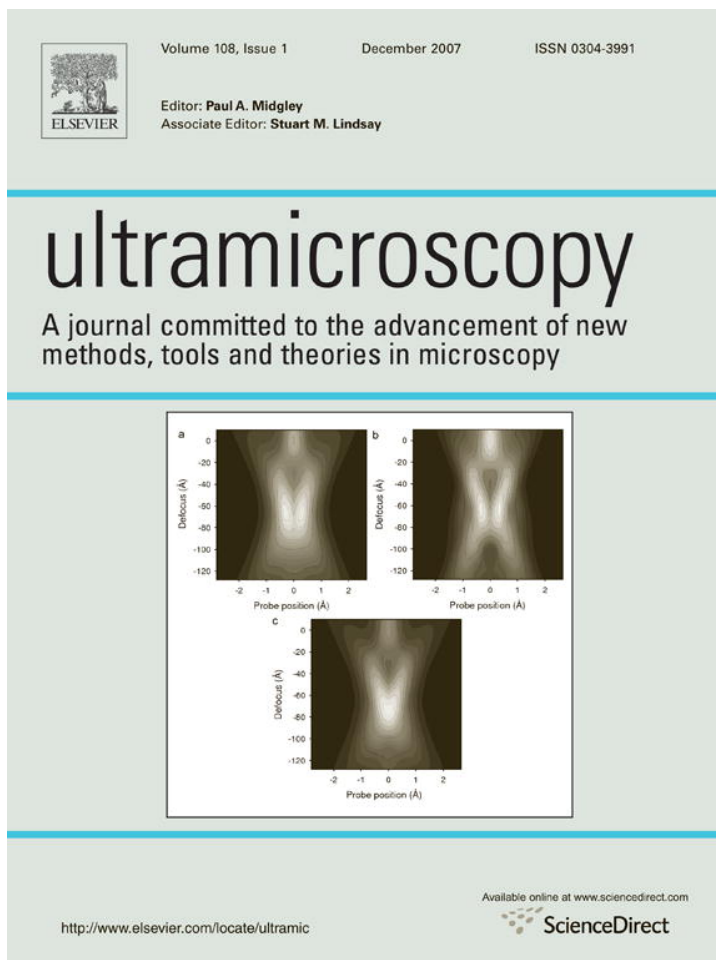


Provided for non-commercial research and education use.
Not for reproduction, distribution or commercial use.



This article was published in an Elsevier journal. The attached copy is furnished to the author for non-commercial research and education use, including for instruction at the author's institution, sharing with colleagues and providing to institution administration.

Other uses, including reproduction and distribution, or selling or licensing copies, or posting to personal, institutional or third party websites are prohibited.

In most cases authors are permitted to post their version of the article (e.g. in Word or Tex form) to their personal website or institutional repository. Authors requiring further information regarding Elsevier's archiving and manuscript policies are encouraged to visit:

<http://www.elsevier.com/copyright>



General three-dimensional image simulation and surface reconstruction in scanning probe microscopy using a dixel representation

Xiaoping Qian^{a,*}, J.S. Villarrubia^b

^a*Mechanical and Aerospace Engineering, Illinois Institute of Technology, Chicago, IL 60613, USA*

^b*National Institute of Standards and Technology,¹ 100 Bureau Drive, Stop 8212, Gaithersburg, MD 20899, USA*

Received 23 November 2006; received in revised form 17 February 2007; accepted 22 February 2007

Abstract

The ability to image complex general three-dimensional (3D) structures, including reentrant surfaces and undercut features using scanning probe microscopy, is becoming increasingly important in many small length-scale applications. This paper presents a dixel data representation and its algorithm implementation for scanning probe microscope (SPM) image simulation (morphological dilation) and surface reconstruction (erosion) on such general 3D structures. Validation using simulations, some of which are modeled upon actual atomic force microscope data, demonstrates that the dixel representation can efficiently simulate SPM imaging and reconstruct the sample surface from measured images, including those with reentrant surfaces and undercut features.

© 2007 Elsevier B.V. All rights reserved.

PACS: 82.20.Wt; 83.85.Ns; 87.64.Dz

Keywords: Dilation; Erosion; Mathematical morphology; Atomic force microscopy; Scanning probe microscopy; Dixel representation

1. Introduction

Scanning probe microscopy (SPM) includes atomic force microscopy (AFM), scanning tunneling microscopy (STM) and a number of other variants. It has become one of the most important nano-scale probing and manipulation tools. It provides a vital tool for dimensional measurement of topographic features at nanometer-scale resolution [1]. The diminishing feature size in semiconductors and the growing academic and industrial research interests in nanotechnologies have led to the widespread use of SPM in a variety of applications. However, conventional SPM instruments, due to their cone-like probe shape and the unidirectional feedback systems, have their images restricted to shapes (“umbras”) characterized by a single height at each lateral position. These instruments cannot

accurately image reentrant or even nearly vertical features of specimens. The dimensional characterization of such reentrant surfaces at the nano-scale, including measurements of side-wall angles, side-wall roughness, and width variability of lines and trenches, are urgently needed in the semiconductor industry as feature size is reduced to follow the International Technology Roadmap for Semiconductors [2].

For a number of years now, probes with lateral protrusions and feedback systems with bi-directional servo control have been incorporated into the newer AFM instruments [3–6]. In these SPM systems, probes are designed with flare- or hammerhead-like shapes to access reentrant surfaces and undercut features. Fig. 1 gives a schematic description of probe shape extension from conventional cone-like probe tips restricted to non-reentrant surfaces, to flared tips for reentrant surfaces, and to hammerhead shaped tips for severely undercut features. These instruments, which are capable of imaging undercut features, have found applications as reference metrology tools at SEMATECH and in a number of semiconductor fabrication facilities.

*Corresponding author. Tel.: +1 312 567 5855; fax: +1 312 567 7230.

E-mail addresses: qian@iit.edu (X. Qian), John.Villarrubia@nist.gov (J.S. Villarrubia).

¹Contributions of the National Institute of Standards and Technology are not subject to copyright.

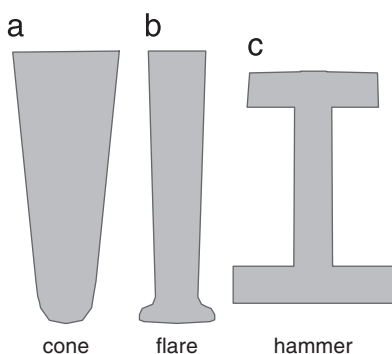


Fig. 1. SPM tip shapes: (a) conventional cone-like probe tip restricted to non-reentrant surfaces; (b) flared tip for reentrant side walls; (c) hammerhead shaped tip for severely undercut features.

Analytical methods for treating the data have not kept pace with the improved hardware. Such methods are needed because SPM images are distorted representations of samples due to the dilation of the image by the tip [7–9]. If S is the sample and P the reflection of the tip through the origin then

$$I = S \oplus P \quad (1)$$

describes the image. A capability for image simulation implies the ability to calculate this dilation. Such simulation is needed in order to understand the tip effect and the lost details on a given specimen due to the ever-blunter tips since the tip sharpness dulls with use [10]. Accurate measurement of specimen topographic features requires methods of reconstructing the specimen shape (to the extent possible) given its image. Sample reconstruction is ordinarily performed using erosion:

$$S_r = I \ominus P. \quad (2)$$

S_r is the set describing the deepest penetration of the tip. As such, the actual sample is a subset of S_r . An alternative to explicit dilation and erosion relies upon matching surface slopes [11]. Although this is in principle equivalent, implementations of the slope method require numerical derivatives, so in practice they usually involve filtering or other operations to improve accuracy and ensure stability.

Algorithmic implementations of these methods have not been able to match the progress in hardware for the following reasons:

- With a single exception to be discussed below, current algorithms are based on grayscale morphology with the assumption of an umbra specimen shape. That is they assume surfaces can be represented as single-value functions. However, reentrant surfaces possess multiple z values for a given (x, y) coordinate and thus cannot be represented by such functions.
- For samples that are not single-valued, the typical procedure to correct for the tip effect has been to subtract a constant offset that corresponds to the overall tip width [4]. When appropriate, such as for sloped sidewalls, a correction may also be made for the effect of

the vertical offset height of the tip flare. This is typically done by scanning an undercut characterizer to estimate the offset height and using the image to estimate the sidewall slope of the feature and performing an extrapolation to correct the resulting width bias [12,13]. These methods do not generate a reconstructed profile of the surface, but only attempt to correct the effect of the tip on a specific measurand. However, these methods are less effective on complex and irregular structures where simplified models are inappropriate.

In this paper, new mathematical morphology software for AFM image simulation and surface reconstruction, applicable for complex three-dimensional (3D) structures with undercut features, is introduced. The approach is based on a representation in which the usual rectangular array of pixels is replaced by an array of “dexels.” A pixel contains a single height value, representing the height of an object’s surface. Points below this surface are inside the object. Points above are outside. Unlike a pixel, a dexel (depth element) may have multiple heights, each of which represents the height of a transition from inside to outside of an object or vice versa. This allows undercut features to be represented by dexels.

In this paper we introduce such a dexel representation and develop algorithms for operations on objects specified in terms of arrays of them. Our implementation is *complete* in the sense that (a) any 3D object may be represented in a dexel form to any desired degree of accuracy, simply by choosing the resolution high enough (i.e., by making the lateral spacing of dexels small enough), and (b) we implement all of the basic set operations—reflection, complement, translation, union, intersection, dilation, and erosion.

Within this group of operations, reflection, complement, translation, and union are treated as primitives that are used to construct the more complex operations, including intersection, dilation, and erosion. Dilation and erosion are directly applicable to SPM, since they represent, respectively, the imaging process and reconstruction of either the sample or the tip.

The implementation is compact in the sense that algorithms are reused whenever possible. For example, duality allows erosion to be implemented in terms of the dilation and complement operators. Similarly, set intersection is implemented in terms of the complement and union operators. In this way complicated algorithms are confined to a relatively few places in the code, reducing the opportunity for coding errors and improving ease of maintenance. This strategy is possible because the complement operation is very efficient within our dexel representation—the complement of a dexel can be performed by changing the sense of a single two-state flag.

The exception to our statement that current methods are limited to single-valued surfaces is two approaches recently introduced by Dahlen et al. [3]. One is a slope-matching technique. The other is billed by the authors as an

“erosion” algorithm (quotes in original) that “is not a direct application of erosion in a strict sense” because it obtains the outer boundary of a surface rather than a complete set of points describing the region. It appears to be a swept volume subtraction method applied to the image surface to reconstruct the sample. It was implemented for 2D profiles and demonstrated [14] on these to give good agreement with TEM cross-sections. The method should be extendable to 3D. Despite the authors’ modesty, it appears to us that Dahlen’s method is a legitimate implementation of erosion for the surfaces that it treats. If it and the method we present here are both correctly implemented, they should yield the same results. The main distinguishing features of the present work are: (1) The dixel implementation is a rigorous implementation of set-theoretic operators. Because of this we can be certain that the theorems of set theory and mathematical morphology apply to them. This allows relatively easy building up of more complex operations out of simple ones. Indeed, we have already made some such extensions. (2) The present implementation is not limited to erosion alone, but as mentioned above includes dilation, union, intersection, and other set operations. (3) The present implementation is already extended to 3D.

In Section 2, we describe the dixel representation. In Section 3, we provide details of how the dixel representation is used to implement mathematical morphology and set operations. In Section 4 we describe the sense in which an array of dexels (a 2D regular grid) approximates a real extended object and show that one may make the approximation as good as needed by choosing the grid spacing fine enough. A pixel is a special case of the more general dixel. We therefore expect agreement between the new dixel-based algorithms and the existing grayscale morphology implementation for those cases where both are applicable. In Section 5 we demonstrate this agreement. We also demonstrate that our implementation gives the correct answer for a calculation on a simple 3D model structure for which the correct answer is independently known.

2. The dixel-based representation

2.1. Selection of computer representation

When choosing a data representation for SPM imaging of 3D structures, representation characteristics such as geometric coverage, compactness, and algorithm efficiency are the key factors. Lack of sufficient coverage is the reason for eliminating grayscale images from consideration—i.e., such images cannot represent reentrant structures. In this section we restrict consideration to methods able to represent general 3D objects. Interestingly, the subject of solid modeling for representing such objects is quite mature, and complex set and morphological operations have also been studied in the solid modeling community. Two common types of 3D representation [15] are constructive solid geometry (CSG) and boundary repre-

sentation (B-rep). A CSG model is based on the notion that a physical object can be divided into a set of primitives that can be combined in a certain order following a set of rules (e.g., simple set operations like union and intersection plus transformations like translation and rotation) to form the object. These primitives and rules are represented in a tree data structure. A B-rep model stores the boundary information for a solid (e.g., vertices, edges, and faces, together with the information on how they are connected). Alternatively, a solid can be described by dividing space into a 3D regular grid. Then a 3D array of ones and zeros (for example) can designate which volume elements (or voxels) in the grid are inside the object and which outside. An octree representation is a tree data structure in which space is recursively subdivided into octants. This is similar to a voxel representation except that volume elements are not all of equal size, so some parts of an object (e.g., the interior) may be described with a coarse representation while others (e.g., near a boundary) may be described with a higher resolution.

Solid modeling has become a mature discipline, in which various computer representations and corresponding modeling algorithms that can model general 3D solids have been thoroughly studied. Thus, the objective of this paper is not to develop a novel computer representation for general 3D solid modeling, but rather to adapt one to the mathematical and computational requirements intrinsic to the particular application we have, i.e., SPM imaging, and to identify the appropriate computer representation and algorithms.

The morphological operations for SPM imaging are analogous to modeling operations in swept volume-based simulation software for numerical controlled (NC) machining path generation, even though the dilation operation is rarely used in NC simulation. The SPM probe movement forming a swept volume is analogous to robotics and NC cutting motion. In NC path generation, software simulation is needed to verify the cutting path, to avoid collision and gauging, and to examine the resulting surface shape and accuracy in comparison to the nominal surface geometry. NC path simulation has received tremendous research. The simulation methods include accurate approaches [16,17] or and approximate methods [18].

The approaches based on CSG and B-rep are theoretically capable of providing accurate NC milling simulation and verification. However, they are computationally intense. The cost of simulation is reported to be $O(N^4)$, where N is the number of tool movements [19]. A complex NC program can consist of thousands of motion steps. SPM imaging can consist of hundreds of thousands movements, making such computation even more intractable.

The approximate methods have $O(N)$ computational complexity and they include the voxel-based approach, dixel (depth element) approach, and octree approach [20,21]. The voxel representation is easy to implement but requires larger storage space. In comparison, the dixel [22]

is a version of run length encoded volumetric data representation where 3D objects are represented as a set of 1D blocks with depth on a grid. Since this is the representation we chose, details will be provided below. The dixel and its variants have been widely used in a variety of NC simulation systems [18,22,23].

Table 1 gives a comparison of the suitability of various 3D computer representations for morphological representation. The comparison items include:

- *Ease of creation*: whether the initial 3D representation can be easily created from SPM imaging data. The creation of CSG and B-rep from SPM data involves the reconstructing higher order analytical surfaces from SPM data.
- *Accuracy*: whether the 3D representation can accurately represent various specimen and tip shapes. Neither voxel nor dixel represent analytical surfaces exactly. However, through controlling the sampling resolutions, they can represent the SPM image data with the same accuracy as B-rep.
- *Efficiency*: whether the 3D representation supports efficient morphological operations. Both CSG and B-rep have $O(N^4)$ computational complexity and dixel has $O(N)$, where N is the number of SPM imaging points.
- *Compactness*: whether the 3D representation can represent an arbitrarily shaped specimen in a compact manner. CSG representation can compactly represent an object through a series of set operations. The compactness of B-rep of a nanostructure from SPM measurement depends on the object and the required accuracy. To maintain representation accuracy, both dixel and voxel need larger number of cells. However, dixel is more efficient in z -axis since it uses the run length encoded representation.
- *Ease of coding*: whether the 3D representation and the corresponding morphological operations can be easily coded.
- *Compatibility*: whether the proposed 3D representation is compatible with grayscale morphological operations in conventional SPM imaging systems. The conversion of an umbra into a voxel requires the unnecessary assumption of a bottom for each pixel height.

It is clear from Table 1 that there is no single representation that is excellent in all aspects. CSG and B-rep are poor in computing efficiency and in initial model creation since

they require the explicit definition of surfaces from SPM images and probe data. The Boolean operations for B-rep are also challenging in terms of coding. However, they can compactly and accurately represent 3D specimen and probe shape. On the other hand, voxel representation is poor in terms of representation compactness since it involves discretizing the sample object in all three dimensions. However, it is easy to code, the initial model creation is easier, and the modeling algorithms are efficient. The dixel representation is a run length encoded version of volumetric data representation and is more compact and more accurate than voxel.

Therefore, we adopted dixel representation in our implementation of mathematical morphology software for general 3D structures.

2.2. Dixel-based representation

The dixel approach is a version of volumetric data representation where 3D objects are represented as a set of 1D blocks with depth on a grid. It is more compact and more accurate than voxel since it does not involve the discretization in the z -dimension.

We may construct a dixel object, A_d , associated with a real object A , as follows. First choose an origin and orientation for a rectangular coordinate system. Define a grid in the x - y plane of this coordinate system. The x coordinates in this grid are given by $x_i = x_0 + id_x$ for $i = 0, 1, \dots, m_x - 1$ with i an integer index, m_x the number of grid elements in the x direction, x_0 the position of the first such element, and d_x the grid spacing in the x direction. The y coordinates are similarly defined. Now imagine a line, L_{ij} (with z ranging from $-\infty$ to ∞) at x_i, y_j for each i, j in the grid. We can define A_d as

$$A_d = \bigcup_{i,j} (L_{ij} \cap A). \quad (3)$$

A 2D example of this is shown in Fig. 2. The object (shown gray) has undercut edges. Those intervals of the lines that are enclosed by the object are shown darker. The dixel object representation includes the collection of locations of the endpoints of these intervals in an indexed grid. Each column in the object is represented by a single dixel. The entire object is then a 2D array of dexels.

To support a complete set representation we allow $-\infty$ and ∞ as possible initial and final heights in a dixel. This is necessary, for example, to represent the complement of a bounded set, since such a complement will be unbounded.

Table 1
Representation comparison

3D representation	Ease of creation	Accuracy	Efficiency	Compactness	Ease of coding	Compatibility
CSG	N	Y	N	Y	N	N
B-rep	N	Y	N	?	N	N
Voxel	Y	Y	Y	N	Y	?
Dixel	Y	Y	Y	?	Y	Y

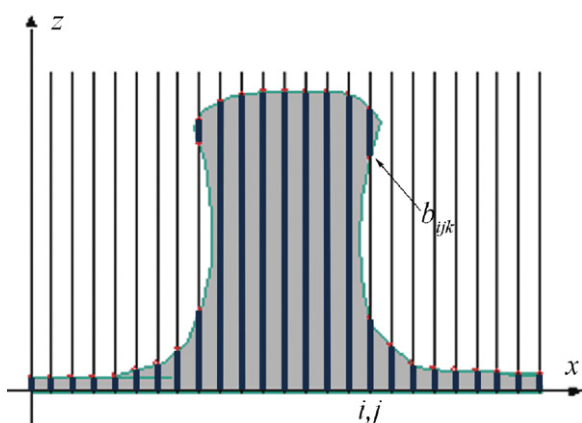


Fig. 2. Dixel representation of an object.

It is also desirable because it makes the umbra interpretation of SPM data that is used in grayscale morphology a special case of the dixel representation.

Formal representation properties of dixel, as noted in Ref. [24], include spatial addressability and spatial hashing, directionality, Boolean simplification, rigid motion, discrete translations, null-set representation, and completeness.

2.3. Data structure

Our dixel data structure is illustrated in Fig. 3. The data representation consists of a flag and a linked list of zero or more heights. (Absence of any heights would be indicated by a null list.) The flag may take on two values, “inside” or “outside,” which may be represented by $-1/1$, $0/1$, Boolean true/false, or any other convenient pair. The value of this flag indicates whether the starting position at $z \rightarrow -\infty$ is inside or outside the dixel. The heights in the height list are ordered from smallest to largest. The sense of inside/outside toggles back and forth at each height. The inside/outside state after the last height in the list is considered to remain in effect as $z \rightarrow \infty$.

Following are some examples:

- The dixel {flag = inside, HeightList = {10}} represents the interval $(-\infty, 10]$. (The starting position at $-\infty$ is inside, and the single height in the height list therefore represents an inside to outside transition.) This is equivalent to a pixel with value 10 in the umbra interpretation of an image.
- The dixel {flag = inside, HeightList = {0, 1, 3}} represents the union of intervals $(-\infty, 0]$ and $[1, 3]$.
- If the last height were omitted, there would be no final transition from inside to out. That is, {flag = inside, HeightList = {0, 1}} represents the union of intervals $(-\infty, 0]$ and $[1, \infty]$.
- {flag = outside, HeightList = {0, 1}} represents the interval $[0, 1]$.
- {flag = inside, HeightList = {null}} represents the universal set $(-\infty, \infty)$. (We start inside and there is no transition to outside.)

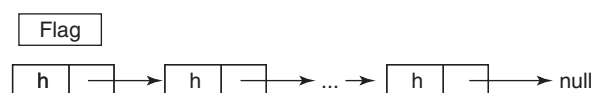


Fig. 3. Dixel data structure: linked ordered height list.

- {flag = outside, HeightList = {null}} represents the null set. (We start outside and there is no transition to inside.)

Just as an image is a 2D array of pixels, a dixel object in its simplest form is a 2D array of dexels. Additional information may be included if desired. For example, our implementation includes data structures to represent the lateral coordinates of the dixel at the 0,0 position and the lateral grid spacing (which we called d_x and d_y above).

2.4. Advantages of the data structure

There are two basic advantages of our data structure.

1. It easily represents umbra objects and objects with undercuts. By allowing dixel intervals to extend to $\pm\infty$ it allows representations of the null or universal sets, and it allows the complement of any set to be represented. The infinity value is represented logically—it is implicit in the flag and number of height values in the list—instead of as an explicit number (e.g., the maximum or minimum number of which the computer is capable) stored as a boundary height. This makes the representation compatible with any of the common data types on a computer that have no explicit representation for $\pm\infty$. (Simply using a large magnitude number to represent $\pm\infty$ in such a data type is machine-dependent and can lead to overflow during dilation or erosion.)
2. The second advantage of such representation is the efficiency of the set-theoretic complement operation. The complement of a dixel is obtained by toggling its flag from inside to outside or vice versa. The rapidity of this operation makes it possible to develop only two basic operations: union and dilation. All other set operations can be efficiently obtained as described in the next section using the complement operation and duality. This also makes the code compact and robust.

3. Set and morphology operations on dixel-represented objects

3.1. General considerations

We build the set and morphological operations on a few primitive operations: dixel *complement*, dixel *union* and block–block *dilation*. All other set operations such as intersection and subtraction, and dilation and erosion operations between dixel objects are derived from these three.

The usual erosion and intersection operations may lead to dangling boundaries. For example, the intervals $[-1, 0]$ and $[0, 1]$ overlap at the single point, $\{0\}$, on the boundary of both. If we were strict we would therefore need to keep careful track of whether each dixel height value in the height list represents the boundary of an open or closed interval. This would increase the complexity of the implementation without a compensating reward—in a real SPM a “0-width” feature (if such a thing could exist) would doubtless be broken unobserved by any tip that made contact with it. Nothing is lost by forbidding such features, since any actual small feature can still be represented by a small but finite size. We therefore implement *regularized* set operations [15]. Regularization closes intervals to the interior—so e.g., $(0, 5)$ becomes $[0, 5]$ —and prunes away boundaries that are not associated with any interior region (like the single $\{0\}$ in our first example). Among other conveniences, this choice insures that the set and morphological operations have the closure property. That is, the resulting object after the set and morphological operations remains a valid object in dixel representation and can participate in subsequent set operations.

3.2. Primitive set operations

3.2.1. Dixel complement

The dixel complement is accomplished by toggling the value of the dixel flag. Since the sense of inside/outside toggles at each boundary this changes all inside–outside transitions into outside–inside transitions and vice versa. Within the conventions of regularized set operations (where boundaries are closed) this is exactly what is meant by the complement. So, for example, the interval $[a, b]$ is expressed in dixel form as $\{\text{outside}, \{a, b\}\}$. Its regularized complement is $(-\infty, a] \cup [b, \infty)$, which in dixel form is $\{\text{inside}, \{a, b\}\}$.

3.2.2. Dixel–dixel union

A simple model of dixel–dixel union is illustrated in Fig. 4. Imagine the two dexels, A and B positioned side by side as shown. Let the “depth” inside A at position z be 1 if z is contained in A and 0 otherwise. The depths inside A and B are “projected” onto the screen at the right, which records their sum. The sum may be 0, 1, or 2. Obviously the depth inside $C = A \cup B$ must be 0 when the depth on the screen is 0 (z is outside both A and B) and 1 otherwise (i.e., when the total depth is 1 or 2, indicating z is inside one or both of A and B). Changes in the total depth obviously can only happen at the z values in the height lists of A and B .

To construct an efficient algorithm to implement this scheme, the initial depths in A and B are set to 0 or 1 depending upon whether their flag values are outside or inside. The total depth is initialized to the sum of these. The value of C 's flag is set to outside if this sum is 0, inside if it is 1 or 2. One then iterates up the two height lists. At each iteration, one examines the next available heights from the two lists. The smaller of the two is drawn from its list, the

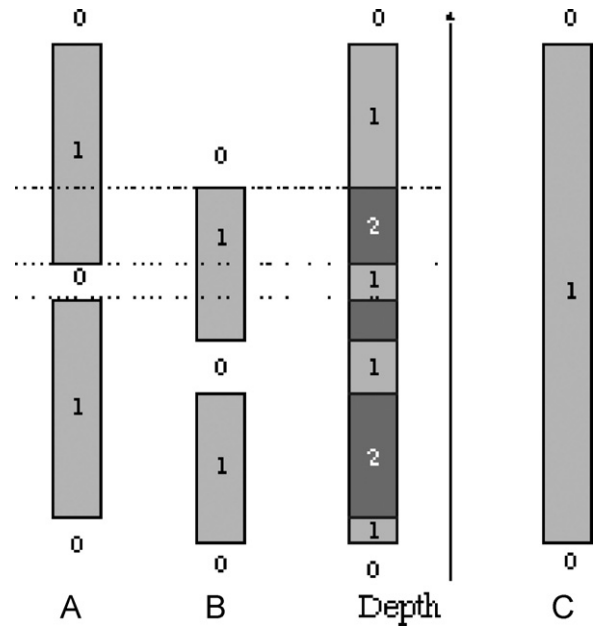


Fig. 4. Dixel–dixel union through keeping track of inside/outside status change.

corresponding depth is updated, and the total depth is updated. (If the two heights happen to be the same, they are both drawn and all depths are updated.) If the total depth changes from nonzero to zero or from zero to nonzero, the drawn height is saved in C 's list. (It represents a transition in C from inside to outside or vice versa.) Otherwise it is discarded. When a list runs out of heights, it is treated as though its next height is ∞ . That is, one proceeds by always drawing heights from the remaining list until it too is exhausted.

3.2.3. Block–block dilation

Each dixel can be thought of as the union of a number of intervals, or “blocks,” each of which is specified by its lower and upper boundaries. One of these blocks is labeled in Fig. 2. In terms of these blocks the dixel object, A_d , defined in Eq. (3) can also be written as

$$A_d = \bigcup_{ij} \left(\bigcup_k b_{ijk} \right), \quad (4)$$

where i and j are the grid indices, and k indexes the blocks within each dixel.

This is a useful conceptual formulation because the dilation of two blocks is particularly simple. We use the following definition of dilation:

$$A \oplus B = \bigcup_{b \in B} (A + b), \quad (5)$$

where $A + b$ with A a set and b a vector is defined by

$$A + b = \{a + b | a \in A\}. \quad (6)$$

That is, it is the set obtained by translating every point in A by b . When A and B are both blocks, denoted by $[h_i, h_{i+1}]$

and $[h_j, h_{j+1}]$, their dilation is another block:

$$[h_i, h_{i+1}] \oplus [h_j, h_{j+1}] = [h_i + h_j, h_{i+1} + h_{j+1}]. \quad (7)$$

3.3. Set operations on objects comprised of dexels

The above primitive operators can be used in combination to generate operators for dixel-represented objects. Reflection, complement, union, intersection and subtraction are straightforward. They also form the basis of two algorithms that will be directly applied for SPM image simulation and surface reconstruction: dilation and erosion.

The reflection of a set, A , denoted $-A$, replaces every vector $\mathbf{a} \in A$ by $-\mathbf{a}$. A dixel object consists of a 2D array of dexels. The lateral (x - y) coordinates of the reflection are accomplished the same way it is done for an image—the order along both axes is reversed. Each dixel must then also be reflected in the z -direction. This is done by multiplying all heights in the height list by -1 and reversing their order (so they are once again in increasing order). If there are an odd number of heights in the list, the flag must be toggled.

The complement of a dixel object is formed by taking the complement of all the dexels forming the object.

If two dixel objects are defined on the same grid, each grid point will be associated with one dixel from each object. The union is formed by forming the dixel unions at each grid coordinate using the dixel–dixel union procedure of the last section.

Intersection is computed by means of DeMorgan's law using the already described union and complement primitives:

$$A \cap B = (A^c \cup B^c)^c. \quad (8)$$

Set subtraction, $A - B$, removes from A all parts contained in B . Subtraction is implemented as

$$A - B = A \cap B^c = (A^c \cup B)^c. \quad (9)$$

The first form is simplest. The second, obtained by applying DeMorgan's law to the first, is expressed in terms of the union and complement primitives.

Dilation may be implemented in steps. Since the dilation primitive defined above is for block–block dilation, the first (and only new) step is to implement dilation of two dexels. We can think of a dixel as a union of blocks so that if a dixel D has n blocks and \mathcal{D} has m

$$D = \bigcup_{i=1}^n D_i, \\ \mathcal{D} = \bigcup_{j=1}^m \mathcal{D}_j, \quad (10)$$

D_i , $i = 1, \dots, n$ are D 's blocks and similarly for \mathcal{D} . It is a general property of dilation that

$$(A \cup B) \oplus C = (A \oplus C) \cup (B \oplus C). \quad (11)$$

See for example Ref. [25, proposition 15]. Using this rule, the dilation of dexels D and \mathcal{D} defined by Eq. (10) can be written as the union of dilations of blocks.

$$D \oplus \mathcal{D} = \bigcup_{i=1}^n D_i \oplus \bigcup_{j=1}^m \mathcal{D}_j = \bigcup_{i,j} (D_i \oplus \mathcal{D}_j), \quad (12)$$

where the final union is over all pairings of blocks in the two dexels. The dilations on the right side of Eq. (12) are all instances of the previously defined block–block dilation primitive, and unions of blocks are a special case of the union of dexels primitive. Hence Eq. (12) defines dixel–dixel dilation in terms of operations we already know how to do.

The second step is to define dilation of the 2D arrays of dexels that comprise dixel objects. This part is completely analogous to grayscale dilation of images. (See Ref. [8, Appendix C].) In grayscale dilation, pixel–pixel dilation amounts to a single sum—the upper interval bound in Eq. (7)—since for pixels there is never more than one block, and that block's lower bound is always $-\infty$. The union of two pixels is a pixel with value equal to the maximum of the two input values. Dilation of dixel objects can use the same algorithm except that the sum (pixel dilation) is replaced by dixel dilation as in Eq. (12), and the maximum (pixel union) is replaced by dixel union as described in Section 3.2.

The erosion operation between two dixel objects is done using dilation based on their duality property Ref. [25, Theorem 25]:

$$A \ominus B = [A^c \oplus (-B)]^c. \quad (13)$$

4. Discussion

As defined above, we can think of a dixel as a set of intervals along a line. Like a line, the dixel has extent in only one dimension. It has no width. This is a useful feature inasmuch as the dilation of two columnar objects, each with width w , is an object with width $2w$. Consequently, if we attempted to associate some nonzero fixed width with dexels, the dilation of one dixel with another would not produce a dixel, but rather an object two dexels wide. By defining dexels to have no width, this is avoided; closure of dilation and erosion with respect to dexels is preserved. This choice is analogous to the usual convention with pixel-represented objects. In grayscale morphology pixels are also commonly treated as having height but no width for the purpose of computing dilation and erosion.

As computationally convenient as this may be, it nevertheless raises questions about the sense in which a dixel or pixel representation can approximate a real object. Fig. 5 illustrates the problem. Suppose the curved solid line labeled A represents the boundary of an extended object. The dixel object, A_d , associated with A is an array of dexels on a regularly spaced grid. This is represented by the thick vertical line segments. A_d differs from A inasmuch as the

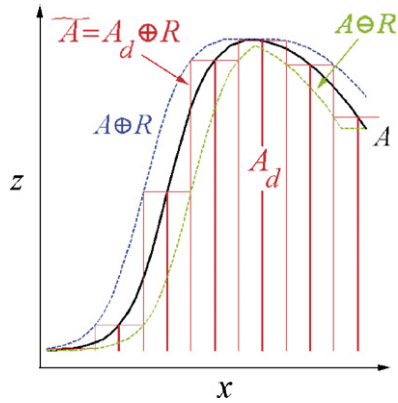


Fig. 5. The sense in which a dixel object approximates a real one. The dixel object A_d consists of an array of dexels (the thick vertical bars), each terminating on the real object, A , it is intended to approximate. Dilation by a horizontal grid element, R , expands A_d into rectangular blocks (\tilde{A} , shown with thinner lines). \tilde{A} is bounded above and below by $A \oplus R$ (outer dashed) and $A \ominus R$ (inner dashed) as shown.

space between dexels is not filled—its total volume is 0. Furthermore, as we have just discussed, dilation of real objects results in an object of width greater than either of the original objects alone. Is some part of this width increase neglected due to the treatment of dexels as having no width? Intuitively, it seems that the importance of these concerns should diminish as the grid spacing grows finer. In the following discussion, we sketch one way this intuitive insight might be placed on a firmer footing.

Define an object, R , to be a horizontal rod of length r in the x direction (left to right in Fig. 5). We consider its origin to be at its center. Then define

$$\tilde{A} = A_d \oplus R. \quad (14)$$

If we choose r to be the grid spacing in the x direction, then \tilde{A} is as shown in the figure. The dilation sweeps each dixel by $r/2$ to the left and right. The occupied intervals in each dixel, originally 0-width line segments, are widened by this procedure into rectangular blocks that span the full width of the grid cell occupied by the dixel. This closes the gaps between dexels and widens the object by $r/2$ on each side.

We chose to define the individual blocks within A_d such that the blocks coincide with the intersection of grid lines with A , as described in Section 2.2. That is, the boundaries of the blocks in A_d coincide with the boundaries of A . With this choice $A_d \subset A$. Dilation is increasing with respect to both its arguments. See, e.g., Proposition 12 and Corollary 13 in Ref. [25], which says that

$$A \subset B \text{ implies } A \oplus D \subset B \oplus D. \quad (15)$$

(Dilation is commutative, $A \oplus B = B \oplus A$, so the increasing property stated here for the first argument applies equally well to the second.) Since $A_d \subset A$ this proposition means in the present case that $A_d \oplus R = \tilde{A} \subset A \oplus R$. Thus \tilde{A} is bounded above by an object that differs from A only by something on the order of the grid size. It is similarly bounded below. Since the blocks in A_d terminate exactly on

the boundary of A , the same argument can be made with respect to the complements of \tilde{A} and A , leading to the result that $\tilde{A}^c \subset A^c \oplus R$. Since R is symmetrical ($R = -R$), Eq. (13) then implies $\tilde{A}^c \subset (A \ominus R)^c$, or $\tilde{A} \supset A \ominus R$. The upper and lower bounds are summarized as

$$A \ominus R \subset \tilde{A} \subset A \oplus R. \quad (16)$$

Now suppose we choose finer and finer grid spacings for our description. As $r \rightarrow 0$ the horizontal rod, R , approaches a single point, $\{0\}$, so $A \oplus R$ and $A \ominus R$ both tend to A . \tilde{A} is squeezed between outer and inner bounds that in this limit both approach the same value (A), and therefore $\tilde{A} \rightarrow A$. It is in this sense that \tilde{A} is an approximation for A . It can be made as close to A as desired by choosing the grid spacing fine enough.

Similar relationships hold if we use these dixel-approximated sets to calculate objects that are interesting for SPM. Given a sample, S , and tip, P , the image is given by

$$I = S \oplus P. \quad (17)$$

Suppose we approximate S and P as dixel objects dilated with R as in Eq. (14), so that $S \ominus R \subset \tilde{S} \subset S \oplus R$ and $P \ominus R \subset \tilde{P} \subset P \oplus R$. We can then compute an approximate image according to

$$\tilde{I} = \tilde{S} \oplus \tilde{P}. \quad (18)$$

Because dilation is increasing with respect to both inputs, we can derive lower and upper bounds for \tilde{I} by replacing \tilde{S} and \tilde{P} in Eq. (18) with their lower and upper bounds

$$(S \ominus R) \oplus (P \ominus R) \subset \tilde{I} \subset [S \oplus R] \oplus [P \oplus R]. \quad (19)$$

The bounds on \tilde{I} are not as tight as those on A in Eq. (16). There are two appearances of R on each side instead of only one. This is owing to the fact that both inputs \tilde{S} and \tilde{P} were approximate.

The other calculated quantity of interest to us here is the reconstructed sample from the measured image, I , and a known tip shape P . This reconstruction is formed by erosion:

$$S_r = I \ominus P. \quad (20)$$

If we define the approximation $\tilde{S}_r = \tilde{I} \ominus \tilde{P}$ one can find upper and lower bounds similarly to the previous examples. The only difference is that $\tilde{I} \ominus \tilde{P}$ is increasing in \tilde{I} and decreasing in \tilde{P} , so the upper bound of \tilde{S}_r must be formed by using the upper bound of \tilde{I} and the lower bound of \tilde{P} while the opposite is done to form the lower bound. In this way one arrives at

$$(I \ominus R) \ominus (P \oplus R) \subset \tilde{S}_r \subset (I \oplus R) \ominus (P \ominus R). \quad (21)$$

Eqs. (19) and (21) behave similarly to Eq. (16) as r gets small, in the sense that the upper and lower bounds both approach the true value of the object we are approximating. In all cases the approximation can be made as close as one likes by choosing the grid spacing fine enough.

This discussion was limited to a 2D example for simplicity and ease of illustration. However, it is equally

applicable in 3D. The only difference is that instead of a horizontal rod, R must be a horizontal rectangular element with dimensions r_x and r_y equal to the 2D grid size.

As a consequence of this discussion, we see that the errors associated with the discreteness of the representation are initially of the order of the grid spacing. Errors may accumulate to a few times larger than this during calculations. One should choose a grid spacing small enough that the associated uncertainties are smaller than those associated with other sources of error. With this choice, the dixel-represented objects can for all practical purposes be substituted for the real ones.

5. Validation using test data

Note that in all the examples below, the simulated AFM images were obtained through the dixel dilation operation, as presented in this paper. All the reconstructed sample surfaces were obtained through the dixel erosion operation.

The data files in these examples are in the dixel representation. Even though the actual AFMs capable of bi-directional servo control and imaging undercut features may have the output format as (x, z) pairs, it is not difficult to convert such pairs into the dixel representation through various interpolation methods, such as linear interpolation.

5.1. Validation for surfaces without undercut features

We first show examples that demonstrate the consistency between the new dixel-based method and grayscale mathematical morphology-based method [8] for a set of surfaces without undercut features. A grayscale- or umbra-represented object is a special case of a dixel-represented one, in which the dixel at index (i, j) has only a single block, $(-\infty, h_{ij}]$ with h_{ij} the value of the pixel at index (i, j) . Consistency of the methods requires that given equivalent

inputs, the results will be representations of the identical output.

Example 1. Fig. 6 shows an SPM image simulation and surface reconstruction for a 2D profile based on the new software. The set of translated probes graphically illustrates the dilation and erosion relations among the true profile, SPM image, and reconstructed profile. The simulation is done through the dilation of the 2D profile with a parabolic tip. The reconstruction is obtained through the erosion of the simulated SPM image by the tip. The profile is a cross-section of a rough phosphor thin film (more in Example 3).

As expected [8], the reconstructed surface approximates and correctly bounds the specimen surface. Both the dilation and erosion results are identical to those from the grayscale morphology software.

Example 2. Fig. 7 shows an AFM image simulation and surface reconstruction for a spike surface. For this simple structure the dixel and grayscale software produced identical results.

Example 3. The structure labeled “sample” in Figs. 8 and 9 is actually an AFM image, available from a previous study [26], of a rough phosphor thin film. For the purpose of this test we pretend this represents the actual surface of a rough sample. In this way we generate a test in which the sample has a more realistic variety of structures than in the previous test. An image is simulated by dilating the sample with a parabolic tip, as shown. The sample is then reconstructed by erosion of the tip from the simulated image. The reconstructed surface differs from the original one due to information loss in the dilation process (a known phenomenon). The new dixel algorithms and the older grayscale algorithms produced identical results for the simulated image and reconstructed surfaces in this test.

Thus, the new mathematical morphology software based on dixel representation supports image simulation and

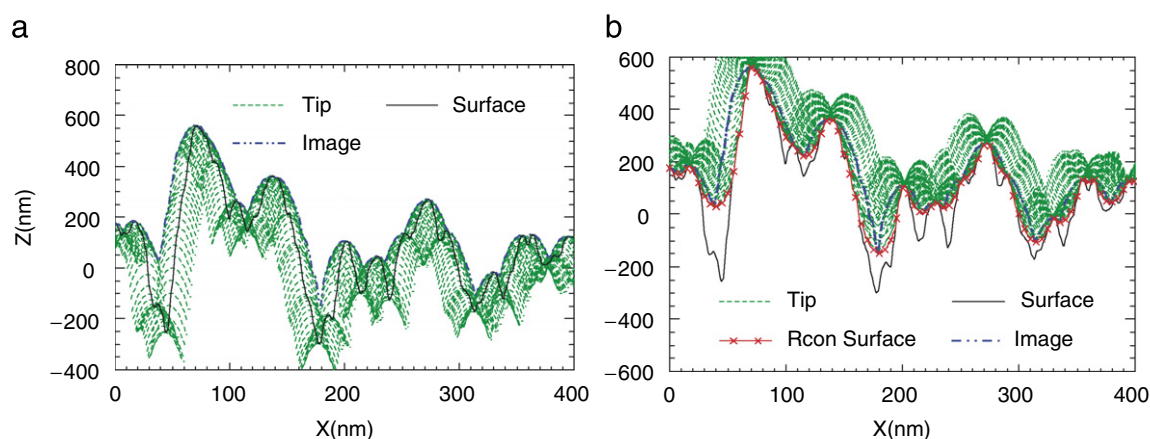


Fig. 6. SPM image simulation and surface reconstruction through morphological operations for a 2D profile. (a) SPM image formation through morphological dilation, (b) specimen surface reconstruction through morphological erosion.

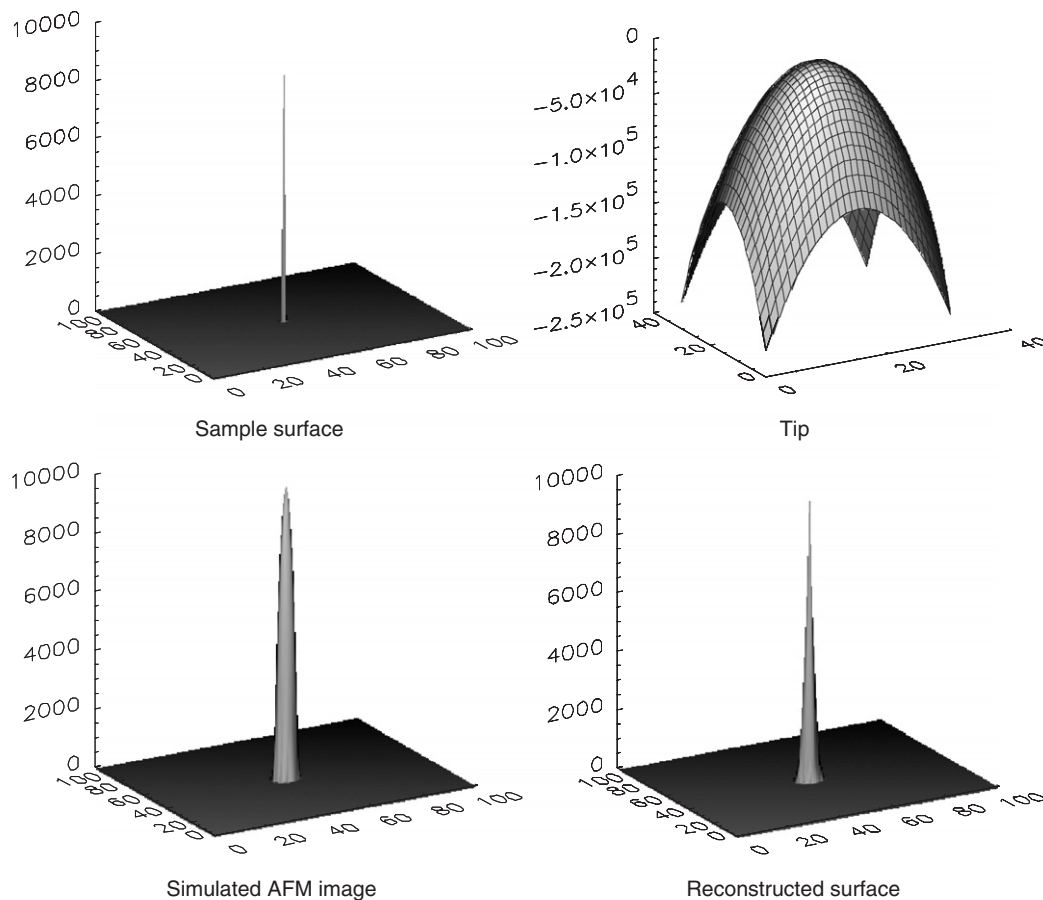


Fig. 7. AFM image simulation and surface reconstruction for a spike surface (Note: the tip is scaled differently from the surface to give a clear illustration of the tip shape).

surface reconstruction for structures without undercut features.

5.2. Validation for surfaces with undercut features

The dixel method is also capable of simulating the SPM imaging of undercut structures and capable of reconstructing the specimen surface of undercut shapes. Since this cannot be accomplished in grayscale morphology, the existing software cannot be used for validation as was done in the last section. Instead we use two other methods, one each in Examples 4 and 5.

Example 4. In Fig. 10 are pictures of image simulation and surface reconstruction for a profile with undercut areas. They are obtained through the dilation and erosion operations by a tip of undercut shape. As Fig. 10a demonstrates, the envelope of the translations of the reflected tip forms the SPM image. As shown in the figure, if dilation is correctly implemented the tip should just touch the simulated image and not go beyond it when the tip apex point is translated to any point along the undercut profile. Likewise, if erosion is properly implemented the tip should touch the reconstructed surface without penetrating beyond it when tip moves along the

simulated image. Satisfaction of these requirements was manually verified for a large number of points along rough and realistic profiles like the one shown. As expected, the reconstructed surface approximates and bounds the original surface.

Example 5. Fig. 11 shows an artificial sample surface, simulated AFM data through dilation with a spherical tip, and the reconstructed surface through erosion for a 3D structure with undercut features. Each 3D surface is shown in two views (x and y cross-sections) and a 3D rendering. To further validate the correctness of the dilation/erosion operation, cross-sectional examination is conducted. Fig. 12 shows a cross-section of the dilation/erosion operation for this 3D structure and the overlay of image, true surface and reconstructed surface. As the figure demonstrates, the probe at the image point just touches the surface but not coincide with the sample surface due to the finite dimension of the tip. However, the tip shape coincides exactly with part of the reconstructed surface. The artificial sample used in for this test was deliberately chosen to be simple—with cross-sections composed of straight lines meeting at sharp corners. For this kind of simple sample, the result of dilation with a spherical tip is easy to anticipate even without a dixel-based algorithm to

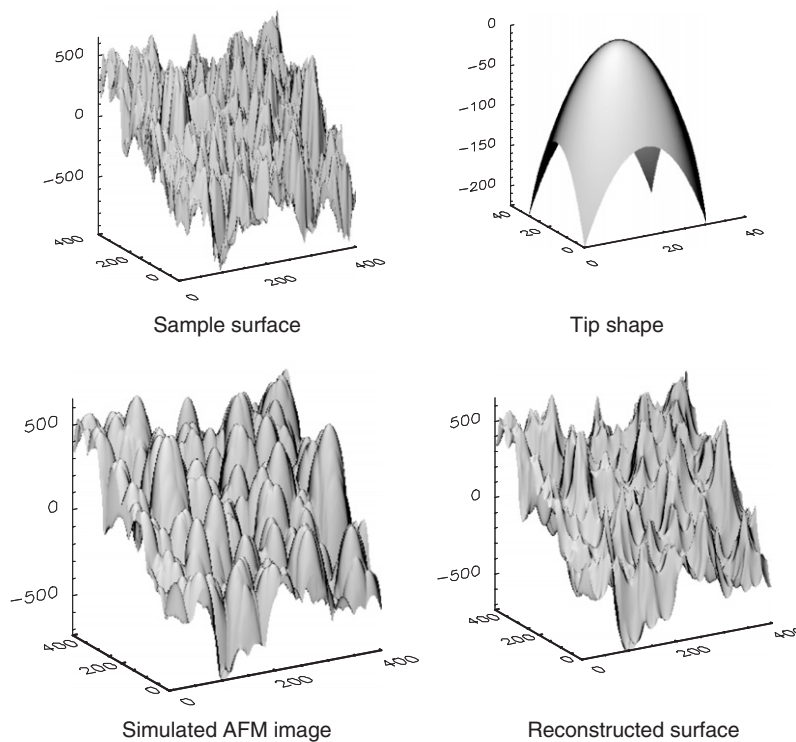


Fig. 8. AFM image simulation and surface reconstruction for a sample surface: surface view.

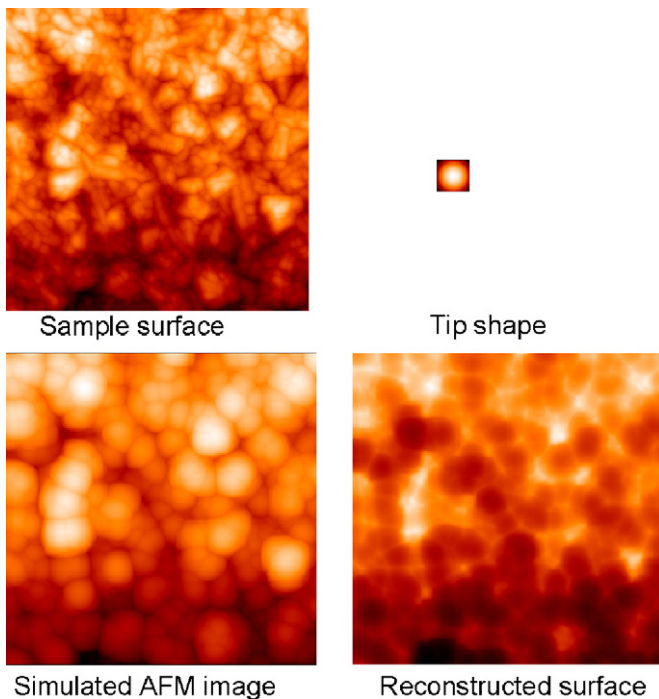


Fig. 9. AFM image simulation and surface reconstruction for a simple surface: top-down view.

compute it. Flat parts of the surface are moved by an amount that depends only upon their orientations and the radius of the spherical tip. Corners produce image areas

with radius equal to the tip radius. The dixel-computed image agrees with these expectations. The reconstructed surface approximates and correctly bounds the sample surface.

5.3. Algorithm efficiency

From the algorithm description in Section 3, it is clear that the dilation/erosion algorithms have complexity of

$$O\left(\sum_{\substack{i=1, m_1 \\ j=1, m_2}} K[i, j] * \sum_{\substack{u=1, n_1 \\ v=1, n_2}} W[u, v]\right),$$

where $K(i, j)$ and $W(u, v)$ give the number dixel blocks at indices (i, j) and (u, v) of the inputs.

When there is no undercut feature in the sample surface and the tip, that is, $K(i, j) = 1$ and $W(u, v) = 1$, the algorithm complexity reduces to

$$O(m_1 * m_2 * n_1 * n_2).$$

We therefore expect compute time of the new algorithms to be *linearly* proportional to the number of dixel blocks in the sample surface and in the AFM tip. The extra computing time needed to process a sample with undercut features compared to a similar sample without undercut features is linear in the number of extra dixel blocks needed for representing the undercut features.

Table 2 lists the computing time for the dilation and erosion operations in the above examples. The time shown

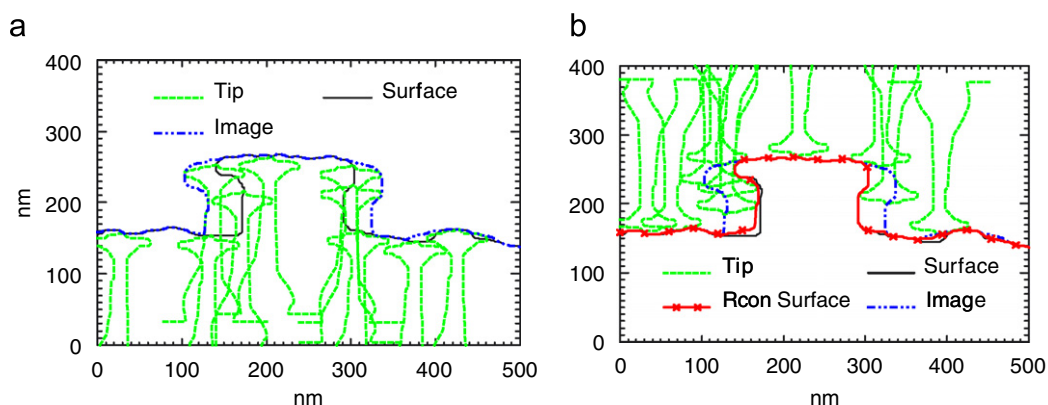


Fig. 10. Image simulation and surface reconstruction for a profile with undercut features. (a) Dilation in image simulation, (b) erosion in surface reconstruction.

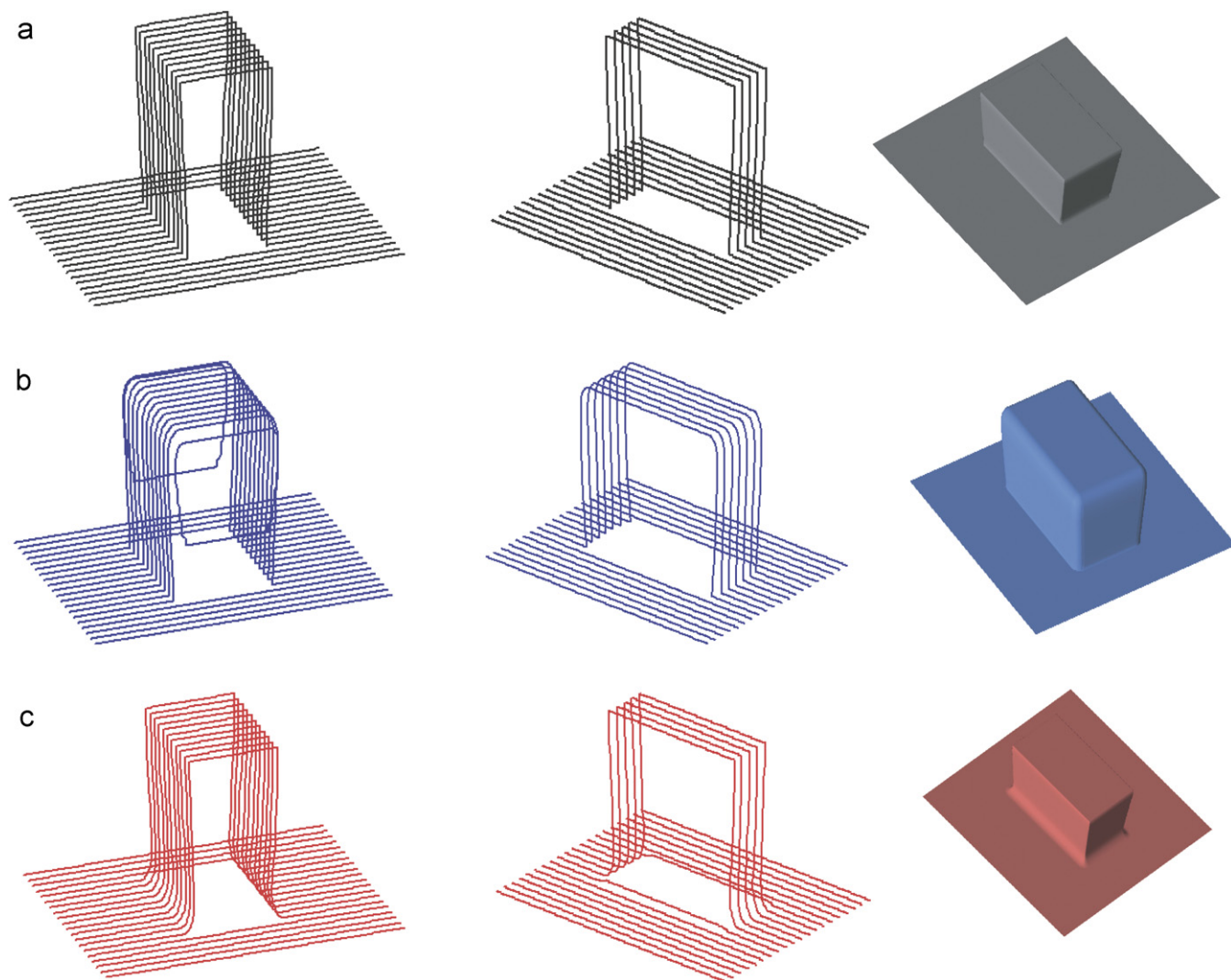


Fig. 11. Image simulation and surface reconstruction for a 3D structure with undercut surfaces, showing cross sections through x (left column) and y (middle column) and a solid rendering (right column). (a) Sample surface, (b) simulated AFM data, (c) reconstructed surface.

in the table is recorded from the tests on a PC (Dell Optilex GX520, Pentium® CPU3.40 GHz, 1.99 GB RAM).² The sample size and tip size are also shown in the table.

²Commercial equipment is identified in order to specify the measurement procedure. Such identification does not imply recommendation or endorsement by the National Institute of Standards and Technology, nor does it imply that the equipment identified is necessarily the best available for the purpose.

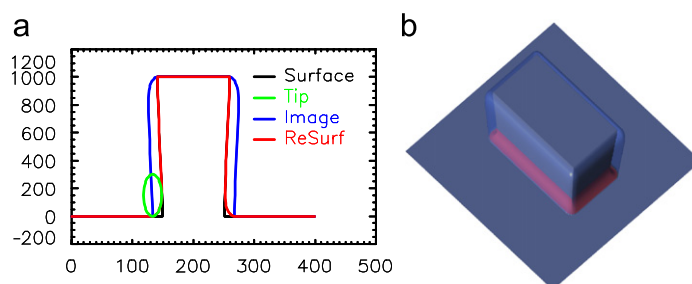


Fig. 12. A cross-section and an overlay of 3D structures in Fig. 11. (a) Cross-section of dilation and erosion operation, (b) overlay of surface, image, reconstructed surface.

Table 2
Execution time comparison

	Example 2		Example 3		Example 4	Example 5
Software	Dxel	Grayscale	Dxel	Grayscale	Dxel	Dxel
Sample size	101 * 101		400 * 400		500 * 1	400 * 400
Tip size	31 * 31		31 * 31		31 * 1	31 * 31
Dilation (s)	0.34	0.11	7.78	1.31	0.05	9.06
Erosion (s)	0.36	0.11	8.00	1.33	0.05	10.25

A comparison has been made between dixel-based and grayscale mathematical morphology software for structures without undercut features. The time for Example 1 is not shown here since its time lapse is very small (0.04 s) for both software. Our present dixel-based implementation is about 3 to 5 times slower than the grayscale morphology method. At least some part of this difference is the inevitable consequence of a more general procedure. In grayscale morphology only the upper of the two bounds in Eq. (7) need be calculated. The lower bound is always $-\infty$. Even when a dixel object consists of only dexels with a single block each, the dixel algorithm still must determine the lower boundary of this block–block dilation, since it can in general be other than $-\infty$. The dixel code must also check for the presence of additional blocks, even if in a particular case they do not happen to be present. Considering the simplicity of the grayscale block dilation (sum) and union (max) functions, this inevitable overhead might contribute a factor of 2 or 3 to execution time of the more general algorithm. This suggests that some other part of our observed speed difference might still be improved by more efficient implementation.

The run times for structures with undercut features as in Examples 4 and 5 are also shown in the table. The comparison between Examples 3 (without undercut) and 5 (with undercut) demonstrates that the existence of undercut features only contributes marginally to the dilation/erosion operation time for typical samples. This is because the number of undercut dexels is usually only a small proportion of the overall dexels in Example 5. In Example 5, there are total 165 760 blocks for the 400 * 400 grid in the dilation and 166 068 blocks for the 400 * 400 grid in erosion operation.

6. Conclusions

This paper presents a dixel computer representation and its algorithm implementation for SPM image simulation and surface reconstruction on general 3D structures. Experimental validation on both simulated and actual AFM data demonstrates that the dixel representation can efficiently simulate and reconstruct various 3D structures, including those with reentrant surfaces and undercut features.

Our contribution in this paper is threefold:

- A dixel-based object representation and its implementation for mathematical morphology are introduced. The representation provides an efficient and compact representation for general 3D structures, including undercut features.
- The implementation is complete in the sense that (a) any 3D object may be represented in a dixel form to any desired degree of accuracy, simply by choosing the resolution high enough, and (b) we implement all of the basic set operations—reflection, complement, union, intersection, subtraction, dilation, and erosion.
- We fulfill a need that has become increasingly important in semiconductor industry: how to reconstruct surfaces and simulate images of undercut features in SPM.

References

- [1] J.E. Griffith, D.A. Grigg, *J. Appl. Phys.* 74 (9) (1993) 83.
- [2] International Technology Roadmap for Semiconductors, Semiconductor Industry Association, San Jose, CA, 2003.
- [3] G. Dahlen, M. Osborn, N. Okulan, W. Foreman, A. Chand, J. Foucher, *J. Vac. Sci. Technol. B* 23 (6) (2005) 2297 (Also in Veeco Application Notes AN83-091404).
- [4] Y. Martin, H.K. Wickramasinghe, *Appl. Phys. Lett.* 64 (19) (1994) 2498.
- [5] T. Morrison, C. Marotta, *Micro Magazine*, August/September (2005).
- [6] R. Kneidler, S. Borodyansky, L. Vasilyev, D. Klyachko, A. Buxbaum, T. Morrison, *Proc. SPIE* 5567 (2004) 905.
- [7] G.S. Pingali, R. Jain, Restoration of scanning probe microscope images, in: *Proceedings of the IEEE Workshop on Applications of Computer Vision*, 1982, p. 282.
- [8] J.S. Villarrubia, *J. Res. Nat. Inst. Stand. Technol.* 102 (4) (1997) 425.
- [9] J.S. Villarrubia, *Surf. Sci.* 321 (1994) 287.

- [10] G. Varadhan, W. Robinett, D. Erie, R.M. Taylor II, Fast simulation of atomic-force-microscope imaging of atomic and polygonal surfaces using graphics hardware, in: *SPIE Conference on Visualization and Data Analysis*, 2002.
- [11] D. Keller, *Surf. Sci.* 253 (1991) 353.
- [12] N.G. Orji, R.G. Dixon, *Meas. Sci. Technol.* 18 (2007) 448.
- [13] N.G. Orji, R.G. Dixon, A. Martinez, B.D. Bunday, J.A. Allgair, T.V. Vorburger, *Journal of Micro/Nanolithography, MEMS and MOEMS*, in press.
- [14] G. Dahlen, M. Osborn, N.-C. Liu, R.W. Jain, W. Foreman, R. Osborne, *Proc. SPIE* 6152 (2006) 61522R-1.
- [15] A.A.G. Requicha, *Comput. Surv.* 12 (4) (1980) 438.
- [16] K.C. Hui, *Visual Comput.* 10 (1994) 306.
- [17] W.P. Wang, K.K. Wang, *IEEE Comput. Graphics Appl.* 6 (12) (1986) 8.
- [18] Y. Huang, J.H. Oliver, NC milling error assessment and tool path correction, in: *Computer Graphics Proceedings, Conference Proceedings July 24–19, 1994, (Proc. SIGGRAPH '94)*, 1994, pp. 287–294.
- [19] H. Voelcker, W. Hunt, The role of solid modeling in machining—process modeling and NC verification, *SAE Technical Paper* 810195, Warrendale, PA, 1981.
- [20] P. Brunet, I. Navazo, *ACM Trans. Graphics* 9 (1990) 170.
- [21] Y. Kawashima, K. Itoh, T. Ishida, S. Nonaka, K. Ejiri, *Visual Comput.* 7 (1991) 149.
- [22] T. Van Hook, Real Time Shaded NC Milling Display, *SIGGraph86*, vol. 20(4), 1986, pp. 15–20.
- [23] A. König, E. Gröller, Real time simulation and visualization of NC milling processes for inhomogeneous materials on low-end graphics hardware, in: F.-E. Wolters, N.M. Patrikalakis (Eds.), *Proceedings of CGI'98 (Computer Graphics International)*, IEEE Computer Society, Hannover, Germany, June 22–26, 1998, pp. 338–349.
- [24] E.E. Hartquist, J.P. Menon, K. Suresh, H.B. Voelcker, J. Zagajac, *Comput. Aided Des.* 31 (1999) 175.
- [25] R.M. Haralick, S.R. Sternberg, X. Zhuang, *IEEE Trans. Pattern Anal. Mach. Intell.* PAMI-9 (1987) 532.
- [26] R. Revay, J. Schneir, D. Brower, J. Villarrubia, J. Fu, J. Cline, T.J. Hsieh, W. Wong-Ng, A study of the surface texture of polycrystalline phosphor films using atomic force microscopy, in: K. Barmak, M.A. Parker, J.A. Floro, R. Sinclair, D.A. Smith (Eds.), *Materials Research Society Symposium Proceedings*, vol. 343, *Polycrystalline Thin Films: Structure, Texture, Properties, and Applications*.