Information Processing Letters 24 (1987) 369–375
North-Holland

6 April 1987

DIGITIZING THE PLANE WITH CELLS OF NONUNIFORM SIZE *

Hanan SAMET and Clifford A. SHAFFER

Computer Science Department and Center for Automation Research, University of Maryland, College Park, MD 20742, U.S.A.

Robert E. WEBBER

Computer Science Department, Rutgers University, Busch Campus, New Brunswick, NJ 08903, U.S.A.

Communicated by Alan Shaw
Received 12 February 1986
Revised 29 July 1986

A central problem in the field of geometric algorithms is to compare algorithms that process vector-type data specified in the continuous plane with algorithms that process raster-type data specified in the digitized plane. A set of criteria is proposed that a digitization of a collection of line segments should satisfy in order to be said to represent the same structure as their continuous plane counterpart. These criteria insure that the digitized collection has the same number of distinct features as the original data. In particular, attention is focussed on the correspondence of the precision with which the locations of endpoints on a continuous plane are specified to the number of grid cells in the appropriate digitized plane. It is shown that these criteria cannot be satisfied by grid cells of uniform size. The result leads to a reformulation of the digitization process that corresponds closely to a quadtree decomposition.

Keywords: Digital geometry, computational geometry, quadtree, computer graphics, polygonal map

1. Introduction

Algorithms for manipulating geometric data come in two flavors: (a) those that assume vector-type data which is specified in a continuous plane, and (b) those that assume raster-type data which is specified in a digitized plane. It is hard to compare these two types of algorithms because of the difficulty in establishing strong analogies between the geometries of the two planes in which their data is specified. In this paper we focus on the type of analogies that should hold between the continuous and digitized plane [10] representations of a two-dimensional geometric object. In particular, we restrict ourselves to a consideration of what it means to have a digitized representation of a collection of straight-line segments and isolated vertices. (Note that the endpoints of the line segments are also referred to as vertices.) These line segments are said to form a planar polygonal map in the sense that they only intersect at their endpoints.

In recent years there has been a considerable amount of interest in the use of hierarchical data structures for handling both vector and raster data. They have been used in applications in computer graphics, solid modeling, image processing, geographic information systems, etc. (see the survey of Samet [11] for more details). Typical hierarchical data structures include the quadtree [7] and the octree [5,6,8]. They recursively decompose objects until all of the parts are homogeneous. One of the deficiencies of such area decomposition rules is that they lead to many small parts in the neighborhood of the object boundary. In

* This research was supported by the National Science Foundation under Grant DCR-86-05557.

addition, if the object does not have rectilinear boundaries, then an exact representation is impossible.

In order to overcome the above deficiencies, alternative decomposition criteria have been investigated that are based on the notions of points, edges, and faces. For example, see the various PM quadtree solutions for two-dimensional data such as polygonal maps [15,9], and for three-dimensional data such as machine part diagrams [1,2,3]. In each of these approaches it is desirable to know how many levels of decomposition are necessary for an exact representation (not an approximation). In the rest of this paper we show that an exact representation is possible for two-dimensional data and derive the number of levels of decomposition that are necessary as a function of the precision with which the vertices of the polygonal map are specified. Our approach is from the standpoint of seeking an 'appropriate' digitization of a polygonal map. The extension of our results to three-dimensional data is straightforward.

Of course, even data specified on a continuous plane must be represented by coordinates of limited or fixed precision. Let d denote the number of bits used to represent the fixed-point coordinates of the endpoints of the line segments. We also associate a measure of accuracy with the digitized plane. Let g denote the number of bits required to specify the locations of the grid cells that compose the digitized plane, i.e., the digitized plane will consist of $2^g \times 2^g$ grid cells. Conceptually, we will view these grid cells as forming closed square regions that are referred to by the integer coordinates of their lower lefthand corner. In the rest of this paper we derive an upper bound on g as a function of d. Normally, a digitization process is defined that maps grid cells in the digitized plane into congruent square regions in the continuous plane. As will be demonstrated in the next section, this requires us to modify the digitization process to permit grid cells to be of varying sizes resulting in a decomposition similar to that imposed by a quadtree [11].

Our goal is to devise a digitized representation for a collection of line segments that is based on the notion that the digitized collection should have the same number of distinct features as the original data. Alternatively, with each grid cell we want to associate either a vertex or a portion of a single line segment. Recall that we assume that line segments do not intersect at any points other than their endpoints. The digitized representation of such a collection of line segments is termed a *well-formed digitization* and is defined as follows.

1.1. Definition. A *well-formed digitization* of a collection of line segments satisfies the following three criteria:

(1) Vertices with different coordinates do not occur in the same grid cell.

(2) A grid cell containing a portion of a line segment does not also contain a vertex that is not on that line segment.

(3) Except in the cell at which they meet, two different line segments must not occupy the same grid cell.

These criteria establish an analogy between points in the continuous plane and cells in the digitized plane. Criterion (1) indicates that, for a given accuracy, two distinct points will not map to the same cell. Criteria (1) and (2) imply that if a point and a line segment are disjoint in the continuous plane, then their digitizations are also disjoint. Criterion (3), which states that two line segments cannot share the same grid cell unless they share a vertex within that grid cell, has as an immediate consequence that if two line segments share two different grid cells, then they must share two distinct vertices—which in turn implies that the two lines are the same. This is equivalent to saying that if two Euclidean lines share two distinct vertices, then they are the same lines. Alternatively, these criteria can be summarized as follows: if two line segments share a cell in the digitized plane, then they intersect in the continuous plane in the region of the inverse image (with respect to the digitization process) of that cell.

It is interesting to note that criterion (3) is impossible to satisfy if the grid cells are not defined as closed regions in the topological sense. In particular, suppose two lines pass through a cell and meet at a vertex, say V, on the border of that cell such that V fails to be 'within' the cell because

the cell is open on that side. In such a case, criterion (3) cannot be satisfied no matter how small the cell is made. Of course, this is avoided at the cost of nonuniqueness in the sense that line segments that intersect a specific grid point can be associated with one of four different grid cells depending on their slope. Similarly, each segment of a line that coincides with the horizontal or vertical lines that are parallel to the x and y axes and that pass through grid points is associated with two grid cells.

2. Analysis

In the following sections we expand on the notion of a well-formed digitization and its effect on the interaction between g (the size of the grid cell coordinates) and d (the precision of the continuous plane coordinates).

2.1. The digitized plane with uniform cell size

If only the first two criteria need to be satisfied, then it is necessary to calculate how close a line segment connecting two points on a $2^d \times 2^d$ grid can lie to another point on the grid without actually touching that point. Note that the grid points all lie on the corner of grid cells.

2.1. Lemma. *For a $2^d \times 2^d$ grid with all grid points separated by* 1, *the minimum horizontal (or vertical) separation between a grid point and a line segment joining two other grid points satisfying criterion* (2) *is bounded from below by* $2^{-(d+1)}$.

Proof. This distance can be asymptotically attained by the following construction. Consider a subset of a $2^d \times 2^d$ grid as shown in Fig. 1. Let A be at (0, 0), B at (1, 0), C at $(0, 2^d - 1)$, and D at $(0, 2^d - 2)$. The distance from A to B and from C to D is 1, and the distance from A to C is $2^d - 1$. We claim that this represents a worst-case situation and, in particular, that the closest approach for this grid will be the distance from D to the line BC. A demonstration that this is indeed the worst-case situation can be found in [14]. □

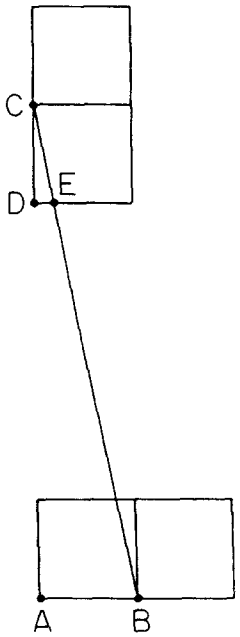

Fig. 1. Example illustrating the worst-case decomposition when only criteria (1) and (2) need to be satisfied.

Problems arise when we try to take into account criterion (3).

2.2. Theorem. *For a well-formed uniform digitization imposed on a $2^d \times 2^d$ grid with all grid points separated by* 1, g *is unbounded.*

Proof. Consider a subset of a $2^d \times 2^d$ grid as shown in Fig. 2. Let A be at (0, 0), B at $(1, 2^d - 1)$, and C at $(1, 2^d - 2)$. Since both lines AB and AC intersect the upper border of the cell containing A (i.e., at B′ and C′ respectively), they will both exist in the cell immediately above A, which contradicts criterion (3). Thus, we have to increase the resolution of the grid. However, as we increase the resolution, the size of the cell containing A shrinks and the intercepts of AB and AC with the upper border of that cell become even closer thereby forcing a further increase in resolution. It is clear that this increase in resolution can continue indefinitely without ever satisfying criterion (3) since a segment of the upper border of a cell is always smaller than the entire upper border of the cell. □

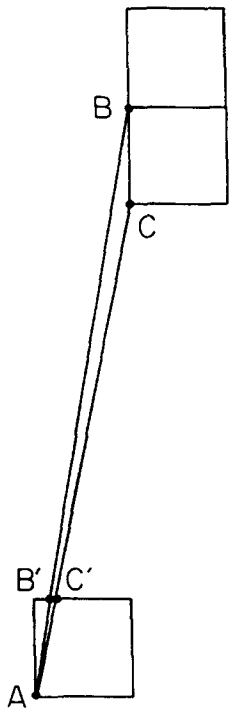

Fig. 2. Example illustrating the problems that arise when attempting to satisfy criteria (1)–(3) with a grid of uniform cell sizes.

This leaves us in the position of having three reasonable criteria for an appropriate digitization of a polygonal map that are inconsistent with the traditional definition of a digitized plane. In particular, note that our poblems with criterion (3) follow immediately from the expectation that each cell of the digitized plane be the same size. Instead of abandoning the criteria that forced the digitized features to correspond to the features of the continuous object that they represent (i.e., criteria (1)–(3)), we propose to permit the cells of the digitized plane to have different sizes. Thus, we replace the traditional digitization process with an adaptive digitization process.

The adaptive digitization process creates a collection of grid cells that are collectively termed the *adaptive digitized plane*. Initially, the adaptive process represents the entire plane by one grid cell. Each grid cell that fails to satisfy criteria (1), (2), and (3), is decomposed into four subcells (each of width one half of the width of the original cell). This process results in an arrangement of grid cell sizes that adapts to the structure being digitized. Such decomposition processes result in variants of the quadtree data structure. In particular, criteria (1)–(3) above correspond to the three properties that define the PM_1 quadtree [15]. An example of a polygonal map and its corresponding PM_1 quadtree is shown in Fig. 3. The PM_1 quadtree assumes a $2^5 \times 2^5$ grid.

In the next section we show that the 'quadtree'-style digitization approach is consistent with criteria (1)–(3). We also derive an upper bound on the corresponding value of g as a function of d. Note that, in terms of the quadtree data structure, g corresponds to the maximum depth of the tree.

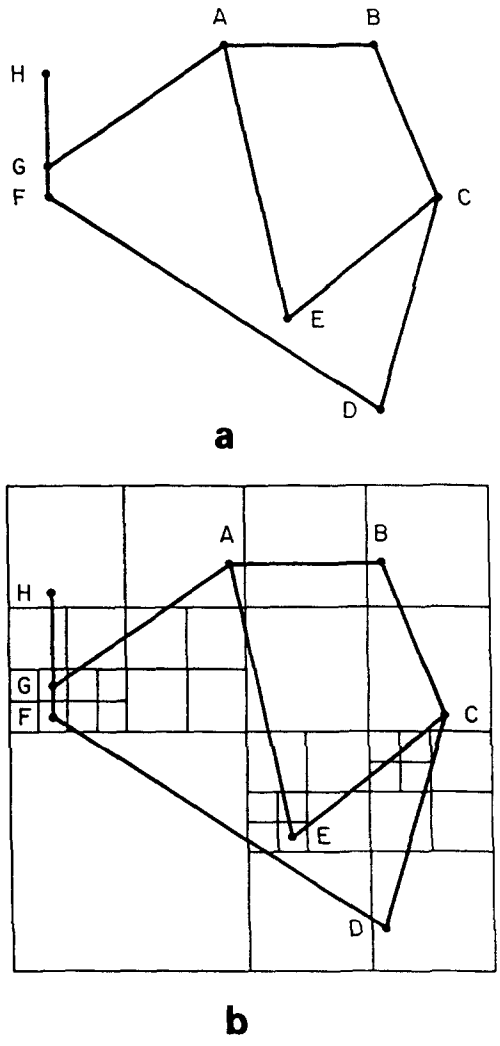

Fig. 3. (a) Example polygonal map and (b) its corresponding PM_1 quadtree embedded in a $2^5 \times 2^5$ grid.

2.2. The adaptive digitized plane

The situation that leads to the maximum value of g for the adaptive digitization process as a function of d is derived in the following manner. Observe that all the points that can be specified with d bits of precision form a $2^d \times 2^d$ grid. Recall that criterion (3) of the digitization criteria stipulates that no cell can contain two line segments unless they meet at a common vertex in the region of that cell. Thus, the problem reduces to how small a cell can be and still fail to satisfy this criterion. This situation is at its worst in the cells immediately adjacent to a cell containing a vertex. Note that the width of the smallest cell that fails to separate two lines originating at a vertex within a neighboring cell decreases monotonically with the angle between the two lines. Actually, we are not looking for the smallest angle between two line segments that emanate from the lower lefthand corner of a cell of a given size. Instead, we are seeking the minimum horizontal (or vertical) separation between these two line segments, i.e., they subtend the smallest portion of a particular side of a cell.

2.3. Lemma. *Let* P *be a grid point in a* $2^d \times 2^d$ *grid with all grid points separated by* 1. *Let* w *be the width of* P*'s grid cell. Let* ℓ_1 *and* ℓ_2 *be two lines that emanate from* P *and that exit from the same side of* P*'s grid cell. The minimum horizontal or vertical separation between* ℓ_1 *and* ℓ_2 *satisfying criterion* (3) *is bounded from below by* $w/((2^d - 1) \times (2^d - 2))$.

Proof. From Fig. 2 we observe that the closest approach between two intercepts, say B′ and C′, of line segments with the border of a cell of a grid point occurs when the cell has its lower left corner A at (0, 0) and the line segments extend from A to B at $(1, 2^d - 1)$ and C at $(1, 2^d - 2)$. An argument based on similar triangles demonstrating that this is indeed the worst-case situation is given in [14]. □

We now turn to the question of what is the worst-case value of g as a function of d when all three criteria are met.

2.4. Theorem. *The maximum level* g *needed by a well-formed adaptive digitization of a polygonal map whose vertices lie on a* $2^d \times 2^d$ *grid is bounded from above by* $4d + 1$.

Proof. Our proof develops from an ordered consideration of the impact of the various criteria for a well-formed adaptive digitization on the worst-case value of g. Criterion (1) has the least impact on the value of g. Since all vertices are restricted to lie on the points of a $2^d \times 2^d$ grid, criterion (1) will be satisfied by any value $g \geqslant d + 1$. The bound is $d + 1$ instead of d because we assume that each cell of an adaptive digitization has a closed boundary; hence, at level $g = d$, two neighboring grid points would be corners of a common cell.

As analyzed in Lemma 2.1, criterion (2) is independent of, and more restrictive than, criterion (1). By 'independent of' we mean that there is a worst-case map that attains its worst cases with respect to criteria (1) and (2) simultaneously. Thus, criterion (2) raises the upper bound on g to $2d + 1$.

Criterion (3) can be thought of as being applied after the adaptive digitization process has proceeded far enough to satisfy criteria (1) and (2). Note that any further decomposition of the adaptive digitization will continue to satisfy criteria (1) and (2). For a given cell c, in order for criterion (3) to be relevant, at least two line segments must intersect cell c without intersecting within cell c. If the line segments do not intersect at all, then their closest approach would be between the endpoint of one line segment and some portion of the other line segment. This situation has been analyzed with respect to criterion (2). Thus, although criterion (3) might force further decomposition in the neighborhood of nonintersecting line segments, the worst-case situation for nonintersecting line segments would be no worse than the criterion (2) bound.

Therefore, any increase in the upper bound on g due to criterion (3) must be the result of the occurrence of line segments with a common vertex that cause a grid cell to contain two line segments that are closer than the closest approach between any vertex and a line segment not containing that vertex. Let c be the cell of smallest width, say w, resulting from criteria (1) and (2) and let T be a

grid point on the lower lefthand corner of that cell. The only way to create a cell smaller than the one containing T is to have two or more line segments pass through one of the sides of the cell. Once these line segments exit the cell c, criterion (3) requires that they occupy separate grid cells. The analysis of criterion (3) performed in Lemma 2.3 did not set an absolute upperbound on g. Instead, it indicated a minimum separation of line segments, t, relative to the width of the cell containing the vertex from which the lines emanated. Applying Lemmas 2.1 and 2.3, we have that $2^{-(d+1)}$ is a lower bound on w, resulting in a minimum cell width, t, of $2^{-(3d+1)}$. Since we assume that all sides of the cell are closed, we need to halve this minimum cell width to insure that a cell containing a single line segment does not even contain a corner point of another line segment. Thus, the minimum cell width is $2^{-(3d+2)}$ which means that $4d + 2$ is an upper bound on g. (Recall that we started with a $2^d \times 2^d$ grid with all grid points separated by 1.) Also note that no further interactions are possible between criteria (1)–(3) that could force t to be smaller. □

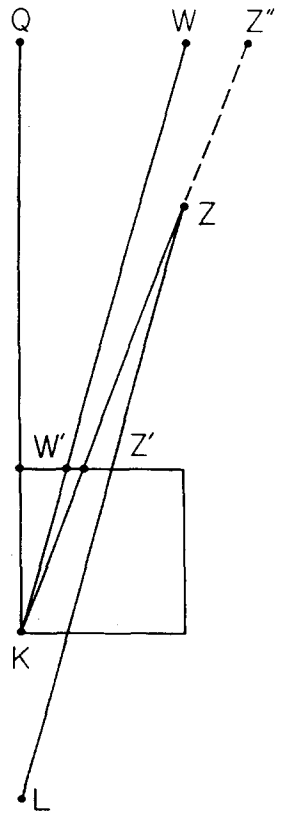

Fig. 4. Example polygonal map with a value of grid depth g very close to the worst case.

The bound obtained in Theorem 2.4 is sufficient to show the superiority of the adaptive digitization process over the uniform digitization process with respect to the representation of 'features' in the continuous plane (as defined by criteria (1)–(3)). Recall that there was no bound in the case of uniform digitization (as discussed in Section 2.1). An interesting question, although irrelevant to the central thesis of this paper, is whether or not the bound obtained in Theorem 2.4 is attainable. While the bounds dictated by criteria (2) and (3) are individually attainable (see the proofs of Lemmas 2.1 and 2.3), the constructions used to derive them were mutually incompatible. However, observe that the digitization of the lines WK, KZ, and ZL in Fig. 4 asymptotically approaches the bound of Theorem 2.4. In particular, W is at $(1, 2^d - 1)$, K is at (0, 1), Z is at $(1, 2^d - 2)$, and L is at (0, 0). For a more detailed analysis of this construction, see [14].

3. Applications

Our results are useful when implementing quadtrees without pointers, e.g., a linear quadtree [4]. In this case, the quadtree is treated as a collection of its leaf nodes so that each leaf node is represented by a pair of numbers (termed a locational code) that corresponds to the level of the block and a concatenation of base 4 digits corresponding to directional codes (i.e., NW, NE, SW, and SE) that locate the node along a path from the root of the quadtree. The value of g indicates the maximum number of bits necessary to specify the locational code. For example, the *segment quadtree* [13] is a scheme based on the linear quadtree for storing a PM_1 quadtree and its maximum depth is given by g as a function of d, the precision with which the vertices have been specified.

Of course, many polygonal maps can be digitized meeting our three criteria without requiring g to be as large as $4d + 2$. For example, Fig. 5 is a roadmap from a cartographic database with which

Fig. 5. The roadmap.

we have been working [12]. Its vertices rest on a 512×512 grid and thus each vertex would require 18 bits of information (two 9-bit coordinates) to represent its address. The map has 685 vertices and 764 line segments. For the adaptive digitization described above, g had a value of 12 whereas our analysis indicated that g was bounded from above by 38. Incidentally, the PM_1 quadtree corresponding to this adaptive digitization required 2701 cells.

The analysis performed here is also of use in computer graphics and solid modeling. In particular, it enables us to model arbitrary polyhedral solids in an exact manner. The extension of our analysis to three dimensions is straightforward. This impacts the use of octree variants such as those described in [1,2,3].

Acknowledgment

We would like to thank Mike Dillencourt for his valuable comments and criticisms.

References

[1] D. Ayala, P. Brunet, R. Juan and I. Navazo, Object representation by means of nonminimal division quadtrees and octrees, ACM Trans. Graphics 4 (1) (1985) 41–59.

[2] I. Carlbom, I. Chakravarty and D. Vanderschel, A hierarchical data structure for representing the spatial decomposition of 3-D objects, IEEE Comput. Graphics and Appl. 5 (4) (1985) 24–31.

[3] K. Fujimura and T.L. Kunii, A hierarchical space indexing method, Proc. Computer Graphics '85, Tokyo (1985) T1-4, 1–14.

[4] I. Gargantini, An effective way to represent quadtrees, Comm. ACM 25 (12) (1982) 905–910.

[5] G.M. Hunter, Efficient computation and data structures for graphics, Ph.D. Dissertation, Dept. of Electrical Engineering and Computer Science, Princeton Univ., Princeton, NJ, 1978.

[6] C.L. Jackins and S.L. Tanimoto, Oct-trees and their use in representing three-dimensional objects, Comput. Graphics and Image Process. 14 (3) (1980) 249–270.

[7] A. Klinger, Patterns and search statistics, in: J.S. Rustagi, ed., Optimizing Methods in Statistics (Academic Press, New York, 1971) 303–337.

[8] D. Meagher, Geometric modeling using octree encoding, Comput. Graphics and Image Process. 19 (2) (1982) 129–147.

[9] R.C. Nelson and H. Samet, A consistent hierarchical representation for vector data, Comput. Graphics 20 (3) (1986); also: Proc. SIGGRAPH '86 Conf., Dallas, August 1986.

[10] A. Rosenfeld and A.C. Kak, Digital Picture Processing (Academic Press, New York, 2nd ed., 1982).

[11] H. Samet, The quadtree and related hierarchical data structures, ACM Comput. Surveys 16 (2) (1984) 187–260.

[12] H. Samet, A. Rosenfeld, C.A. Shaffer and R.E. Webber, A geographic information system using quadtrees, Pattern Recognition 17 (6) (1984) 647–656.

[13] H. Samet, C.A. Shaffer and R.E. Webber, The segment quadtree: A linear quadtree-based representation for linear features, Proc. Computer Vision and Pattern Recognition '85, San Francisco (1985) 385–389.

[14] H. Samet, C.A. Shaffer and R.E. Webber, Digitizing the plane with cells of non-uniform size, Tech. Rept. TR-1619, Dept. of Computer Science, Univ. of Maryland, MD, January 1986.

[15] H. Samet and R.E. Webber, Storing a collection of polygons using quadtrees, ACM Trans. Graphics 4 (3) (1985) 182–222.