

Representation Modes

- Spatial objects so far have been quite abstract.
- Now we study the practical implementation of geometric information.
- The problem is the representation of infinite point sets (of the Euclidean Space) in a computer.
- There are two approaches:
 1. Approximating the continuous space by a discrete one (*tessellation*)
 2. Constructing appropriate data structures (*vector mode* or *half-plane* representation)

Additional literature:

Ralf Hartmut Güting, Markus Schneider: *Realm-Based Spatial Data Types: The ROSE Algebra*. VLDB Journal 4(2): 243-286 (1995).

<http://www.informatik.fernuni-hagen.de/import/pi4/papers/ROSEAlgebra.pdf>

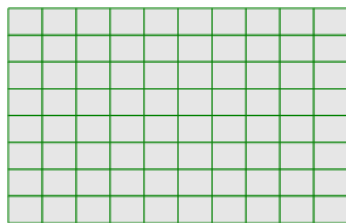
Tessellation

Tessellation is the process of sub-dividing the space of interest into discrete elements.

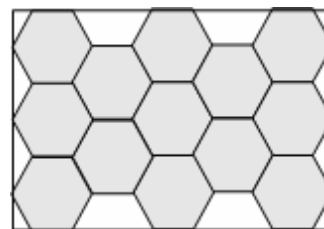
This process defines a *discrete model*, also called *spatial resolution model*, *tiling*, or *meshes*.

There are two tessellation modes:

- *Fixed*, in which the subdivision is based on a grid or *raster*, which is a collection of polygonal units of equal size.
- *Variable*, which handles units of decomposition of variable sizes.



Grid squares



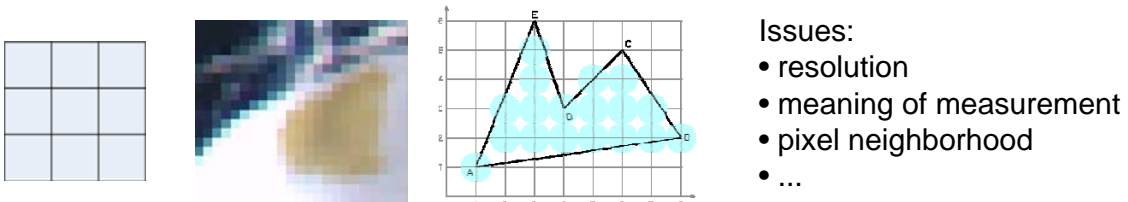
Hexagonal cells

Tessellation (2)

The size of units may change according to the level of resolution required (e.g., in the context of a zooming operator in visualizing maps).

In a raster representation the space is partitioned into a finite number of cells, called *pixels*. Each pixel has an address in the plane (e.g., x, y).

Regular tessellation is frequently encountered in the context of remote-sensing instruments (satellites) and applications. There, field-based data (satellite imagery) is still represented as a function from space (x, y coordinates) to a range of measurement. However, the function domain is no longer an infinite set of points but a finite set of pixels.



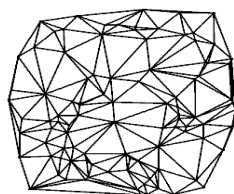
Tessellation (3)

There is a variety of variable (irregular) tessellation models out there, the most popular ones being *Delaunay triangulation* and *Voronoi tessellation*.

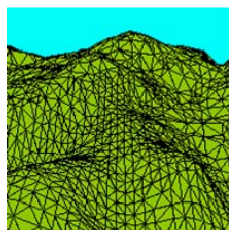
Delaunay triangulation: For a set \mathbf{P} of points in the plane, it is the triangulation $DT(\mathbf{P})$ of \mathbf{P} such that no point in \mathbf{P} is inside the circumcircle of any triangle in $DT(\mathbf{P})$.

Voronoi tessellation: For a set \mathbf{P} of points in the plane, tessellation associates with each point $p \in \mathbf{P}$ a polygon that is the set of points whose closest point in \mathbf{P} is p .

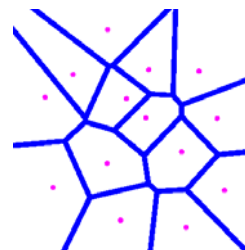
Delaunay triangulation of a discrete point set \mathbf{P} is the *geometric dual* of the Voronoi tessellation for \mathbf{P} .



Delaunay triangulation



Digital terrain
model



Voronoi diagram

Vector Mode

- Geometric objects are constructed from primitives (points and edges).
- A point is represented by a pair of coordinates.
- Compared to a raster representation, a vector representation is not eager in memory.

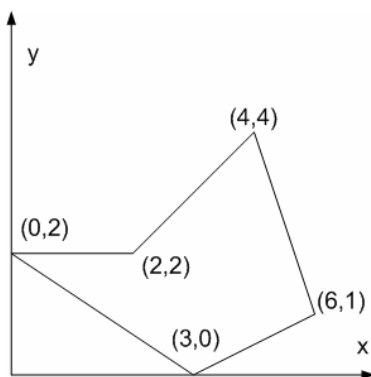
There exists a large number of variants to represent polylines and regions in vector mode:

- Polyline $L = \text{list of points } \langle p_1, p_2, \dots, p_n \rangle$, with each p_i being a vertex. Each pair (p_i, p_{i+1}) with $i < n$ represents one of the polyline's edges.
- A polygon is represented as a list of points, the list is supposed to be a closed polyline.
- A region is simply a set of polygons.

Note that a polygon with n vertices has $2n$ possible representations.

Vector Mode (2)

There is no apparent distinction between a polyline structure and a polygon structure. The software has to do respective consistency and conformance tests. This is also true for other polygon types (e.g., convex polygons).



Vector representation of polygon P

Structure notation:

point: $[x: \text{real}, y: \text{real}]$ (pair)

polyline: $\langle \text{point} \rangle$ (list)

polygon: $\langle \text{polyline} \rangle$ (list)

region: $\{ \text{polygon} \}$ (set)

$P = \langle [4,4], [6,1], [3,0], [0,2], [2,2] \rangle$

Vertex notation:

Vertices are indexed $(1,2,\dots)$, and a polygon is described as a list of index numbers.

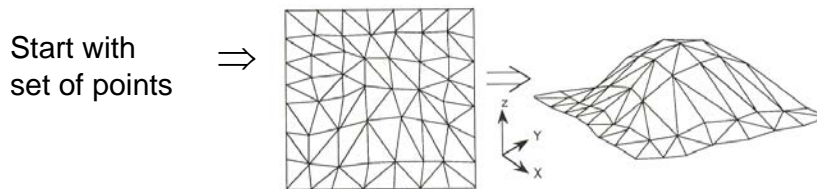
Memory requirements are proportional to the number of vertices.

Vector Mode (3)

Field-based data can be represented in the vector mode as well. There are several approaches, the most prominent ones being the Digital Elevation Models (DEMs):

- Digital (and thus finite) representation of an abstract modeling of space
- Useful to represent any continuous function of the 2-dimensional space
- Through interpolation, value for any point in space can be obtained

Most popular DEM are *Triangulated Irregular Networks* (TINs), which are based on a triangular partitioning of 2D space.



The elevation value is recorded for each vertex, and inferred for any other point by linear interpolation of the three vertices of the triangle that contains the point.

Half-plane Representation

All spatial objects are defined with a single primitive, *half-planes*.

Model, which has been developed quite some time ago, has received some recent attention in the context of *constraint databases*.

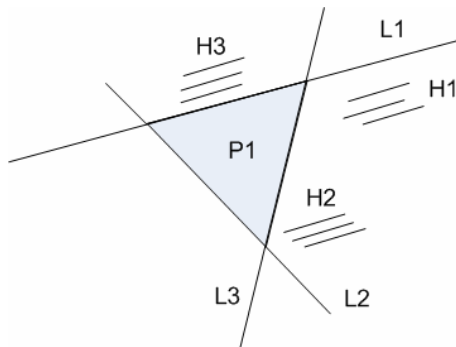
(Def) A *half-space* H in the d -dimensional space \mathbf{R}^d is defined as a set of points $P(x_1, x_2, \dots, x_d)$ that satisfy the inequation of the form

$$a_1 x_1 + a_2 x_2 + \dots + a_d x_d + a_{d+1} \leq 0$$

- A representation of H is the vector $[a_1, a_2, \dots, a_{d+1}]$.
- A *convex d -dimensional polytope* P is defined as the intersection of some finite number of closed half-spaces. If H is part of the half-spaces defining P , $H \cap P$ is the *face* of P .
- The union of a finite number of polytopes is a *d -dimensional polyhedron* Q in \mathbf{R}^d . Q is not necessarily convex, its components are not necessarily connected and they may overlap.

Half-plane Representation (2)

One way to define regions, polylines and points is to consider polyhedra of respective dimensions 2, 1, and 0. A convex polygon with n edges is then defined as the intersection of n half-planes delimited by lines.



Example of a polygon defined as the intersection of three half-planes H1, H2, and H3.

A line segment is a convex polytope of dimension 1, defined by the intersection of two half-lines

A polyline is a polyhedron of dimension 1, obtained by the union of some number of line segments.

A point is 0-dimensional polytope, and a set of points is a 0-dimensional polyhedron.

Geometry of a Collection of Object

So far, the focus has been on entity-based models and on their vector-based representation. In particular, the focus has been on single objects.

Now we are interested in collections of objects and relationships among objects in the same collection.

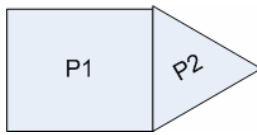
There are three commonly used representations of collections of spatial objects, whose main difference is the expression of *topological relationships* among objects.

- Spaghetti Model
- Network Model
- Topological Model

Note that the explicit representation of topological relationships provides more knowledge and is helpful in efficiently evaluating queries.

Spaghetti Model

The basic idea of this model is that the geometry of any spatial object of the collection is described independently of other objects. No topology is stored in such a model but must be computed on demand.



P1 = $\langle [0,0], [0,2], [2,2], [2,0] \rangle$
P2 = $\langle [2,0], [2,2], [3,1] \rangle$

- Redundancy representation: boundary between adjacent regions is represented twice.
- Enables heterogeneous presentation of several types of spatial objects.
- Pros: simple; easy to enter data about new objects
- Cons: data is stored redundantly; it is difficult to determine or compute some properties, e.g., points shared by two polygons; inconsistencies can arise (for adjacent regions).

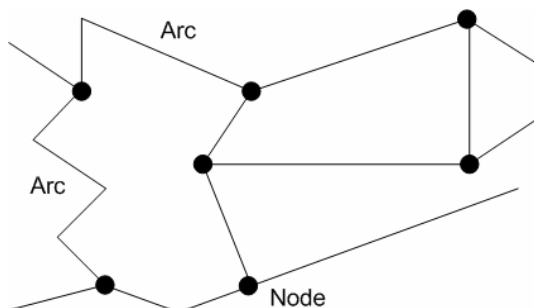
Network Model

Initially designed to represent networks in graph-based applications, such as transportation services or utility management (gas, water, electricity and so on).

In this model, topological relationships among points and polylines are managed (\Rightarrow set of geometric objects to consider is slightly more complex)

The two new concepts are *nodes* and *arcs*:

- A node is a distinguished point that connects a list of arcs.
- An arc is a polyline that starts at a node and ends at a node.



Objects of interest are:

- point: $[x: \text{real}, y: \text{real}]$
- node: $[\text{point}, \langle \text{arc} \rangle]$
- arc: $[\text{start-node}, \text{end-node}, \langle \text{point} \rangle]$
- polygon: $\langle \text{point} \rangle$
- region: $\{ \text{polygon} \}$

Network Model (2)

There are two types of points: *regular points* and *nodes*

- A node is either an arc end-point (extreme) or an isolated point in the plane; other line and polygon vertices are regular points.

Depending on the implementation and application requirements, the network is either *planar* or *non-planar*.

- In a planar network, each edge intersection is recorded as a node, even though the node does not correspond to a geographic object.

One advantage of the network approach is the intrinsic description of a networked topology, with a *notion of connectivity* that is useful for optimal path searches.

No information about the relationships between 2D objects is stored in this model.

Topological Model

The topological model is similar to the network model except that the network is planar.

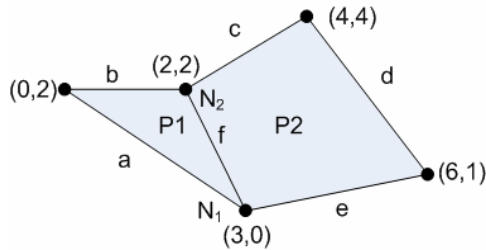
⇒ induces a planar subdivision into adjacent polygons, some of which may not correspond to actual objects

Objects of interest are:

- point: [x: real, y: real]
 - node: [point, <arc>]
 - arc: [start-node, end-node, left-poly, right-poly, <point>]
 - polygon: <arc>
 - region: {polygon}
- As in the network model, a node is represented by a point and a list of arcs starting/ending at that node. This list can be empty.
 - With each arc, the two polygons having the arc as common boundary are stored.
 - A polygon is represented by a list of arcs.

Topological Model (2)

- There is some redundancy, for efficiency reasons. For example, each polygon can be accessed through either polygons or arcs.
- However, there is no redundancy in the stored geometry, as each point and line is only stored once.



One can extract the following objects:

- P1: $\langle a, b, c \rangle$
- P2: $\langle c, d, e, f \rangle$
- f: $[N_1, N_2, P1, P2, \langle \rangle]$
- N₁: $[[3,0], \langle a, f, e \rangle]$

There are several advantages to the topological model

- topological queries can be efficiently computed
- Updates can be done consistently

Drawbacks: some objects in the database have no semantics; complexity of the structure may slow down some operations.