

# AN INTELLIGENT AND DIRECT MANIPULATION DIGITIZING SYSTEM

François Anton and Christopher M. Gold

Chaire industrielle en géomatique appliquée à la foresterie, Centre de Recherche en Géomatique  
Pavillon Casault, Université Laval, Ste-Foy, Québec, Canada. G1K7P4

Tél: (418) 656-549 | Fax: (418) 656-7411

E-Mail: FRANCOIS@PLATO.GMT.ULAVAL.CA  
CGOLD@VM1.ULAVAL.CA

## ABSTRACT

The Voronoi tessellation is of interest in the development of GIS methods for various reasons, especially as a form of topology. These reasons include Voronoi diagram dynamic maintenance and the idea of a "proximal query", whereby an object may be specified merely by placing the cursor within its Voronoi region. This is a great advantage in the digitizing operation, because traditional systems need great care in generalizing the topological relations. The Voronoi model merely requires the specification of each end point by pointing to its Voronoi "bubble". Because the topology is maintained during the digitizing activity, a truly interactive system is possible, unlike the usual "rebuild and then correct the errors" procedure. "Collisions" are detected before they occur, and action taken, rather than awaiting a batch rebuild. This paper discusses the implications of this approach for the construction of a fast, efficient, interactive, user-friendly and "intelligent" (in the sense of help for geometric construction design and topological accuracy) digitizing system. Emphasis will be placed on the development of effective interaction with the user, and the minimization of digitizing time.

## RÉSUMÉ

La tessellation Voronoi est d'intérêt dans le développement de méthodes de SIG pour diverses raisons, spécialement en tant que type de topologie. Il s'agit de la maintenance dynamique du diagramme Voronoi et du concept de "requête proximale", par laquelle un objet peut être spécifié simplement en plaçant le curseur dans sa région Voronoi. Ceci présente un grand avantage pour la numérisation car, les systèmes traditionnels nécessitent beaucoup de précaution dans la génération des relations topologiques. Le système Voronoi exige simplement la spécification de chaque extrémité en pointant sa "bulle" Voronoi. Un système réellement interactif est possible, différent de la procédure habituelle qui "reconstruit et corrige conséquemment les erreurs" car, la topologie est maintenue pendant la numérisation. Les "collisions" sont détectées avant leur apparition et, l'action est prise, plutôt que d'attendre une reconstruction en bloc. Cet article discute des implications de cette approche sur la construction d'un système rapide, efficace, interactif, convivial et "intelligent" (en tant qu'aide à la conception géométrique et exactitude topologique). L'accent sera mis sur le développement d'une interaction réelle avec l'utilisateur et la minimisation du temps de numérisation.

## RESUMEN

La teselación Voronoi es de interés dentro del desarrollo de métodos de SIG por varias razones, especialmente como tipo de topología. Se trata del mantenimiento dinámico del diagrama Voronoi y el concepto de "requerimiento proximal", por el cual un objeto puede ser especificado simplemente poniendo el cursor dentro de su región Voronoi. Eso constituye una gran ventaja para la digitalización, ya que los sistemas tradicionales necesitan mucha precaución en la generación de las relaciones topológicas. El sistema Voronoi solo requiere la especificación de cada extremidad apuntando su "burbuja" Voronoi. Un sistema realmente interactivo es posible y diferente de la procedura habitual que "reconstruye y despots corrige los errores". Entonces, las "colisiones" son detectadas y la acción está tomada en vez de esperar una reconstrucción en bloque. Este artículo discute de las implicaciones de esta acerca en la construcción de un sistema rápido, eficaz, interactivo, conviviente y "inteligente" (como ayuda a la concepción de construcciones geométricas y exactitud topológica). El accento estará puesto en el desarrollo de una real interacción con el utilizador y la minimización del tiempo de digitalización.

## Introduction

Map digitizing is the main way to enter data into a map with computer assisted drawing (CAD) or a geographic information system (GIS) software. This action supposes more than drawing a series of points (connected or not). It primarily implies managing topology through time in order to achieve dynamic spatial analysis, a recent trend in using GISs. Present systems do not allow immediate topology, and thus direct manipulation. Indeed, they construct topology by a fully hidden batch process. Furthermore, they require precise pointing for selecting objects and the selection tools are often limited to graphical pointing devices (mouse or tablet) and the presentation tools (see Jeffroy and Lambert [ 1992]) to the screen. Their user-friendliness is generaiy poor because of hidden processes and lack of ergonomic design. This paper will analyze the product requirements, functional and interface specifications of a digitizing system, and then propose a solution using a dynamic spatial data structure.

### 1) Analysis of' digitizing actions

**Map digitizing.** A digitizing system is firstly an automatic treatment tool of cartographic information. It converts map spatial data objects into numeric format (x and y coordinates). Besides this metric information, topological information can be added expressing spatial adjacency relationships between objects. Cartographic information can take various forms: geometric information (points, lines, polygons), graphical information (colour, line weight, line style, fill pattern, etc.), and symbolic information (graphical symbols, to which is attributed semantic information, as for instance traffic lights and its significance in terms of road networks and its specific attributes in a dedicated database). The automatic treatment of geometric information must include map registration tools, topological data structure digitizing tools, drawing digitizing tools, and geometric construction digitizing tools. Within map registration tools, we should be able to manage the different coordinate systems (map projection and zone if necessary, ellipsoid, geodetic datum) and convert data from one system to the other. We should also be able to present reference points registration results, offer various kinds of transformation models, and display the final results in order to evaluate them.

Among drawing digitizing tools, we can suggest: point, line, circle, ellipse, curve, rectangle, polygon filling and text placement. These tools are those we can find within CAD software such as MicroStation or AutoCad. The survey geometric construction tools are less common than geometric construction ones. Nonetheless, they are useful for cartographic completion. The surveying geometric constructions include methods using only angular measures, methods using only distances measures, and methods using both.

The automatic treatment of graphical information must include graphical interface tools for selecting the various graphical parameters: colour (chosen within a palette), line weight (chosen for instance within a scale), line style (chosen within a palette), filling pattern (chosen within a palette) and, punctual symbol (chosen within a palette). Palette creation and editing tools must be available (in particular for filling pattern and punctual symbols) in order to customize the software for various application domains (civil engineering, urban and regional planning, finance, etc.). in addition, the automatic treatment of graphical information being similar to the symbolic information case, these two kind of treatments should be inter-related, as it is the case within some GISs (TIGRIS from Intergraph). Therefore, these interface tools should be merged in a symboiogy dialog box, that would be linked with the relational data base management system.

**Map digitizing operations.** Map digitizing is composed firstly of the definition of a digitizing process: a map is not digitized randomly! This fact implies the adoption of an order- of digitizing of spatial objects (isolated points, then lines; clockwise order; from left to right and from top to bottom; etc.). This digitizing order can also include a map sheet assembly board if there is a great number of folds within the map, a coverage naming convention, and a time planning. After having defined a digitizing process, it is time to prepare the map by identifying the reference points the map, identifying the non trivial nodes, and identifying polygon labels. Finally, we can proceed to digitize the map. This last phase is composed of map registration: selection of the coordinate system, editing reference point coordinates and digitizing these, choice of the transformation model (Helmert, affine, polynomial), checking the map registration, and eventually redoing one or more of the last phases; and digitizing map spatial objects. The map digitizing operation analysis will be concerned firstly with the general operations of operating systems. Secondly, it will include the following basic digitizing actions: adding a point, adding a line, adding a curve, deleting a point, deleting a line, undoing the last change, snapping, placing text, filling an area, selecting graphical parameters, joining two objects, and placing a symbol or a label, and thirdly, the analysis will concern spatial analysis actions such as cursor nearest object detection, neighbour detection, buffer zone and map overlay. Various GISs were compared: Arc/Info, Genamap, MicroStation, Tigris (general GISs) and a forest applications GIS: Terrasoft. The results are in the table 1 at the end of this article (Genamap excepted, because it is similar to Arc/Info). We can see a wide range of courses of action. Arc/Info and Genamap are characterized by few interactive operations: the command language is required to do a lot of tasks, and the functions available through digitizing tablet cursor buttons are limited to geometric objects and label editing. Furthermore the operations of these GISs separates the work session, firstly between geometric objects editing and symbolic information editing, and secondly, between on the one hand, object editing and, on the other hand, geometric construction editing and spatial analysis. Arc/Info is characterised by a simpler and **more efficient** digitizing course of action: the number of steps in it is **far** less numerous than in Genamap. MicroStation, Terrasoft and Tigris are characterized by more user-friendly operations, nearer to CAD software, offering more user-friendliness in general tasks (studied also within operating systems), drawing (especially MicroStation), spatial analysis (Tigris and also Terrasoft), and topological ones (Tigris). The selection components of these **GISs** include always menus for all the available functions. The voice interaction within these software is always limited, when it exists, to the confirmation or rejection of information entered through the digitizing tablet. To conclude GISs operation analysis, there are two groups of GISs according to user-friendliness: a first group is constituted by Arc/Info and Genamap. This group has a poor user-friendliness. The other group is constituted by the other GISs and has an appreciably better user-friendliness, one that is more digitizing tablet oriented in the case of MicroStation and Terrasoft and more object-oriented in the case of Tigris.

## 2) Implications for designing a digitizing system GUI

**Interface specification.** The design of the graphical user interface has to be done from an ergonomic stand point, thus adapting to the user (according to his or her average profile), the tasks he or she has to fulfil, and the working environment. We will use here a method allowing us to deal with the paradox of ergonomics in design (Pinsky and Theureau [ 1984]): the analysis of the course

of action, developed by L. Pinsky and J. Theureau at the CNAM (Pinsky and Theureau [1987]). This conceptual framework is guided by following principles: “analysis focusing on the genesis of user actions, with identification of the meaning of actions from the user point of view, clinical analysis of a set of actions, through verbal and action protocols, analysis conducted in situations as close as possible to real work conditions” (Jeffroy and Lambert [1992]). The design of a new system must always refer to an existing system and be interactive, including analysis of reference, simulation, and future situations. Within the reference situation, we will study two cases: the use of software more or less equivalent to future software (operating systems and recent GISs already studied), and the use of the existing prototype (presently under development at Laval University). It is convenient to firstly analyze how some fundamental tasks such as creation, opening, selection, deselection, copying, move, deletion, closing, duplication and pasting are operated within graphical operating systems and commonly used software (word processors, communications, etc.).

Within operating systems, one graphical operating system (Apple System 7) and two graphical interfaces (Microsoft Windows 3.1 and Sun OpenWindows) have been tested. Some characteristics allow us to distinguish these systems. What mostly distinguishes Windows 3.1 from Apple System 7 is the deletion of an object: while in addition to the object selection, Windows requires the selection of a menu item (“Delete” item of “File” menu), Apple System 7 merely requires dragging the object to the wastepaper basket. What distinguishes OpenWindows from Apple or Windows is the fact that there is a true deselection under OpenWindows (by clicking the middle button of the mouse) while the user has to do another selection in order to deselect under the other systems. To abstract operating system course of action analysis, there is a progressive trend towards user-friendliness from Microsoft Windows to OpenWindows through Apple System 7. Commonly used software reproduces the same course of action for these general tasks as the operating systems under which they have been implemented.

At this stage, graphical interface tools should allow the software to be more user-friendly than the simple graphical window with the command language text window of some GISs. It should also respect the OpenWindows course of action for general actions. The choice of digitizing tablet cursor as the primary selection component and symbols (mobile palettes or fixed button bars) as the primary presentation component offer a good user-friendliness for graphical editing. This user-friendliness is the best with mobile palettes, because they are moveable and can be closed according to use. Coordinate system choice and map registration parameter editing imply dialog boxes due to the nature of the information (alphanumeric and multiple choice). Alphanumeric information is better entered in text boxes (editing control) while multiple choice information is better entered through combo-boxes or check boxes. Button-shaped commands can be inserted within a map registration dialog box in order to launch registration calculation (after the last reference point), redo registration, control and edit entered data or edit residuals. Due to the geometric nature of the information treated in map digitizing, the graphical interface should be composed of a graphical window with horizontal and vertical scrolling (because map dimensions often exceeds screen ones), and moveable graphical palettes. These palettes must include a sufficiently visible and memorable symbol for each command and cross-reference to the other palette buttons. This last solution is better than unrolling palettes, that can take a lot of space when highly complex of actions, and they are less manageable than simple ones. In order to decrease switches between the digitizing board and the keyboard, all commands should be available by menus, and data structure digitizing commands should be directly available through prior palette and the digitizing cursor buttons. Also, the user’s visual target switches (between the map on the digitizing board and the computer screen), should be avoided in an

ergonomic approach Some CAD software use a loudspeaker in order to confirm (**high-pitched** sound) or reject (double deep sound) digitized data .We could also design a more general **multi-media** solution, where the computer not **only says** "ERROR" if a transmission problem occurs or "COLLISION" if a collision is imminent, but also recognizes vocal commands such as "POINT" (to add a point). "LINE" (to add a line), "DELETE" (to delete the selected object) or "JOIN" (to join two or more objects). We could thus widen the selection components (voice recognition and presentation components (artificial voice) ranges. This **allow US** to design even **more User-friendly**, but also ergonomic applications and to reduce digitizing time. A study achieved by Stuttgart University computer science institute (Kundel and Strothotte [ 1988]) has shown that with the same graphical application, the time necessary to accomplish the same tasks is less with bimodal interfaces (menu and voice) than with unimodal interfaces. **To** the WYSIWYG (what you see is what you get), it could be convenient to add the "WYSIWID" (**what you say is what I do**) and the "WYHIWID" (what you hear is what I did).

**Direct manipulation GUI requirements.** "Direct manipulation" graphical user interfaces have the following characteristics: "continuous representation **of** the object of interest, **physical** actions or labelled button presses instead of complex syntax and command names, rapid incremental reversible operations whose impact on the object of interest is immediately visible." (Gorner, Vossen and Ziegler [ 1992]). These interfaces obey the "WYSIWYG" principle already mentioned. Direct manipulation has gone through comparative studies, that show its superiority increases with the duration of the tests and order of intricacy of accomplished tasks. According to Shneiderman (Shneiderman and Margono [ 1987] in Gorner, Vossen and Ziegler [ 1992]), the main theoretical advantages of direct manipulation for digitizing are the following: "novices can learn basic functionality quickly", "experts can work extremely rapidly to carry out a wide range of tasks, even define new functions and features" and "reduced anxiety of the user because of the easy reversibility of the actions". These considerations imply no hidden operations and continuous map construction and maintenance within a digitizing system. This can not be the case within vectorial or raster GISs that construct and update topology through hidden batch processes and display the map in a discontinuous way while updating, because of non-immediate topology. From a more technical standpoint, the software and its GUI must not refer to device coordinates but logical coordinates, to obey the WYSIWYG principle.

### 3) Advantages of the Voronoi structure

**The Voronoi data structure.** This data structure is also known as Thiessen polygons and has been widely covered by Gold [ 1993], Aurenhammer [ 1991], and many others. The Voronoi data structure implemented here is a dynamic data structure of points and line segments based upon a continuous model of space (fig. I). It is actually an irregular tessellation, where space is subdivided according to the proximal zone of each object. Each of the objects of the data structure correspond exactly to one and only one real object. From this standpoint, it can be considered as a combined vision of vectorial and raster data structures, taking their respective advantages. Vectorial systems give the flexibility of objects (points or lines) and the exact correspondence between real objects and mapped objects. Raster systems bring the direct (implicit) knowledge of cells' spatial adjacency relationships.

The Voronoi system geometry is a topological geometry, where space is described not only

by its metric properties but also by its topological ones, while traditional systems geometry is metric, and topology is reconstituted from metric properties (vectorial GISs) or metric squared space (raster GISs). The point and line segment Voronoi diagram implying a subdivision of the space around each object (according to its influence zone), lines are actually double half-lines, inseparable one from the other, but having each one its own influence zone. Also, polygons do not exist as individual objects but, as the set of objects whose influence zone totals up polygon area (Gold [ 1991 ])

**Advantages of Voronoi data structure for map digitizing.** Within a Voronoi system, objects and their spatial adjacency relationships have dual representations (linked by a reversible isomorphism) stored in the data structure. We can pass easily and exactly from one representation to the other, knowing immediately the spatial adjacency relationships of any object. The objects' representation is the point and line segment Voronoi diagram, and their spatial adjacency relationships representation is the Delaunay triangulation. To each Voronoi diagram boundary corresponds one and only one edge (spatial adjacency relationship) of the dual Delaunay triangulation. The main advantage of a topological geometry is related to figure invariants. This has important implications on digitizing. Firstly, this topological geometry implies a topological error-free digitizing: polygon closing is no longer achieved by a metric tolerance or an automatically computed intersection but by an appropriate digitizing operation (join the first and last vertices of a polygon). Furthermore, whatever isometric geometric transformation (translation, rotation and their combinations) is applied to a map, topological relationships remain unchanged. Also, topology is not changed by a coordinate system change. In addition, map editing is facilitated because a change in an object's metric properties does not systematically imply the deterioration of its topologic properties, as it is the case within batch topological reconstruction of vectorial systems. Topology changes are reflected by switches within the Delaunay triangulation. Therefore, topology changes take place locally, dynamically (local, reversible switches), and interactively. Indeed, the interactive aspect is due to the fact that the different topology maintenance phases are visible (we can see the Voronoi diagram changes while a mobile point is moving to its final destination). Furthermore, the Delaunay triangulation allows a total ordering of the space and thus, a spatial search that is no longer global or partial, but local and linear ("linear walk"). This local spatial search takes place from the last triangle used (corresponding to the last object treated) and corresponds well to the map digitizing process. In fact, a map is not digitized randomly, but by adjacent regions and with a certain order (first isolated points, then arcs, etc.). Hence, undoing the last operation is faster when digitizing, because the last entered object is usually neighbouring the last searched object. In addition, the Voronoi data structure map construction process is sequential, interactive and all geometric operations are visible while they are being done. The map construction process is highly interactive, because firstly, the Voronoi system can warn the user when a collision is imminent and allow: an "intelligent snap"; or resuming digitizing after having created a new node at the computed crossing point; or rejecting the collision. Secondly, the topology maintenance is done as the data structure is constructed because Voronoi geometric operators are local. Therefore, as the graphical pointing device is moving, the nearest object from the cursor position can be highlighted and all geometric operations (including topological events) can be shown. This fact allows us to design a "direct manipulation" graphical user interface, that will be treated next.

The knowledge of spatial adjacency relationships (stored as objects) being direct, non-metric requests of spatial analysis can be highly facilitated by this data structure. We can quote the polygon in which a point is located and the objects neighbouring a given object. Furthermore, this data structure allows a better control of some GIS basic operations. Indeed, line crossing, detected while

the mobile point moves, polygon closing (crossing or join), and buffer zones are easily and precisely obtained, because topological information has been digitized along with metric information

#### 4) Design of the Voronoi “intelligent” and interactive digitizing system

**Design of the Voronoi digitizing system GUI.** Given the transparency of Voronoi operators the design of a “direct manipulation” graphical user interface is possible. The kind of adopted simulation is a compromise at the level of realization time (Jeffroy and Lambert [ 1992]). Indeed, the prototype on an experimental site is the most powerful, but requires a little more time than the static mock-up, but less than the dynamic mock-up. The tested software is the Voronoi digitizing system under development. It includes a Voronoi engine dynamic-link-library and a direct **manipulation** graphical user interface. The course of action will be analyzed with the same tasks as with GISs. The course of action corresponding to general tasks is the same as the OpenWindows system and the general design of the graphical user interface is done according to OpenWindows operations design-

A tot of actions are different in Voronoi digitizing, mostly due to the different way the Voronoi data structure considers the space: objects and their influence zone (fig. 2 and 3). Indeed, selecting an object merely requires that one points to its Voronoi region (with mouse or, better, the “SELECT/MOVE” button of digitizer cursor), because its Voronoi region is the set of points dedicated to the considered object in the space subdivision produced by the Voronoi model. Also, as in the OpenWindows system, there is a true deselection by clicking the mouse right button or a digitizing tablet cursor button within an object’s Voronoi region. This is indeed a truly “intelligent” and innovative (within GISs) way to act. Adding a point is done by placing the cursor on the desired location, and either pressing the “ADD A POINT” button on digitizer cursor button or **saying** “POINT”. Moving a point is done by merely pointing at it and dragging it to its final destination. Furthermore, the Voronoi view of a line is topologically interesting, and has also some implications in digitizing operations. Adding a line adds four objects in the data structure: the two extremities (considered as points) and the two half-line segments composing the line. Adding a line is done by placing the cursor, either pressing the “ADD A LINE” button on digitizing tablet cursor or saying “LINE” and dragging either digitizing tablet cursor or mouse to its final destination. When a collision is imminent, the system will warn the user by saying “COLLISION”. Then, the user can resume digitizing or reject collision if it was not desired. If digitizing is resumed, a new created node will divide each line into two. These kinds of actions corresponds better to the geometric nature of each object and allow a better topology control within the data structure. Indeed, this implies that objects are not defined by their boundaries (a curve, that is to say a linear definition), but by the way they subdivide the space (thus by a spatial definition), and this allows better control because we know the Voronoi subdivision must be a partition of the space, helping avoid the problem of sliver polygons. The deletion of a line does not imply the deletion of their extremities. Line deletion is done by selecting it and either pressing the “DELETE LINE” or the “DELETE” button on the digitizing tablet cursor, or saying “DELETE”. Point deletion is operated by selecting it, and either pressing the “DELETE POINT” or “DELETE” button of digitizer cursor or saying “DELETE”. Undoing the last action is merely operated by either saying “UNDO” or pressing the “UNDO” button on the digitizing tablet cursor. All these actions could have been operated through menu items, but it is simpler using the tablet cursor or voice recognition.

The selection of a polygon is also different: it implies the selection of the set of objects whose Voronoi influence zones totals the polygon area. This requires merely pointing within the polygon

area and either selecting a menu item. "SCAN" or saying "SCAN", that selects all the `objects` (half-lines and points) enclosing the `cursor` position. Snapping is a join operation within Voronoi system and this gives better manageability because joining two or more objects merely requires selecting objects, done in an easier way by pointing to the objects' Voronoi regions and either pressing the "JOIN" button on the digitizer cursor or saying "JOIN". Polygon closing correspond to conscious, topologically error-free action: either a join giving the last side of the polygon or a line to line collision dividing the previous polygon into two new ones. All other commands can be available through menu items only, and their operation is facilitated by Voronoi digitizing proximal query, whose operation is exactly the same as under OpenWindows. Nonetheless, the "intelligent" character of this system is important for topological accuracy. In addition, the recently developed digitizing system allows the user to be aided in geometric design by the Voronoi diagram, because the user can see each object's Voronoi influence zone, marked out by the geometric equidistant curves between two objects. These geometric curves are: the bisector of the line segment joining two points; the bisector of two line segments; and a parabola for a point and a line segment. Finally, because the nearest object is always highlighted, if it is a line segment we see the projection of the cursor onto it.

**Design of the Voronoi interface.** The topological data structure digitizing tools must be composed of nine Voronoi data structure basic operations. These operations can be classified according to their level of split and their level of merge and have been described in Gold [ 1992 ]. The Voronoi interface must convert drawing and geometric construction operations into the nine Voronoi data structure basic operations. If it is simple for figures or geometric objects that are a finite set of points and lines (points, lines, rectangles), it is less simple for other objects such as circles and curves. Here, we should consider if another object can be placed within the complex object or not. In the first case, we should decompose the complex figure into small line segments while, in the second case we should consider these objects as complex symbols, whose influence area is not decomposed into sub-components. Therefore, the user should be prompted to determine if he or she wants a complex symbol or an object decomposed into geometric primitives.

### **Conclusion: A truly intelligent and interactive digitizing system**

The Voronoi data structure offers an interesting way to build a digitizing system, that is user-friendly, obeying the WYSIWYG principle because of its immediate topology, and allowing a proximal query. The Voronoi digitizing system is designed in an ergonomic fashion, allowing the user to concentrate on the digitizing tablet. It also constitutes an "intelligent" way of digitizing, aiding the user in geometric design.

### **Acknowledgements**

The funding of this research was made possible by the foundation of an Industrial Research Chair in Geomatics at Laval University, jointly funded by the Natural Sciences and Engineering Research Council of Canada and the Association de l'Industrie Forestière du Québec.

References :

- Aurenhammer F., 1991. Voronoi Diagrams - A Survey of a Fundamental Geometric Data Structure, ACM Computing Surveys, v. 23 n° 3, pp. 345-405.
- Gold, C.M., 1993. An outline of an event-driven spatial data-structure for managing time-varying maps. The Canadian Conference on GIS, Ottawa, March 1993, pp. 880-888.
- Gold, C.M., 1992. An Object-based Dynamic Spatial Model, and its Application in the Development of a User-friendly Digitizing System. Proceedings, Fifth International Symposium on Spatial Data Handling, Charleston, August 1992, pp. 495-504.
- Gold, C.M., 1991. Problems with Handling Spatial Data - the Voronoi Approach. CISM Journal, v. 45 n° 1, pp. 65-80.
- Gorner C., P. Vossen and J. Ziegler, 1992. Direct Manipulation User Interface, in: Methods and Tools in User-Centred Design for Information Technology, North-Holland, Amsterdam, pp. 237-279.
- Jeffroy F., and I. Lambert, 1992. An ergonomics Framework for user activity centred software design, in: Methods and Tools in User-Centred Design for Information Technology, North-Holland, Amsterdam, pp. 43-92.
- Kundel, K., 1988. Visualisation and Direct Manipulation in User Interfaces: Are we Overdoing it'?. Lecture Notes in Computer Science n° 439, Springer-Verlag, Berlin, pp. 183-193.
- Pinsky L. and J. Theureau, 1987. L'analyse du Cours d'Action - analyse du travail et conception ergonomique. Collection d'Ergonomie et de Neurophysiologie du Travail du CNAM, n° 88, Paris.
- Pinsky L. and J. Theureau, 1984. Paradoxe de l'ergonomie de conception et logiciel informatique, Revue des Conditions de Travail, no 9.

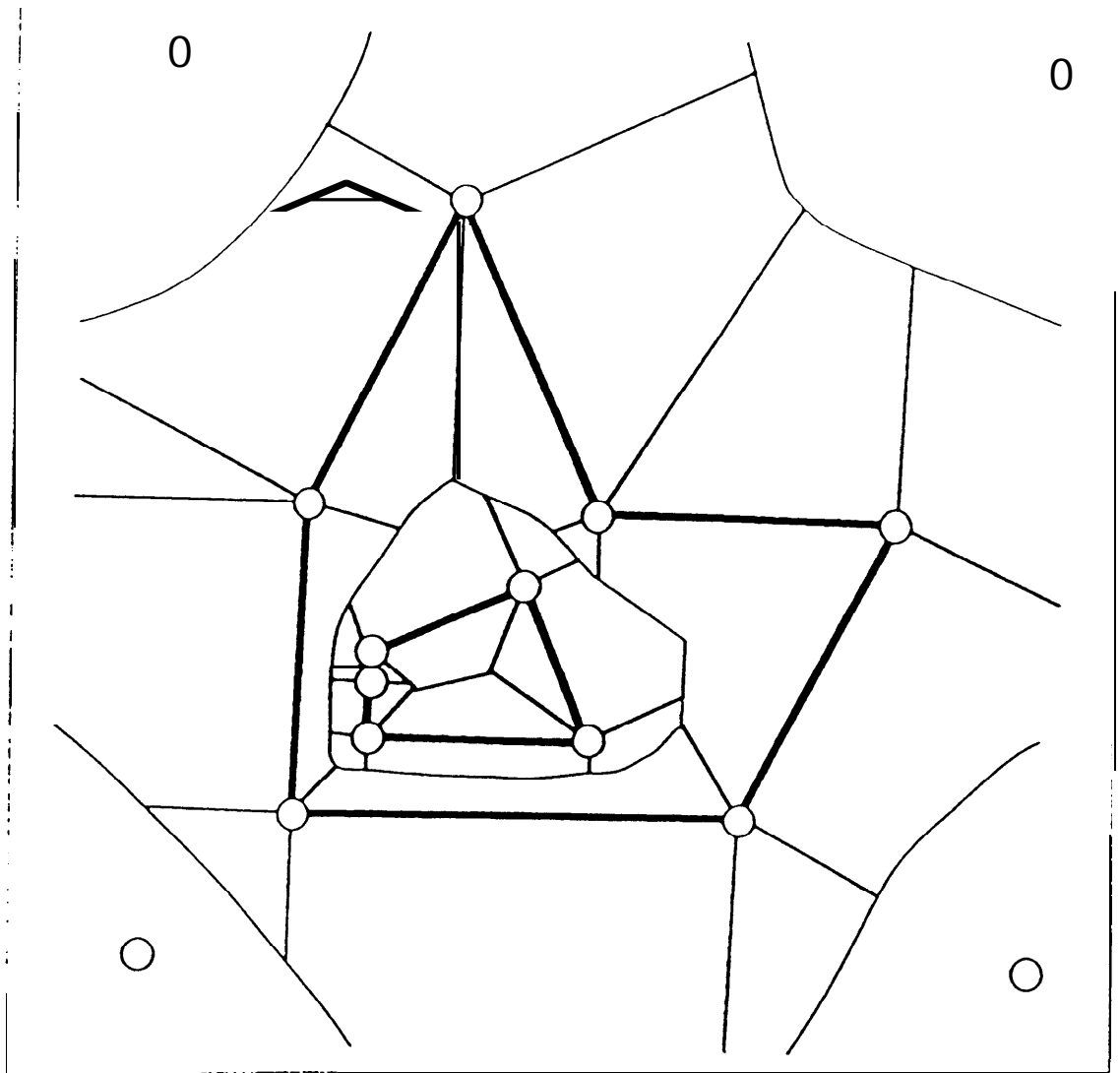


Figure 1

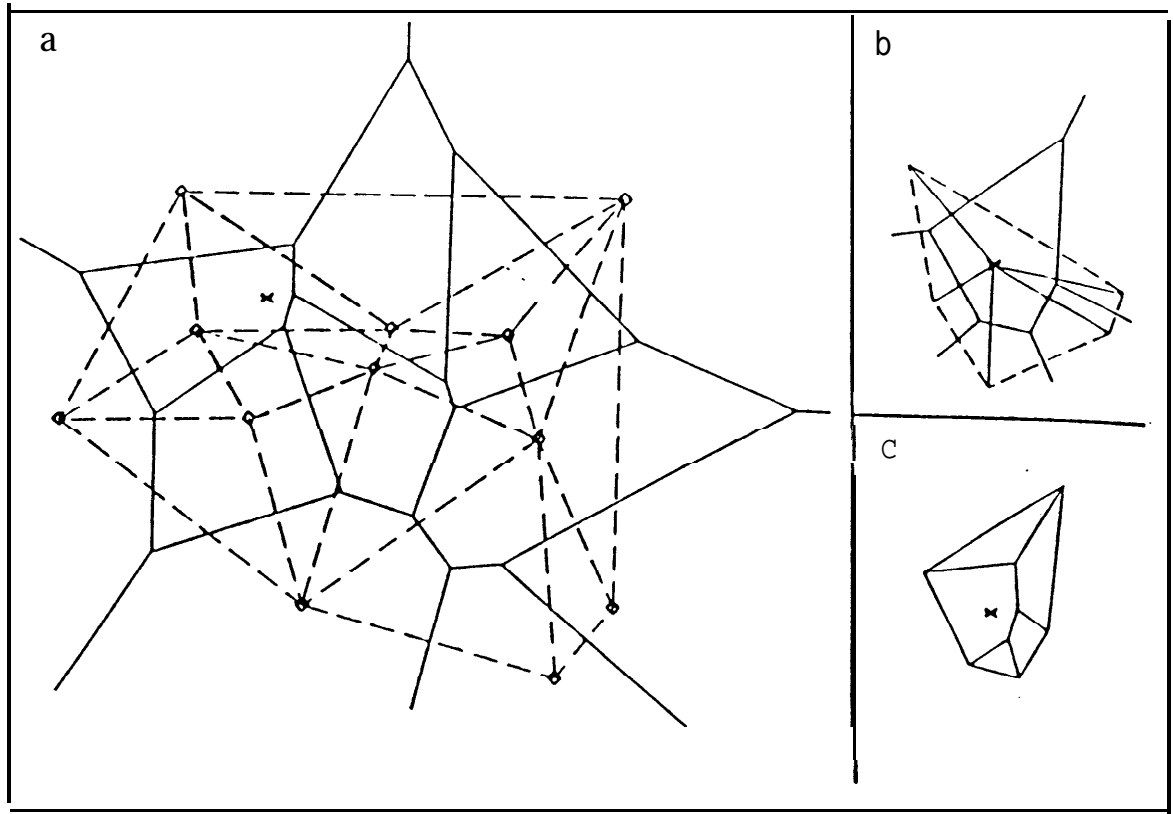


Figure 2. a) Simple **data set with Voronoi** regions;  
 b) Voronoi region after insertion of point **X**;  
 c) Neighbouring areas stolen by point **X**.

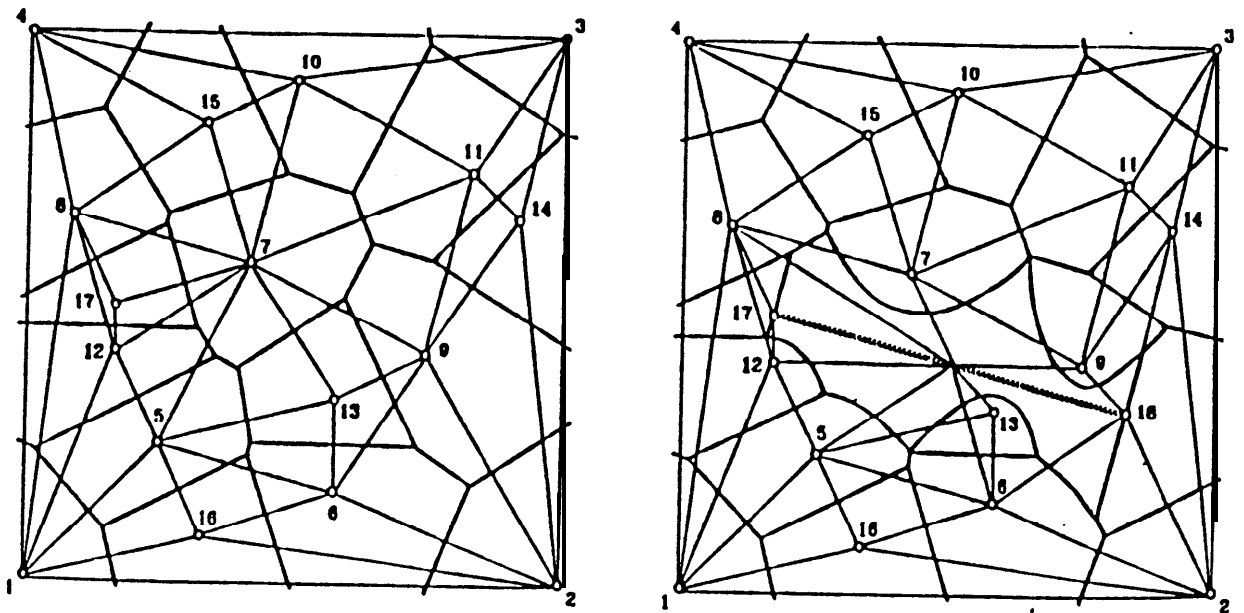


Figure 3. a) A simple point Voronoi **diagram**.  
 b) Addition of a line **segment**.

O.S.	Windows 3.1	Apple Sys 7	OpenWindows 3	
Create	Menu	Menu	Menu	
Open	DBLCLK LB	DBLCLK LB	DBLCLK LB	
Select = SEL	CLK LB	CLK LB	CLK LB or MB	
Deselect	Select another	Select another	CLK MB	
Copy	SEL+CTRL-DRAG	SEL+MENU*2	SEL+MENU*2	
Move = MOVE	SEL+DRAG	SEL+DRAG	SEL+DRAG	
Delete	SEL+MENU	MOVE	MOVE	
Close	CLK LB c.i.	CLK LB c.i.	CLK LB c.i.	
Duplicate	SEL + MENU	SEL + MENU	SEL + MENU	
Paste	MENU	MENU	MENU	
<hr/>				
G.I.S.	ARC/INFO 6	TERRASOFT	TIGRIS	VORONOI
Create	command	MENU	MENU	MENU
Open	command	MENU	MENU+DIALOG	MENU+DIALOG
Select = SEL	command	Buttons A & 6	MENU	Button 0
Deselect	command	Button 8	MENU	Button E
Copy	command	DM Button	MENU+DIALOG	
Move = MOVE	command	DM Button	MENU+DIALOG	SEL+DRAG
Delete	command	DM Button+SEL	MENU	SEL+MOVE
Close	command	DM Button	MENU	CLK LB
Duplicate	command		MENU	
Paste	command		MENU	
Add a point	Button 1	(DM Button)+A	CLK Data	Button A or I or V
Add a line	Button 1 or 2	(DM Button)+A/D	CLK Data/Reject	But. A /2+DRAG or V
Add a curve	Button 3	(DM Button)+A/D	CLK Data/Reject	But. C or V
Delete a point	Button 4	DM Button+SEL	MENU+CLK	Button4 or V
Delete a line	Button 5	Idem	Idem	Button 5 or V
Undo		Button 5	MENU	Button E
Snap	command (auto)	Button 4/2 (P/L)	MENU+SEL*2	Button 6
Join		DM But.+SEL*2	MENU+SEL*2	SEL*2+Button 7
Drawing style	commands	DM Button	MENU+CLK+DB	MENU
Select nearest object	command	MENU	MENU+CLK	Button 0
Place feature	command	DM Button+A/D	MENU+CLK+DB	MENU+CLK+DB
Place text	command	DM Button+A	MENU+CLK+DB	MENU+CLK+DB
Fill an area	command	DM Button	MENU+CLK+DB	MENU+CLK+DB
Overlay	command	DM Button+SEL	MENU+CLK+DB	MENU+CLK+DB
Buffer zone	command	DM Button+SEL	MENU+CLK+DB	MENU+CLK+DB
Select neighbours	command	MENU	MENU+CLK	MENU+CLK
Request	command	MENU	MENU+DB	MENU+DB

N.B.: CLK=click; DBLCLK = double click; c.i. =close icon; P/L = Point/Line; V = Voice; DB = Dialog Box  
 LB = left button; MB = mid button; RB = Right button; DM = Digitizer Menu  
 DRAG = Dragging; auto = automatic; CTRL = CTRL key

Table 1