ELSEVIER

ISPRS Journal of Photogrammetry & Remote Sensing 55 (2000) 139–149

PHOTOGRAMMETRY & REMOTE SENSING

www.elsevier.nl/locate/isprsjprs

Taxonomy of space tessellation

Y.C. Lee [a,*], Z.L. Li [b], Y.L. Li [b]

[a] *Department of Geodesy and Geomatics Engineering, University of New Brunswick, Fredericton, N.B., Canada*
[b] *Department of Land Surveying and Geo-Informatics, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong, People's Republic of China*

Received 15 April 1999; accepted 4 March 2000

Abstract

When we map an area or create a digital database for it, the first task is often to partition the space into smaller units. There are traditionally two methods of partitioning: vector and raster. A vector partition delineates the boundary of features by polylines while a raster partition subdivides the space into a regular matrix of square or rectangular pixels. These two are complementary methods of subdividing the space either by features or by unconstrained space cells. In the third dimension, they are extended to polyhedra and voxels, respectively. We will argue in this paper that the terms "vector" and "raster" cannot describe all cases of tessellation. With advances in data modelling, variations of the two traditional methods have been developed, such as the representation of a feature by pixels and not by polylines. At present, there is a lack of systematic terminology to describe the various methods of tessellation. In this paper, we will propose a taxonomy for three-dimensional space tessellation. Its essential feature is to distinguish between abstract concepts of tessellation and their encoding methods. We recognise that tessellation of geographic space is carried out in different stages with increasingly precise mathematical meaning. This provides us with an insight into the process of spatial tessellation and a model to systematically describe the various structures. These concepts could form a basis for spatial data models.

Keywords: spatial tessellation; raster and vector structures; spatial data models

1. Introduction to space tessellation

The creation of geographic databases, similar to the creation of analogue maps, involves a series of abstraction and an implementation. At the abstraction levels, we extract features and their characteristics from the world and give these features increasingly more concrete geometric and semantic meaning. An implementation (or encoding) puts mathematically well-defined features into a form that could be stored in and manipulated by a computer.

At an abstract level, when we perceive a space (the world, for example) to be composed of countries of the world, we have partitioned the space into countries. We will call partitioning the space into non-overlapping cells a planar tessellation or simply *tessellation* (Fig. 1). In this paper, we will not con-

* Corresponding author.
E-mail address: yclee@unb.ca (Y.C. Lee).

0924-2716/00/$ - see front matter
PII: S0924-2716(00)00015-0

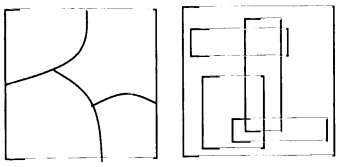


Fig. 1. A planar and non-planar tessellation of space.

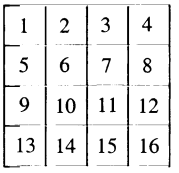


Fig. 3. Space-primary tessellation of space.

sider non-planar tessellation of space such as R-trees (Guttman, 1984).

At the implementation level, the purpose of tessellation is to create "buckets" to contain data in a computer. After partitioning the space into geographic features, we can assign each feature a data record (bucket) in a computer file to store its attributes. We will call those tessellation that follow the outline of features a *feature-primary tessellation* (FPT). Some tessellations do not follow the geographic features — the partitioning of space into regular squares, called pixels, is an example. We will call that tessellation that uses somewhat arbitrary decompositions a *space-primary tessellation* (SPT) of space. The FPT and SPT result in *feature cells* and *space cells*, respectively.

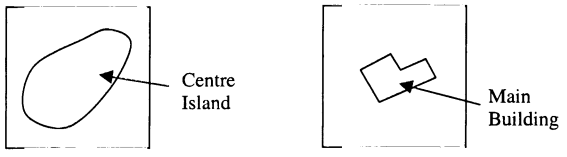
It is important, therefore, to note that in either tessellation, both the space and its attributes are stored. The main difference between FPT and SPT is that in the first case, emphasis is placed on the existence of features. There is a one-to-one relationship between a feature cell and a feature, and the space occupied by the feature will become an attribute of the feature (Fig. 2).

In the SPT case, emphasis is placed on the existence of space, and feature is an attribute of it (Fig. 3). The fact that a space cell falls upon a feature is more coincidental than intentional. If a space cell is large, it could fall upon many features. Even when the space cells are very small, the same tessellation can be placed over different themes of data, causing each cell to be associated with multiple features, one from each theme. On the other hand, a feature could be decomposed into many space cells. Hence, in a SPT, there is a many-to-many relationship between space cells and features (Fig. 3).

The major advantage of a FPT is that it can be implemented in such a way that everything belonging to a feature can be encapsulated into a computer record. This makes it easier to retrieve all attributes belonging to a feature and easier to encode explicit relationship (particularly topological ones) among the features. A SPT, on the other hand, highlights the space and downplays the boundaries between features. This makes it more convenient for handling spatial propagation and dispersion problems where the topology among the space cells is more useful than the topology among the feature cells.

It would appear that the vector model is equivalent to the FPT and the raster model is equivalent to the SPT. We will argue in this paper that this is not always the case. The problem with the use of the terms "vector" and "raster" is the confusion between a conceptual tessellation and its implementation in a computer. At the conceptual level, a tessellation is just a partition of space into smaller cells. Vector and raster, on the other hand, are methods of implementing these cells in a database.

In this paper, we will introduce a taxonomy of tessellation that will clarify a number of confusing concepts. Such taxonomy is particularly useful when a clear definition of data models would significantly enhance the sharing and interchange of spatial data. Peuquet (1984) has proposed a conceptualisation of spatial models. We will address a similar problem here from a tessellation point of view.



Feature Type	Feature Name	Space
Island	Centre Island	$p_1,p_2,p_3,\ldots,p_m$
Building	Main Building	$p_1,p_2,p_3,\ldots,p_n$

Fig. 2. Feature-primary tessellation of space.

It should be noted that a tessellation, such as the rectangular grid pattern on a topographic map, could be used purely for geo-referencing purposes and is never related to features on the ground. We do not consider tessellations in this context. However, we do consider tessellations for spatial indexing purposes, because the cells so created will be associated with features in the database.

In the next section, we will provide a definition of space. We will then describe the various forms of feature abstraction in the different spaces used by GIS practitioners in creating a spatial database. In Section 4, we will address the encoding of these abstractions at the implementation level.

2. Space

An understanding of space is vital to the study of space tessellation. The subject has been addressed by Cartell (1991), and the difficulties in defining space have been elaborated in Worboys (1995). In broad terms, space is composed of a set of elements called the universe and a set of conditions these elements must satisfy. These conditions come in various forms, including unary and binary relations defined over the universe.

A *vector space* consists of a set of points (the universe) and the condition that they follow the rules of vector addition and scalar multiplication of vectors. We refer to a vector space as an *Euclidean space* when the rules of Euclidean geometry are also observed. For a *topological space*, we again have a set of points forming the universe. The condition here is that they must observe some rules regarding neighbourhood, a concept central to the study of topology that is elaborated in Section 3.3.2.

When we talk about space in a Geographic Information Systems (GIS) context, we often apply a very broad definition that includes characteristics from Euclidean, topological, and other relevant spaces. We believe that an intuitive understanding of the characteristics of these spaces is enough to address the spatial tessellation problem without having to formally define the spaces.

In this paper, we refer to the elements of a space from a geometric point of view. We have hence used terms such as points, lines, faces, and volumes instead of equivalent terms in combinatorial topology because algebraic operations are not our prime concern here.

3. Geographic feature abstraction and tessellation

We recognise three types of abstraction regarding space tessellation: real-world, database, and mathematical (Fig. 4). The three types correspond to the three steps in the abstraction process. However, it should be stressed here that we do not always think about tessellation in a strict top–down order.

In the first step, the space we operate on is the real world, and the main concern is to identify and delineate features of interest to us. In the second step, we operate on the database space in the sense that the storage and management of features in the database are the main incentives for tessellation. In other words, we identify the shape of the data "buckets", which could be different from the real-world tessellation of features. In both steps, the geometric shapes of the tessellation cells are not well defined. It would be useful to make a distinction between a feature and an entity. When tessellating the real-world space, we are delineating *features*. When tessellating the database space, we are delineating *entities* that could be different from the features.

In the third step, we operate on a mathematical space, usually Euclidean. This is where the tessellation cells are given exact geometric meanings. After the third step of abstraction, we are ready to encode the tessellation for computer storage and access. We will elaborate on these different types of abstraction in the following sections.

3.1. Abstraction on the real-world space

In this first step of tessellation, we identify and delineate geographic features of interest to users of

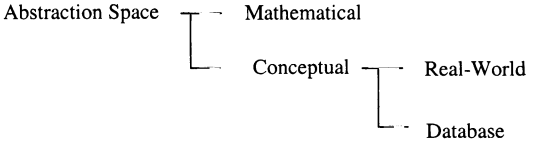


Fig. 4. A hierarchy of abstraction spaces.

the database. An important consideration of this abstraction is the extent and shape of the features.

When we conceptualise the world in terms of points, lines, areas, and solids, we think of these "geometric" forms in a very loose way, so loose that they are neither geometric nor algebraic. That is to say, they are not mathematically well defined to allow us to perform geometric or algebraic calculations on them. Even without the mathematical definition, this highly conceptualised tessellation is useful because it forms the basis for the next abstraction. It provides the definition of features without which the other abstractions would be meaningless.

3.2. Abstraction on the database space

We will eventually approximate the features mathematically so that we can operate on them. Before we do that, however, we need to consider the database space (analogous to the sheet of paper on which conventional maps are being produced) to accommodate these features. This space has a dimension, again in a loose sense when we consider it at a conceptual level. There is a need to break this space into non-overlapping cells for the convenience of management. We call this a *tessellation* of the database space.

3.2.1. A classification of database space tessellation

There are two types of tessellation for this purpose: one matches the geographic features we have identified in the first abstraction while the other is composed of space cells often arbitrarily defined. The first type closely matches the technique traditionally used in producing line maps where the extent of a geographic feature is delineated by lines. The attributes of these features are then represented by annotations, line patterns, or area patterns. The second tessellation has been used in producing orthophotos or similar image-based products. In this case, the image is broken into arbitrary units such as light-sensitive grains on a photograph or screen dots on a printed version of it. The attributes of these space cells are then represented by size or colour.

We sometimes regard point features and line features as degenerate cases of area features. This is in conformance with the concept that all geographic features in the real world occupy space, and they are represented as points and lines only due to limitations of scale. In Section 3.3.2, we will demonstrate the topological ambiguity this could cause.

We have identified two broad classes of space tessellation previously, the FPT and SPT. The FPT uses features as the primary units of representation (the entities are features) while the SPT uses space cells as the primary units (the entities are not features). In the FPT case, the tessellation can be hierarchical in that the features are nested within each other. An example of this is a tessellation of administrative boundaries for different levels of government.

In a SPT, not all space cells are arbitrary but some are constructed under the constraints of features. Take for example a Triangulated Irregular Network (TIN). The triangular patches of a TIN do not represent real world features per se, but they are constructed with reference to spot heights. Another example is the dual of Delaunay triangulation, the Voronoi polygons, which are constructed from point or line features.

In Fig. 5, the different types of database space tessellation are shown. We will elaborate on the constrained and unconstrained SPT in the next two sections.

3.2.2. Unconstrained SPT

An unconstrained SPT produces space cells independent of the location and shape of the features,

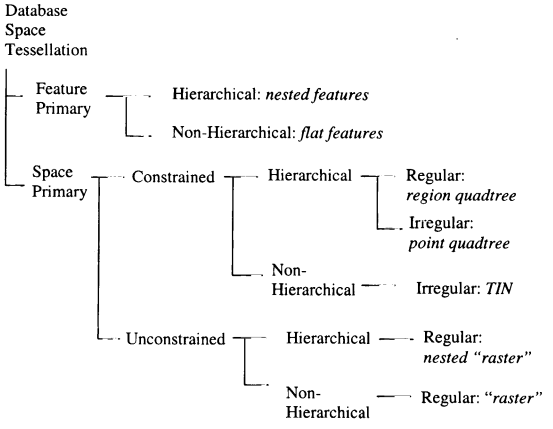


Fig. 5. A classification of tessellation on the database space.

although the size of each cell (the resolution) is usually adjusted to capture the smallest detail. The tessellation can use hierarchical or non-hierarchical decomposition. In a hierarchical decomposition, a space cell would be subdivided into progressively smaller cells. In the unconstrained case, all cells are regular (Fig. 5) meaning that they are identical regular polygons each with sides of the same length and the same interior angles. A very popular choice is the use of square space cells similar to pixels in a raster image. Other examples are triangular cells on a sphere. A hierarchical tessellation for an unconstrained SPT could be used for a multi-resolution representation of a region.

A taxonomy of regular tessellation suggested by Bell et al. (1983) used a notation that indicates the order of each vertex of the basic polygonal cell traversed in a certain orientation. An interior polygonal cell is used to avoid degenerate cases around the boundary. For example, the regular tessellation based on a square or rectangle is denoted as [4.4.4.4] or [4^4] (see Fig. 6a) and one based on a triangle is denoted as [6^3] (see Fig. 6b). It can be seen that in this notation, the sum of the powers indicates the number of sides of an interior cell. The tessellation in Fig. 6c is denoted as [4.8^2] and that in Fig. 6d is denoted as [4^3]. We will call this the *pattern code* of a regular tessellation.

The use of irregular cells is not practical for the unconstrained case. Irregular cells could result, however, from transformation of a regular tessellation. For example, when the regular tessellation of a remote sensing imagery has been geo-referenced, the square cells could become irregular quadrilaterals (Fig. 7). In our classification, we do not consider geometric transformations (or distortions) of a regular tessellation change the nature of the tessellation.

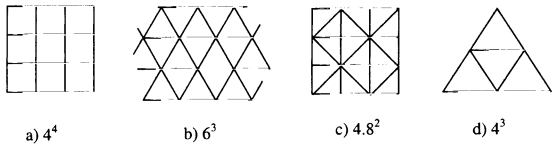


Fig. 6. Pattern codes for regular tessellation (see explanations in text).

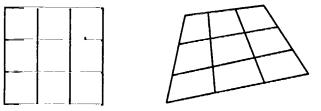


Fig. 7. Distortion due to geo-referencing of a regular tessellation.

3.2.3. *Constrained SPT*

A characteristic of a constrained SPT is that although the resulting space cells do not correspond to features in the real world, their construction cannot completely ignore the features. Very often, the size and shape of the space cells are adjusted locally to suit the distribution and configuration of the features.

In a hierarchical and regular decomposition, such as a region quadtree (Fig. 8a), a decomposition produces regular cells. In the irregular case, such as a point quadtree (Fig. 8b), a decomposition produces irregular cells. A variation of a quadtree decomposition based on triangles instead of squares, called a Quaternary Triangular Mesh (QTM), was proposed by Dutton (1990) (Fig. 8c). Although Dutton's tessellation was originally proposed to form a coordinate system on a sphere, an application not considered in this context, it could conceivably be used in place of ordinary quadtrees to represent features.

In a non-hierarchical decomposition, there will only be one level of subdivision. Examples of non-hierarchical irregular tessellation are flat TIN and Voronoi tessellation (Fig. 9). When the constraining features are distributed regularly, a constrained decomposition in the non-hierarchical case can produce regular cells. We regard that as a special case of an irregular tessellation. In general, there is no use for regular cells in this kind of tessellation.

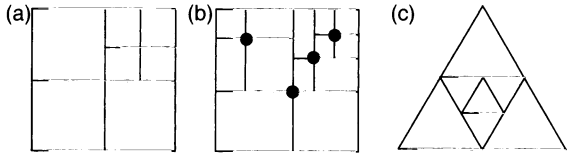
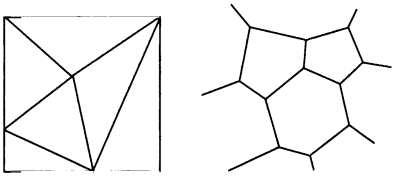


Fig. 8. Examples of hierarchical and constrained tessellation. Cases (a) and (c) are regular, case (b) irregular.



a) TIN b) Voronoi Tessellation

Fig. 9. Examples of non-hierarchical and constrained tessellation.

For constrained tessellation, the shape of the space cells is decided by the tessellation method. For instance, a triangulation method will produce triangular cells and a quadtree method will produce square, triangular (Dutton, 1990), or possibly cells of other shape. The size of the cells, on the other hand, is affected somehow by the location and distribution of geographic features. For a TIN, a non-hierarchical case, the triangular tessellation is constructed upon a collection of elevation or depth points. Although the triangular cells so obtained do not correspond to real-world features, their size is constrained by the elevation or depth points.

In the case of a region quadtree, a hierarchical case, the condition for the recursion to stop is when the quadrants are small enough to be homogeneous. There are variations of the region quadtree constrained not by area features but by point and line features.

For a point quadtree (Finkel and Bentley, 1974), also a hierarchical case, recursion stops when all points are processed (Fig. 8b). A point quadtree, similar to a TIN, adjusts the size of its quadrants according to the distribution of the points. Other than the region and point quadtrees, many other different versions have been proposed. A review of them can be found in Samet (1989).

3.3. Abstraction on the mathematical space

After we have formed a tessellation of the database space, we need to give the geometric entities (the features as well as the tessellation cells) more precise mathematical meanings. At the mathematical level, we need to identify the geometric space (usually of two- or three-dimensional Euclidean) and give geometrical meaning to the abstract points, lines, areas, and solids we have used so far. We need these mathematical elements so that we can manipulate the geographic features in the Euclidean space.

The abstraction of the real world is necessarily an approximation to it. This is particularly true in the mathematical sense. For one thing, real world "lines" (which are boundaries between areas) are highly complex and cannot be represented exactly using a mathematical function.

We do perceive some geographic features (such as survey monuments, triangulation stations, spot heights, and soundings) as something of interest at a particular spot. The mathematical representation of these legitimate point features using a set of coordinates is appropriate. However, data capture devices cannot measure a dimensionless point. The most precise devices can at best obtain measurement of an extremely small area. After we have mathematically defined the entities, we can then perform "data collection" in the mathematical space using methods such as interpolation to generate measurements at a point or along a line.

3.3.1. A classification of geometric primitives

In the last step of mathematical abstraction, we approximate real-world lines with mathematical functions and we reduce a very small area into a single point. We also choose a co-ordinate system and a set of geometrical primitives to represent the tessellation cells. Popular geometrical primitives are 0-dimensional points, 1-dimensional polylines, 2-dimensional polygons, and 3-dimensional polyhedra. In theory, it is only necessary to recognise one primitive called a point and an optional set of *connectors* (or *connecting rules*). These connectors imply certain topological relationships between the geometric primitives being connected. A digital elevation model (DEM) is a feature-primary tessellation with the features represented by unconnected points (Fig. 10).

Points are connected to form a line, lines are connected to form areas, and areas are connected to form solids. Pixels and voxels are just special cases of areas and solids, respectively. For the sake of convenience, however, we will include pixels and voxels as primitives as well. We will not, however, use the terms pixels and voxels here because they

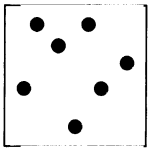
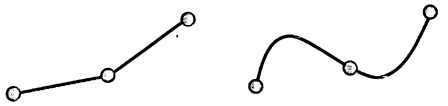


Fig. 10. Irregular DEM: an irregular FPT represented by points.



a) Linear Polynomial b) Higher Order Polynomial

Fig. 12. Interpolating function as point connector.

imply elements of uniform size and shape. Instead, we will call them *2D blocks* and *3D blocks*, and the blocks can differ in shape and size (Fig. 11).

Although a 2D block can be represented by a small polygon composed of four points as shown in Fig. 3, it is convenient sometimes to regard it as a rectangle with a position. In that case, we only need to record its size (one width for a square and an additional height for a rectangle) and a pair of coordinates. A cubic block, on the other hand, can be represented by one width and a triplet of coordinates.

To connect an ordered list of points to form a line, we need an interpolation function. A very popular one is a linear piecewise polynomial (Fig. 12a) that joins every two adjacent points with a vector, and this results in what we called a vector representation of a line or a polyline. Higher order polynomials (Fig. 12b) and other functions could be used, such as a cubic spline, to join the points. The use of non-polygonal cells for tessellation has been proposed by Chen (1985).

To form an area, we connect the end point of a line to the start point of the same or another line. If the lines are polylines, then the area formed is a polygon. To form a solid, we connect areas together. A polyhedron is formed when the areas making up the solid are polygons. For blocks, the only operator required is a logical union operator.

There is not much need to recognise complex objects in a tessellation model since a space is decomposed to cells that are as simple as possible in geometry. There are many constructive methods for forming complex objects from simple primitives. Examples include the boundary model (BRep) of representing solids through a series of areas and the constructive solid geometry (CSG) model that put building block primitives together to form more complex solids. One of the objectives of space tessellation is to decompose a given space into geometric primitives so that constructive methods can then be used to form more complex entities in the database.

We have mentioned earlier that data collection from the real world does not actually measure areas instead of points or lines. After the mathematical abstraction of the tessellation, it is then possible to obtain data with respect to a mathematical point or mathematical line using techniques such as interpolation.

a) Point b) 2D Block c) 3D Block

Fig. 11. The geometric primitives.

3.3.2. *Topological ambiguities of geometric entities*

The selection of a geometric entity for a feature not only affects its accuracy and shape, it also affects the representation of topological relationship among the features. One of the unique aspects of spatial data handling is that topological relationships are often derived from coordinates. It is hence very important that the geometry of the primitives captures the correct topological relationships among them. In this section, we will describe some of the topological ambiguities that could result from a wrong choice of geometrical representation. To give the proper mathematical treatment to this topic, we will first introduce the concept of a topological cell. This will eventually lead to a definition of the interior of a cell, a property that could cause topological ambiguity. This treatment of neighbourhood and the concept of interior are due to Henle (1979), and these concepts are central to the discussion of topological relationship between geometric entities (Worboys, 1995; Pigot, 1991; Egenhofer and Franzosa, 1991).

A *cell* is a figure topologically equivalent to a *disk*, which is loosely speaking an unbroken figure with just one boundary curve. A polygon is a cell with a finite number of vertices forming its boundary. In more precise terms, a closed disk $D = \{P \mid \|P\| \leq 1\}$, and it includes the boundary curve. In this expression, P is a point, and the norm $\|P\|$ for this point is a generalised concept of magnitude in a vector space. If the vector space is the Euclidean space, we recognise an Euclidean norm which is defined as $(x^2 + y^2)^{1/2}$ for a point (x, y). The boundary of the cell forms a closed path that is topologically equivalent to a circle $C = \{P \mid \|P\| = 1\}$.

A cell is called a polygon when its boundary curve is formed by a finite number of vertices. Cells are used to form complexes. For instance, a sphere is a complex formed by putting two cells together. A *polyhedron* is a complex topologically equivalent to a sphere.

To give more precise meaning to the notion of "interior", we need to first define the concept of neighbourhood. A *neighbourhood* of a point P is any open disk $\{P \mid \|P\| < 1\}$ that contains P. Note that a neighbourhood does not contain the boundary curve and can be of any size. If we have a closed subset A of a space, we can define what we mean by P being *near* to A. A neighbourhood being an open disk has the mathematical advantage that it approaches the boundary as its limit. This captures better the concept of a continuous space.

Let us take the two-dimensional case. A point P is *near* A if every neighbourhood of P, regardless of size, contains a point of A. By this argument, only points within A and on its boundary are near A. For any point outside A, there will be a neighbourhood of it not containing part of A. If A is an open subset, a similar argument holds and all points inside and on the boundary are near A.

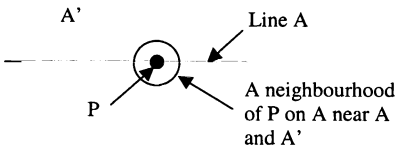


Fig. 13. A line in a two-dimensional space has no interior.

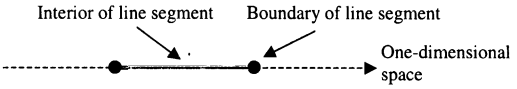


Fig. 14. Interior of a line in a one-dimensional space.

This definition of nearness apparently applies to the definition of interior as well. Indeed, with some refinements, it could, and the *interior* of A is the set of points not near its complement A'. The boundary of A hence consists of all points that are near both A and A'.

Take the case of a point in the two-dimensional Euclidean space. Its complement is the open disk not containing this point. At the point, we can see that it is near both A and A' and, hence, is its boundary. From this observation, it follows that there is no point in A that is not near A' and, therefore, A has no interior (Fig. 13). We can argue in a similar way that a line in a two-dimensional Euclidean space has a boundary but no interior.

This lack of an interior is caused by the dimension of the geometric primitives being lower than that of the space. In other words, in a three-dimensional Euclidean space, points, lines, and polygons have no interior but only boundaries. When the dimension of the space is two, lines will have interiors but points do not. In this case, the two end points of a line segment are its boundaries (Fig. 14).

There are several topological ambiguities that could be caused by this phenomenon. Without an interior, it is impossible to distinguish between sharing a common boundary and overlapping the interior points of a line (Fig. 15). A similar problem happens to two polygons in a three-dimensional space — we cannot determine whether two such polygons are overlapping or adjacent to each other vertically. It is useful to note that the use of 2D blocks or 3D blocks

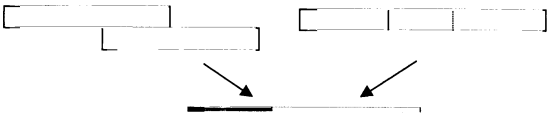


Fig. 15. Topological ambiguity of lines.

in two-dimensional and three-dimensional spaces, respectively, would not cause such topologically ambiguities. However, the exclusive use of 2D blocks or 3D blocks as geometric primitives cannot handle point features effectively although we can argue that our data collection methods from the real world cannot really measure 0-dimensional point.

4. Encoding the tessellation

At this stage, we have defined a tessellation of the database space and have given each cell a geometric meaning. Tessellation encoding employs the most efficient method to represent it in computer storage.

Suppose we form a space-primary, regular 4^4 tessellation based on squares. We have several choices of encoding such a tessellation. One of them is to use a polyline to represent each block although this method is highly inefficient (Fig. 16). Another method is to use a matrix of blocks (each with a dimension plus a location). Due to the regularity of this tessellation, a geometric compression on the coordinates can be used to reduce the storage size. A well-known method is to assign a linear ordering to the blocks (such as row-major), record an origin for the matrix, and store the block size. From this, the coordinates of each block can be calculated. This method of encoding produces what we called a *raster representation* and the blocks are called *pixels*.

After geometric compression of a raster representation, we could also perform a compression of the attributes by grouping pixels of same attributes together. Again, many different methods are available including run-length encoding.

Take quadtree as another example. It is a space-primary, hierarchical, and regular tessellation. There are many methods for encoding a quadtree, including

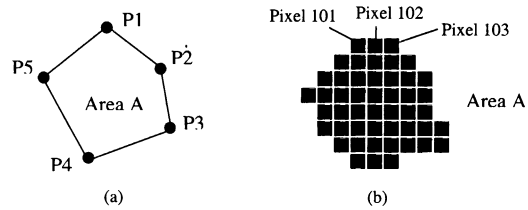


Fig. 17. Two examples of feature-primary tessellation.

the use of an actual tree based on pointers or a linear encoding without pointers (Gargantini, 1982) called *linear quadtree*. It is interesting to note that a quadtree can also be used as an attribute compression technique for a raster representation.

For feature-primary tessellation and space-primary tessellation that are non-hierarchical and irregular, it is more common to use points, polylines, polygons, and polyhedra for their representation. This produces what is commonly known as the vector representation.

In summary, the following are the common encoding methods: absolute coordinates, relative coordinates, directional coding (Freeman, 1974), uncompressed raster, run-length, and linear quadtree.

5. Examples

We will show in this section that the classification scheme suggested in this paper can place tessellation methods in their proper places. The method is particularly useful for the emerging method of combining "vector" and "raster" representations in the same

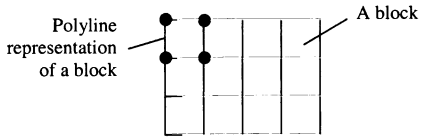


Fig. 16. Polyline representation of 2D blocks.

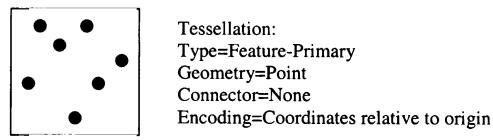
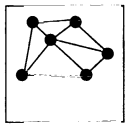


Fig. 18. DEM.



Tessellation:
Type=Space-Primary, constrained, non-hierarchical, irregular
Geometry=Point
Connector=Linear polynomial
Encoding=Coordinates relative to origin

Fig. 19. TIN.

Tessellation:
Type=Space-Primary, constrained, hierarchical, regular (4^4)
Geometry=2D Block
Connector=Logical Union
Encoding=Linear Quadtree

Fig. 21. Quadtree compression of a raster image.

scheme. We will first suggest here a notation for describing a tessellation and then use a few examples for illustration.

The notation is composed of the following parts:

1. Tessellation type to indicate its abstraction on the database space. For instance, space-primary, constrained, hierarchical, and regular. For a regular tessellation, its pattern code, such as 42 as explained in Section 3.2.2, should be included.
2. Geometry primitive to indicate if a point, a 2D block, or a 3D block is used to represent the tessellation cells.
3. Connector to indicate how the primitives are related to each other to form lines or to cover regions.
4. Encoding to indicate the encoding methods used for compute storage of the tessellation.

The first example is what we normally call a vector representation (Fig. 17a). The second example is a FPT encoded with pixels (Fig. 17b). The two examples illustrate how the proposed taxonomy helps to highlight the similarities and differences of the two tessellation methods. The third example shows an irregular DEM (Fig. 18). Fig. 19 is a triangulation of the irregular DEM. If each of the cells in the tessellation in Fig. 19 were to be represented by blocks of pixels as in Fig. 17b, its geometry will be changed to "2D blocks" and its connector will be changed to "logical union".

Figs. 20 and 21 illustrate two applications of a quadtree tessellation. In Fig. 20, the quadtree tessellation is used to partition a space for spatial indexing of point features. The stopping condition is when each quadrant contains less than or equal to 100 points. So in this case, the tessellation is space-primary but constrained by the distribution of the point features. For encoding, we assign a unique identifier to each block according to the linear quadtree ordering. A logical union connector is used here to imply that the topology among the blocks is important. In Fig. 21, maximal quadtree blocks are used to compress a raster representation of an originally unconstrained space-primary tessellation. The resulting tessellation is, however, constrained because the final configuration of the tessellation is affected by the distribution of the features in this case.

Tessellation:
Type=Space-Primary, constrained, hierarchical, regular (4^4)
Geometry=2D Block
Connector=Logical Union
Encoding=Linear Quadtree

Fig. 20. Quadtree tessellation for spatial indexing.

6. Conclusions

In this paper, we have described the various forms of space tessellation and the different geometric primitives and encoding methods that could be used to implement them in a database. At the same time, we have proposed a taxonomy for a more systematic description to avoid the confusion created by the use of the terms "vector" and "raster". In our proposal, we differentiate between a conceptual model and an implementation model for tessellation. The conceptual model describes how users perceive the partition of space into manageable units at an abstract level.

We recognise here two main types of tessellation — feature-primary and space-primary producing feature and space cells, respectively. Feature cells are mostly irregular in shape. The judgement on whether a tessellation is feature-primary or space-primary depends on whether a feature or a portion of space not directly related to features is more important to the application. For space-primary tessellation, we also recognise that the space cells resulting could be formed arbitrarily or constrained by the presence of features. In addition, constrained tessellation can be hierarchical or non-hierarchical. Unconstrained space

cells are mostly regular in shape while the constrained ones that are non-hierarchical are mostly irregular in shape.

We suggest that there are three geometric primitives that could be used to represent the feature or space cells. They are points, 2D blocks, and 3D blocks. The blocks are really points associated with a size. In most cases, these primitives do not exist in isolation but are explicitly linked by connectors. A study of the topological concept of interior shows that the wrong choice of geometric primitives could lead to topological ambiguities.

In order to implement these geometric primitives, we need a set of encoding rules. Many such rules exist, some produces what we traditionally call vector representations and some produces what we call raster representations.

Confusion in terminology usually reflects confusion in concepts, and confusion in spatial concepts results in ambiguous spatial data models. We hope that a more in-depth and perhaps more rigorous classification of space tessellations could help further exercises in spatial data modelling, a task central to data interoperability and sharing.

Acknowledgements

We acknowledge the partial support of grant G-S461 from the Hong Kong Polytechnic University in this research.

References

Bell, S.B.M., Diaz, B.M., Holroyd, F., Jackson, M.J., 1983. Spatially referenced methods of processing raster and vector data. Image Vision Comput. 1 (4), 211–220.

Cartell, A.C., 1991. Concepts of space and geographical data. In: Maguire, D.J., Goodchild, M.F., Rhind, W. (Eds.), Geographical Informational Systems: Principles and Applications vol. 1 Longman, UK, pp. 119–134.

Chen, Y.C., 1985. An Introduction to Hierarchical Probe Model. Technical Report, Department of Mathematical Sciences, Purdue University Calumet, Hammond, IN.

Dutton, G., 1990. Locational properties of quaternary triangular meshes. Proceedings of the Fourth International Symposium on Spatial Data Handling Zurich, Switzerland. pp. 901–910.

Egenhofer, M.J., Franzosa, R.D., 1991. Point-set topological spatial relation. Int. J. Geogr. Inf. Sys. 5 (2), 161–174.

Finkel, R.A., Bentley, J.L., 1974. Quad trees: a data structure for retrieval on composite keys. Acta Inf. 4 (1), 1–9.

Freeman, H., 1974. Computer processing of line-drawing images. ACM Comput. Surv. 6 (1), 57–97.

Gargantini, I., 1982. An effective way to represent quadtrees. Commun. ACM 25 (12), 905–910.

Guttman, A., 1984. R-trees, a dynamic index structure for spatial searching. Proceedings of the SIGMOD Conference, Boston, June. pp. 47–57.

Henle, M., 1979. A Combinatorial Introduction to Topology. Dover Publications, New York.

Peuquet, D.J., 1984. A conceptual framework and comparison of spatial models. Cartographica 21 (4), 66–113.

Pigot, S., 1991. Generalized singular 3-cell complexes. In: Waugh, T.C., Healey, R.G. (Eds.), Proceedings of Autocarto 10, Falls Church, Virginia, USA. pp. 368–392.

Samet, H., 1989. The Design and Analysis of Spatial Data Structures. Addison-Wesley, Reading, MA.

Worboys, M.F., 1995. GIS: A Computing Perspective. Taylor & Francis, London.