui* are not saved, a dialog box will pop up to issue a warning and ask for further instruction.

## X DEFAULTS

*sample3D-gui* allows the user to preset defaults in a customization file called *.Xdefaults* in the user's home directory. The format within the file is where *program\_name* is the local name for *sample3D-gui*. See X (1) for more details. On X11 use of the *xrdb* command is necessary to update the X resource database when *.Xdefaults* is changed.

Keywords recognized by *sample3D-gui* are:

<b>AutoRaise</b>	If " on enable the auto-raise mode.
<b>Background</b>	Set the background color for color displays.
<b>Bell</b>	If " on enable the bell mode.
<b>BodyFont</b>	Specify the font for the standard text display.
<b>Border</b>	Set the border color for color displays.
<b>BorderWidth</b>	Set the border width of the window.
<b>CONTOUR</b>	Specify the path to <i>contour</i> program (for <i>SPLAT</i> ).
<b>PDRAW</b>	Specify the path to <i>pdraw</i> program.
<b>DRAWPLOT</b>	Specify the path to <i>drawplot</i> program.
<b>DRAWMASK</b>	Specify the path to <i>drawmask</i> program.
<b>SPLAT</b>	Specify the path to <i>splat</i> program.
<b>BLEACH</b>	Specify the path to <i>bleach</i> program.
<b>ETCH</b>	Specify the path to <i>etch</i> program.
<b>NETCH</b>	Specify the path to <i>netch</i> program.
<b>SPLAT2CIF</b>	Specify the path to <i>splat2cif</i> program.
<b>Foreground</b>	Set the text color for color displays.
<b>Highlight</b>	Set the highlight color for color displays.
<b>MenuFont</b>	Specify the font for the command menu.
<b>Mouse</b>	Set the mouse cursor color for color displays.
<b>PatternFile</b>	Specify the default pattern file.
<b>ReverseVideo</b>	Set the reverse-video mode.
<b>TitleBar</b>	If " on the title bar is displayed on startup.
<b>TitleFont</b>	Specify the font for the title bar.
<b>WindowGeometry</b>	Specify the <i>sample3D-gui</i> window geometry at startup; this enables automatic creation and placement of the window.



**ENVIRONMENT**

**DISPLAY**            To get the default host and display number.

**SEE ALSO**

X(1), Xlib Documentation  
*A User's Guide for Sample-3D v1.0: Lithography Simulation.*

**AUTHOR**

Edward W. Scheckler,  
HUNCH and many original routines by H.C. Wu,  
original CIF to SPLAT routines, and X11 port by J.R. Camagna.  
University of California at Berkeley

**NAME**

**ptom** - pdraw to Mathematica conversion

**SYNOPSIS**

**ptom** [-e] infile outfile

**DESCRIPTION**

**PTOM** Takes an input file of polygons in pdraw 3D plot format, and converts it into a list of polygons recognizable by Mathematica. The file created by **PTOM** is a Mathematica command line assigning a list of polygons to the name *pgons*. To use in Mathematica, load the output file, and display the graphics from a Mathematica console:

```
In[1]:= << outfile;
```

```
In[2]: Show[Graphics3D[pgons]];
```

The data may be manipulated as any other Mathematica output.

**OPTIONS**

**-e** forces polygon edges to be included in the Mathematica geometry description.

**AUTHOR**

Edward W. Scheckler

**SEE ALSO**

pdraw(L), math(L), psfix(L)

**NAME**

**cutaway - pdraw data file clipper**

**SYNOPSIS**

**cutaway *xmin ymin xmax ymax filename***

**DESCRIPTION**

**CUTAWAY** Takes files in *pdraw* 3D plot format and cuts away everything outside of the box with corners (*xmin*, *ymin*) and (*xmax*, *ymax*). The result is stored in *filename.cut* which is also in *pdraw* format. This program is designed to create cutaway views of 3D process simulation data generated by **NETCH**.

**AUTHOR**

Edward W. Scheckler

**SEE ALSO**

**pdraw(L)**, **netch(L)**

## NAME

**pdraw** - 3D plot program for X-windows and Postscript (by K.K.H. Toh)

## SYNOPSIS

```
pdraw [-v vx vy vz] [-o options-file] [-Pprinter] [-s scale] [-e] [-h] [-ill phi theta] [-ior] [-nosort] [-noplot] [-print] [-ps] [-tl "toplabel"] [-xl "xlabel"] [-yl "ylabel"] [-zl "zlabel"]  
plotfile1 plotfile2...
```

## DESCRIPTIONS

**Pdraw** is a program for drawing 3D plots on X10 or X11 windows. The program will also produce a POSTSCRIPT plot which can be dumped out to an APPLE Laserwriter. **Pdraw** reads in x-y-z data from a *plot-file* and manipulates that data according to options specified either in the command-line or in a *options-file*. The *plot-file* can be compressed (see `compress(1)`); compressed files will be uncompressed automatically. The program then plots lines on a screen or dumps the plots to a POSTSCRIPT file. Some simple illumination and hidden line capability is included.

The *plot-file* input data consists of alternating x, y and z values, in the format shown below.

## Data File Format (plot-file)

```
xmin xmax ymin ymax zmin zmax  
ncurves  
npts  
x1 y1 z1  
x2 y2 z2  
..  
..  
..  
npts  
x1 y1 z1  
x2 y2 z2  
..  
..  
..
```

In the above, *xmin*, *xmax*, *ymin*, *ymax*, *zmin* and *zmax* are lower and upper bounds of the desired plot, *ncurves* are the number of curves to be plotted, and *npts* are the number of points in each curve. The data file can consist of more than one set of curves to be plotted; each set (i.e. one set for each separate graph) is separated from the next by a blank line.

If the plot file name end in ".int", the illumination intensity for each polygon must be specified explicitly. The illumination intensity is given as a real number between 0 and 1, and is inserted after *npts* and the first point in the polygon.

If the plot file name ends in ".pif", a format similar to the Profile Interchange Format for IC process simulation is used.

Upon starting up the program, **pdraw** will read in the data stored in the *plot-file*, as well as any plotting options specified either in the command line or in the *options-file*. **Pdraw** then uses the given view

direction to rotate and transform the plot onto a plane perpendicular to the viewing vector. Currently, only parallel projection is supported. If the program is being run under X-windows, the plot will then be drawn on the screen. The viewing vector can be changed using the "H", "J", "K", "L" and "O" keys on the the keyboards; the plot on the screen can be rotated sideways using the "H" or "L" keys, rotated up or down using the "J" and "K" keys, and drawn with the original viewing vector using "O". The "A", "S", "D" and "F" keys will produce 90° rotations. "Z" will plot the image projected on the x-y plane ( $z=\text{constant}$ ), "Y" will plot the image projected on the x-z plane ( $y=\text{constant}$ ), and "X" will plot the image projected on the y-z plane ( $x=\text{constant}$ ). The final view angle will be saved and used for the POSTSCRIPT plot. Finally, the user will be prompted as to whether or not the POSTSCRIPT plot is to be sent to a printer.

## OPTIONS

### **-v vx vy vz**

reads in the viewing eye position, relative to (0,0,0). The plot will be rotated and transformed so that the z-axis is parallel to this position. For example, a view position of (1,0,0) means that the 3D structure is being viewed with parallel projection from the x-axis.

### **-o options-file**

reads plotting options from the file *options-file*. Each option specification consists of a keyword and its corresponding value. The parser recognizes only a limited set of keywords; their values are either numbers, quoted strings, or the words "on" and "off". All the words in the option specification must be on the same line. The pound sign (#) indicates that the remainder of the line is a comment to be ignored by the parser.

#### **List of Options (options-file)**

xlabel "LABEL"	#[default = "X-Axis"]	- for the x-label
ylabel "LABEL"	#[default = "Y-Axis"]	- for the y-label
zlabel "LABEL"	#[default = "Z-Axis"]	- for the z-label
toplabel "LABEL"	#[default = "3D Line Plot"]	- for the top-label
equalscale on/off	#[default = on]	- for equal x-y scaling
postscript on/off	#[default = on]	- for postscript (PS) plot
printplot on/off	#[default = off]	- send PS file to printer
noplot on/off	#[default = off]	- no graphics plot
printer "PRINTER"	#[default = \$PRINTER]	- define the printer
line on/off	#[default = on]	- draw the line
linechange on/off	#[default = off]	- change the linetypes
marker on/off	#[default = off]	- draw the marker
markerchange on/off	#[default = off]	- change the markertypes
hiddenline on/off	#[default = off]	- for hidden-line drawings
nosort on/off	#[default = off]	- for hidden-line drawings
scale [0.1 - 1.0]	#[default = 1.00]	- scales the PS plot
xticks [1 - 20]	#[default = 2]	- no. of x-divisions
yticks [1 - 20]	#[default = 2]	- no. of y-divisions
zticks [1 - 20]	#[default = 2]	- no. of z-divisions

### **-Pprinter**

specifies which printer to which to send the postscript plot. The current default sets the printer name to the environment variable \$PRINTER. If this variable is not set, then the

printer used is the lp550M printer in 550M Cory.

**-s scale**

sets a scale factor. This is used only for POSTSCRIPT plotting.

**-e** forces *unequal* scales to be applied to the x, y and z axes. The boundary of the 3D object will then resemble a cube.

**-h** draws polygons with hidden-lines.

**-ill phi theta**

When the hidden-lines feature is on, this option produces a greyscale illumination plot of the surface. Phi and theta describe the orientation of a distant light source. The shading of a surface polygon is proportional to the cosine of the light angle of incidence. Direct incidence produces a white polygon. Incidence from the back produces a black polygon. To determine the orientation (front or back) of a polygon, the program assumes that the points of the polygon are listed in clockwise order when viewed from the top (see -ior). Also, the first three points must not be co-linear. This feature is only available in the X11 version.

**-ior** Program does not assume that polygons are oriented. The illumination shading is taken as the absolut value of the illumination and surface-normal cosine.

**-nosort**

prevents sorting for the hidden-line option.

**-noplot**

prevents plots on the graphics display.

**-print**

sends plots to the printer automatically.

**-ps** turns the postscript plotting option off. This can also be done by setting the POSTSCRIPT environment variable to OFF. e.g. % setenv POSTSCRIPT off

**-tl toplevel**

**-xl xlabel**

**-yl ylabel**

**-zl zlabel**

sets label options.

**host:display**

opens a window on the given host and display

**=geom**

**-rv**

**-bw border-width**

**-bd color**

**-fg color**

**-bg color**

sets input options for the X-window system.

**BUGS**

Not really that many. The POSTSCRIPT labels need to be adjusted. Some implementations of X11 do not allocate illumination or intensity plots correctly. The labels don't come out well when the picture is rotated beyond the default view. The parser needs to be improved. There should be a better way to put change linetypes and markertypes. Log axes might be nice. Also should incorporate drawplot modifications here.

**AUTHOR**

Kenny K.H. Toh, modified by Edward Scheckler

**FILES**

*dataplot.ps* temporary POSTSCRIPT file

**SEE ALSO**

`contour(L)`, `drawplot(L)`

**NAME**

contour - contour plot program for X-windows, and Postscript (by K.K.H. Toh)

**SYNOPSIS**

**contour** [-o *options-file*] [-P*printer*] [-s *scale*] [-c *level*] [-cstep *step-size*] [-ex] [-grid] [-grey]  
 [-j *joinlevel*] [-l] [-noplot] [-print] [-ps] [-old] [-tl "*toplabel*"] [-xl "*xlabel*"] [-yl "*ylabel*"] [-  
 3D] *contour-file*

**DESCRIPTIONS**

Contour is a program for drawing contour plots on X10/X11 windows or HP2648 terminals. The program will also produce a POSTSCRIPT plot which can be dumped out to an APPLE Laserwriter. Contour reads in data on a 3D surface from a *contour-file* and manipulates that data according to options specified either in the command-line or in a *options-file*. The *plot-file* can be compressed (see *compress(1)*); compressed files will be uncompressed automatically. The program then draws contours on a screen or dumps the contours to a POSTSCRIPT file.

The 3D surface input data consists of z-values of the 3D surface, arranged on a rectangular grid of size  $(x_{max} - x_{min}) \times (y_{max} - y_{min})$ . The data file format is shown below.

**Data File Format (contour-file)**

```
xmin xmax ymin ymax
nxpts nypts
z1 (x0,y0)
z2 (x1,y0)
..
..
..
```

In the above, *xmin*, *xmax*, *ymin* and *ymax* are lower and upper bounds of the grid, and *nxpts* and *nypts* are the number of grid divisions in x and y. Alternately, if the -3D flag is specified, the data-file could consist of triangles, specified in *pdraw(L)* format, i.e.,

**Data File Format (plot-file)**

```
xmin xmax ymin ymax zmin zmax
ncurves
4.0
x1 y1 z1
x2 y2 z2
x3 y3 z3
x1 y1 z1
4.0
x1 y1 z1
x2 y2 z2
x3 y3 z3
x1 y1 z1
..
..
```



Upon starting up the program, *contour* will read in the data stored in the *contour-file* and will then find the maximum and minimum z-values of the surface. It will then prompt for a contour step-size (i.e. the contour increments), and read in any plotting options specified either in the command line or in the *options-file*. The plot will then be drawn on the screen if possible. Finally, the user will be prompted as to whether or not the POSTSCRIPT plot is to be sent to a printer.

## OPTIONS

### -o *options-file*

reads plotting options from the file *options-file*. Each option specification consists of a keyword and its corresponding value. The parser recognizes only a limited set of keywords; their values are either numbers, quoted strings, or the words "on" and "off". All the words in the option specification must be on the same line. The pound sign (#) indicates that the remainder of the line is a comment to be ignored by the parser.

#### List of Options (*options-file*)

xlabel "LABEL"	#[default = "X-AXIS"]	- for the x-label
ylabel "LABEL"	#[default = "Y-AXIS"]	- for the y-label
toplabel "LABEL"	#[default = "CONTOUR PLOT"]	- for the top-label
grid on/off	#[default = off]	- draws a grid
equalscale on/off	#[default = on]	- for equal x-y scaling
noplot on/off	#[default = off]	- don't draw graphics plot
postscript on/off	#[default = on]	- for postscript (PS) plot
printplot on/off	#[default = off]	- print PS file automatically
printer "PRINTER"	#[default = \$PRINTER]	- define the printer
contlabel on/off	#[default = on]	- for contour labels
joinlevel high/low	#[default = --]	- for joining curves
scale [0.1 - 1.0]	#[default = 1.00]	- scales the PS plot
linetypes [1 - 3]	#[default = 2]	- no. of contour linetypes
xticks [1 - 20]	#[default = 4]	- no. of x-divisions
yticks [1 - 20]	#[default = 4]	- no. of y-divisions

### -P*printer*

specifies which printer to which to send the postscript plot. The current default sets the printer name to the environment variable \$PRINTER. If this variable is not set, then the printer used is the lp550M printer in 550M Cory.

### -s *scale*

sets a scale factor. This is used only for POSTSCRIPT plotting.

### -c *level*

forces the program to compute the contours at a single value of z, specified by *level*. The contours will be written to the file *image.cont*. The output data is organized in SAMPLE plot format, i.e.,

#### Single contour (*image.cont*)

```
xmin, xmax, ymin, ymax
ncurves
npts
pt1.x pt1.y
pt2.x pt2.y
```

- ```

..
..
..
npts
pt1.x pt1.y
..
..

```
- cstep *step-size***  
defines the contour step-size. If this is not defined, the program prompts for the step-size.
  - ex** expands the data into a triangular mesh. The mesh is stored in *contour-file.3D*. Thus, if the initial contour file is named *im.cont*, then the surface mesh will be stored in *im.cont.3D*.
  - grid**  
forces a grid to be drawn.
  - grey**  
instead of contours, a greyscale plot will be drawn. Greyscale postscript plots can also be made this way.
  - j *joinlevel***  
causes contour curves to be joined where possible. This is done by defining a boundary layer around the rectangular border, and setting the z-value of that boundary layer at either the maximum or minimum z-value. *joinlevel = HIGH* or *high* sets the border z-value to its maximum value: this is useful for plots which have high average z-values. *joinlevel = LOW* or *low* sets the border z-value to its minimum value: this is useful for plots which have low average z-values.
  - l** suppresses the contour labels.
  - noplots**  
prevents plots on the graphics display.
  - print**  
sends the POSTSCRIPT lot to the printer automatically.
  - ps** turns off the postscript plotting mode. This can also be done by setting the POSTSCRIPT environment variable to OFF. e.g. `% setenv POSTSCRIPT off`
  - old** accepts an older contour format for the *contour-file*, based on a rectangular  $50 \times 50$  array. The data file format is similar to that described earlier except for the first 3 lines. *nxpts* and *nypts* (both equal to 50) are omitted. Also, *xmin* and *ymin* are the coordinates of the lower left corner of the grid while *xlength* and *ylength* define the area being examined.

#### Old Data File Format (contour-file)

```

xmin ymin xlength ylength
z1
z2
..
..
..

```

**-tl toplabel**

**-xl xlabel**

**-yl ylabel**

sets label options.

**-3D** specifies the plotfile to be of **pdraw(L)** format. As such, the plotfile should consist of triangles only. For example, "contour -3D curves.plot.3D" will produce contour plots of the 3D plotfile *curves.plot.3D*.

**host:display**

opens a window on the given host and display

**-d host:display**

**=geom**

**-rv**

**-bw border-width**

**-bd color**

**-fg color**

**-bg color**

**-fn font-name**

sets input options for the X-window system.

## AUTHOR

Kenny K.H. Toh, modified by Edward W. Scheckler

## FILES

*dataplot.ps* temporary POSTSCRIPT file

*image.cont* file produced by **-c level** option

## SEE ALSO

SPLAT, drawplot(L), pdraw(L)