PERIODICA POLYTECHNICA SER. CIVIL ENG. VOL. 39, NO. 1, PP. 3–20 (1995)

SUBRANGE DATA TYPE APPLIED TO DEFINITION OF SUCH VIRTUAL COORDINATE SYSTEMS WHERE THE DOMAINS ARE CONSTRAINED BY GEOMETRIC BOUNDARIES OF EACH OBJECT

István KÁDÁR* and Erik PAPP**

*Department of Geodesy
College for Surveying and Country Planning
H–8002 Székesfehérvár, Hungary

**Department of Geodesy
Technical University
H–1521 Budapest, Hungary

Received: Jan. 20, 1995

Abstract

Modern dynamic program languages such as ADA, PASCAL, MODULA'2 give us further possibilities to generalize the idea of classical real numbers. There are many applications in GIS where it is not possible to effectively model the invidual values of attributes neither with integer data type (N) nor positive integer numbers (Z). For instance an elevation of object such as attributes could oscillate between lower and upper limits. Those limits are always determined by the information environment. In case of subrange data type the declaration may be the following: high-type = 200 ... 700, where 200 is the lower and 700 is the upper limit within a territory.

Test Results Using Subrange Data Type

Our aim is to present the attempted applications of subrange data types in case of 2D and 3D space attributes generally.

First we transform the polyfill function in a graphic software package in such a way that during the filling procedure the function enumerates the pixels from zero and stops at a desired point, which we wanted to determine with subrange data type, instead of coordinates. If the given point is the same pixel where the filling begins then the 'place code' is zero, and if the last pixel's then the value of place code is very close to the polygon area. The difference depends on the domain in which the filling occurs in cases as

- the polygon is an *open 2D interval,* namely just those pixels will colour whose centre is inside the theoretical boundary of polygon (the domain of subrange data type is *always smaller* than the area of polygon)
- the polygon is a *closed 2D interval,* just those pixels will colour it which are inside the polygon or on its boundary (the domain of subrange data type is *always larger* than the area)
- the filling method is similar, it has almost become standard in digital geometry, namely one half of those pixels which are on the polygon border will belong to the domains of neighbouring polygons and the other half will take over from the neighbours. In such a case the domain of subrange data type is *smaller or greater,* but perhaps it should be *exactly equal* to the area of the polygon.

In the case of polygon system the above described subrange data type can be used in the same way as the original polyfill function works if for example inside a certain given area we would colour the whole forest or fish-pond. If the function does not find the desired point during the filling procedure inside the first polygon then the enumeration will continue at the following polygon skipping the empty domains between the polygons. According to the above mentioned subrange data type is in practice a 2D domain which *consists of more parts* and may be *quite irregular* with *rugged edges* and *holes,* moreover inside holes may be possible one or more *other polygons.*

We modified the well-known structure of Quad-tree and R-tree for the other experiment series so it will be suitable for effective location. Because without modifying, the leaf generating procedure completely *fills the given object with different size leafs.* If a monitoring command stops the filling procedure at the point which we wanted to localize and *summarize the area of generated leafs* then this amount — as a generalization of our previous attempts — *gives unambiguous position.*

It proved to be the simplest and quickest solution when the object — which serves as a domain — was *sequentially decomposed into triangles* and we paid attention in which triangle could be found the point. *The position is the sum of the areas of previous triangles completed with a value* determines the pixel in the last triangle where the point is situated.

Naturally that would be the real solution if we could determine the position with binary searching, namely logarithmic searching. It would be necessary a kind of algorithm to this which could divide optional shape objects or group of objects sequentially in a way that the two parts of the area would be approximately the same size. Only that case can realize a searching method where the necessary elements of objects are minimum.

We tested also the version when 2D binary tree was used instead of quad-tree. According to our experiences in both cases relatively much time is required for solution. For this reason we tried the inorder traversal instead of preorder one. At such a filling method the average runtime and sheet number length were minimum.

Keywords: Virtual machine, storage, coordinate system, inteligent database, constraint-based reasoning

Introduction

Neither geodesy nor cartography was able to afford to create real consistent systems till digital geometry made available for them the own precise means for solution 2D and 3D tasks (ROSENFELD, 1979).

In digital geometry out of select resolution there is no need to compromise: there is no construction error, rounding error at all, no error accumulation, neither inconsistency nor neglecting. The image of digital (digitized) space is a *raster map,* which contrasts with traditional maps consist of finite number and finite size points, mostly squares. The same is valid for the lines, line segments and other map elements, too.

The 'empty' raster map which consists of raster points of suitable size and number according to the current requirements and possibilities constructs a common geometric event space both geodesy and cartography. Single 'empty' raster point has *elementary geometric information,* a figure consists of more 'empty' raster points has *complex geometric information.*

This latter essentially is the common collective noun of the usual location, shape, size, orientation et cetera.

Data Structure

Because the most simple way to store the content of a 2D map is a quadratic matrix therefore the $n \times n$ size *two dimensional array* consisting of n rows and columns is the most simple form which suits best to the idea of traditional map sheet in a computer (BURTON – KOLLIAS – KOLLIAS, 1987). We can refer to an individual raster element not only with its coordinate (namely telling which row and column section point is in the array) but one can use any code based on another neighbourhood relationship. Essentially the neighbourhood relationship belonging to tree structure base of every hierarchical system, such as administrative, cartographic or any kind of which defined by more digit number one or more dimensional number system (*Fig. 1*).

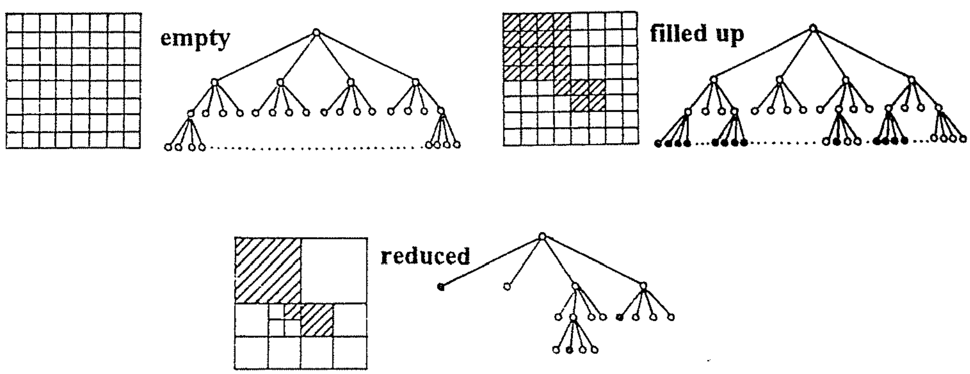


Fig. 1. Quadtrees

As we know the sheet number of Unified National Mapping System in Hungary (EOTR) is split the original domain in quarters so the digits of sheet numbers form four digit numerical systems. Applying this procedure such square shape raster array where length of side is integer power of '2' so one can refer to 4, 16, 64 ... size arrays individually using sheet number. Certainly by using shorter sheet number we can refer to larger sheet which is imposssible directly in case of using array structure. The solution above which is called quadtree (SAMET – WEBBER, 1983, 1985; SAMET, 1984) is the simplest and wide-spread version among tree structures. The most effective means of structured location determination as it is called 'positiontree' or 'spatialtree' is based on a quadtree also. However, the simplest and most efficient solution would be the 'bintree' (binary tree).

Position Determination

If we would like to refer to a certain point of a polygon (to the part of cm^2 of parcel, building et cetera) usually we should have such a circumspect and economical way as we used to refer to a certain day of a certain month. *The most important principle: we have to measure the 'spatial point' and 'spatial interval' one and the same unit* or we should use such kinds of units where the simple conversion is solvable vice versa. This is the basic requirement of system theory, however, the coordinates do not satisfy these requirements consequently they are not the most suitable for LIS or GIS systems at all. For determination of point positions interpreted on plumb line the meter, dm, cm, mm are correct because the one dimensional 'spatial interval' (the distance) has been measured with the same units. For determination of point positions in the case of 2D it must have to measure in m^2, dm^2, cm^2, mm^2 (namely in subarea) because the two dimensional 'spatial interval' (the area) has been measured with the same unit also. Point positions in 3D must have to measure with m^3, dm^3, cm^3, mm^3 (namely in subvolume) because the three dimensional 'spatial period' (the volume) has been measured with the same unit, too. Following this principle it is possible to determine a point position in percentage or per thousand of its domain (area or volume) or using any kind of ratio also. In other terms *the position should be dimensionless number.* It is necessary *to give the point positions directly* instead of indirectly. Because the coordinate always fixes a foot position on convenient coordinate axes consequently it does not *answer for the question 'where is the point?', the answer refers to the location of foot points.* 'Simplicity' and 'unity' are the two most important characteristics of every real 'long term' solution as of the up-to-date solution, too. It would be the most simple way according to the days in a calendar, the seconds, or the buildings in the streets to number the raster elements. In order to distinguish from parcel number, house number and lots of other kinds of number we will call it *position number* (KÁDÁR – PAPP, 1994).

As far back as 1899 George Pick an American mathematician worked out simple and precise formulas and theses in support of the above described ideas in mathematical way. The famous formula $T = B + K/2 - 1$ expresses the area of an arbitrary shape polygon as a function of number K and B the pixels which are on the polygon boundary or inside the polygon (KÁRTESZI, 1966). The number values of T, B and K assigning the cardinality of a certain kind of geometric type are set or subset. Of course the position of raster elements is system sensitive, for example the position of signed element is 103rd *(Fig. 2a)*, becomes 52nd (*Fig. 2b*) and finally gets 12th position number (*Fig. 2c*).

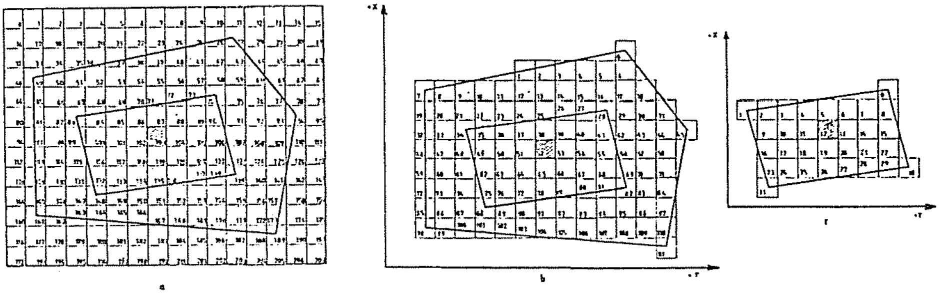


Fig. 2. Nested position numbering system

It is possible to attach one or more 2D or 3D coordinate systems to every polygon or polyhedron and when it is necessary (but only at that occasion) of using simple formulas we can compute coordinates from position numbers. Two or three coordinates can be computed from one position number. Thus the position number has close connection with the point number. *The position number attaches to every potentially possible point and the point number is attached to the existing control points.* When we want to determine the positions from x, y coordinates the only thing which we have to know is the positions of vertices of polygon defining the domain of position system. We need the same data also when we want to determine the coordinates from position number. Consequently, *every structured position system of such a recursively handles tree structure where the geometric data are only in the root of tree namely the size of rectangle or square which covers the whole polygon.* The position system needs the same size of memory in a computer as the position of polygon vertex system. However, this does not mean that there is additional memory required. Already every existing GIS/LIS system contains this information virtually in form of coordinates. We should have to realize and comprehend the possible resulting advantages from reorganizing of data structure belonging to system and application.

Digital Generalization of the Traditional Circumference and Area Concepts

We initiate the following micro structure for gapless connection of polygons (KÁDÁR, 1992). It could be seen a general quadrilateral as raster polygon sets together with the so-called digital boundary line sets (*Fig. 3*). Every set can be generated by computer software using coordinate list containing four points without any additional information.

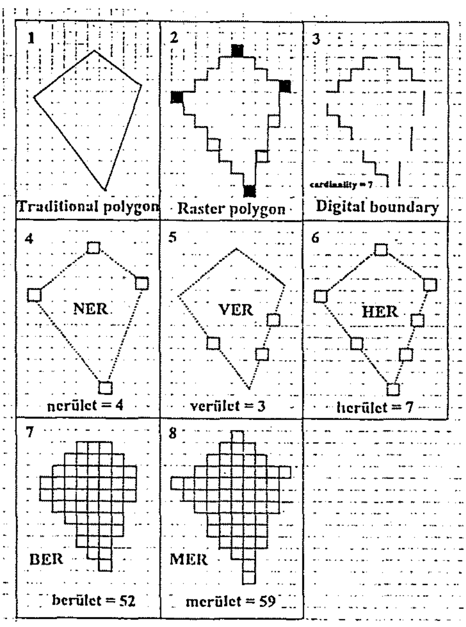


Fig. 3.

We enlarged the traditional concepts of circumference family with four elements and area family with two elements and changed only the first letters of the Hungarian names at the creation of the new names. The following list contains the new concepts and their short explanations (MIZSEI – KÁDÁR – ZALEZSÁK, 1989).

Circumference

1. *Kerület* (circumference) Name: *KER* Value: *k* or *kerület*

$$k = h + z$$

2. *Herület:* Set name: *HER* Cardinality: h or *herület*
   h = the amount of border points what is $h = k$ in the case of well oriented rectangle, in general case it is different from one another.
3. *Zerület:* Set name: *ZER* Cardinality: z or *zerület*

$$z = h - k$$

   we use the first letter of zero for their marking
4. *Nerület:* Set name: *NER* Cardinality: n or *nerület*
   n = the amount of vertex set of the polygon
5. *Verület:* Set name: *VER* Cardinality: v or *verület*
   v = the amount of raster elements which central points fit just (symmetrically) to the boundary line of the polygon

Area

6. *Terület* (area) Name: *TER* Value: t or *terület*

$$t = b + \frac{h}{2} - 1$$

7. *Berület:* Set name: *BER* Cardinality: b or *berület*
   b = the amount of the inside raster elements
8. *Merület:* Set name: *MER* Cardinality: m or *merület*
   m = the amount of all raster elements of the polygon

One part of this new idea can be found in Pick formula (b, h) but k has been used instead of h. The relations between above mentioned raster sets could be seen on *Fig 4*.

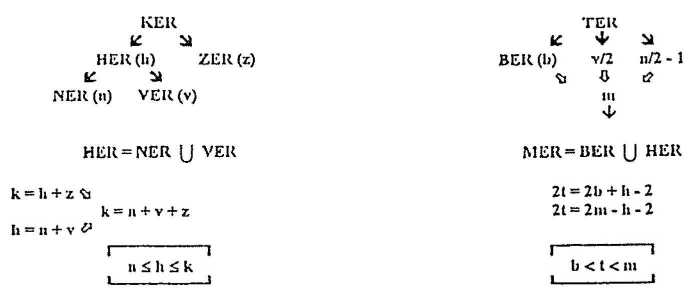


Fig. 4. Digital generalization of the traditional circumference and area concepts

To represent the above mentioned ideas a demonstrative program was made which draws a small polygon net to the screen from the data of an optional co-ordinate line and polygon list asked in interactive way, then calculates the characteristics of plots concerning to the area family and circumference family and draws into the suitable polygon (screen copy).

The *'digital straight boundary'* has very close connection with digital line (ROSENFELD, 1974) but is not identical with it. The digital line (*Fig. 5a*) can be considered like demarcation line but the digital boundary does not contain 'nowhere belong' raster elements rather 'border' and not 'boundary'. The production of digital boundary is happen with the easy recursion or circuit like digital line. The digital boundary projects a traditional polygon like equal area projection *(Fig. 5b)*.

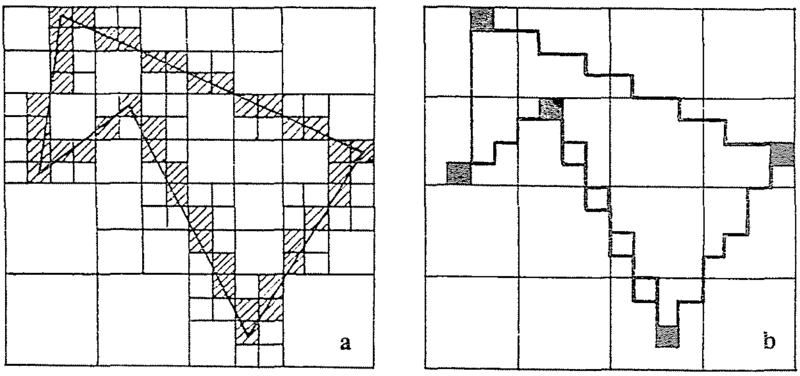


Fig. 5. *a*) Digital straight lines formed rasterpolygon, *b*) Digital straight boundary

Every digital straight boundary is symmetrical its centre. Consequently the parts which are taken off (add) from the area of neighbouring polygon situated on the right side of symmetry centre are given back (take off) from the other half which situated on the left side of symmetry centre. So one condition of equal area projection is satisfied. The other condition follows directly from Pick theorem: the sum of square sections cutting out from n vertices polygon equals exactly $(n-2)\frac{1}{2}$ the difference between berület and area. In fact the $(n-2)\frac{1}{2}$ corresponds to the $(n-2)180°$ or $(n-2)\pi$ well-known formula but in discrete geometry the angle has been used in terms of fraction parts in the area of raster element. In case of triangle this can be seen in *Fig 6.*

A question comes up: why this 'hair-splitting' is necessary, why not the time-honoured methods such as area, circumference 2D and 3D co-

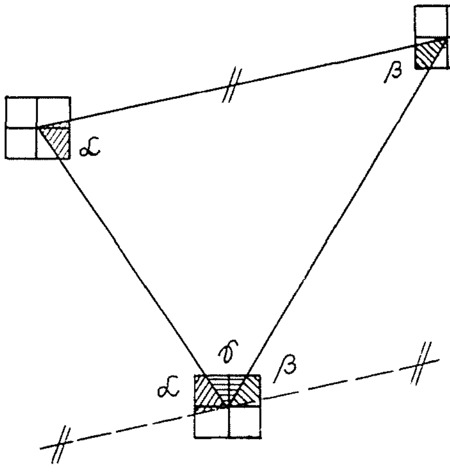


Fig. 6.

ordinates are suitable? Data bases — according to their definitions and standards — require the use of very clear domains and associated data values. In such a kind of systems frame error, rounding error it cannot happen more frequently so their consequences do not arise at all. This statement is completely independent from the user's desired accuracy in an existing GIS/LIS system or at the reliability of available data.

Test Results Using Subrange Data Type

First Experiment

Firstly we transform the polyfill function in a graphic software package in such a way that during the filling procedure the function counts the pixels from zero and then stops at a desired point, which we wanted to determine with subrange data type, instead of coordinates (see Appendix A1). If the given point is the same pixel where the filling begins then the 'position' is zero, and if the last pixel then the value of the position is very close to the polygon area. The difference depends on the domain in which the filling occurs in case as

- the polygon is an *open 2D interval,* namely just those pixels get colour where centre is inside the theoretical boundary of polygon (the domain of subrange data type *always smaller* than the area of polygon) (see Appendix A2)

- the polygon is *a close 2D interval,* just those pixels get colour which are inside the polygon or on its boundary (the domain of subrange data type are *always larger* than the area) (see Appendix A3)
- the filling method is similar to one that has almost become standard in digital geometry, namely one half of those pixels which are on the polygon border will belong to the domains of neighbouring polygons and the other half will take over from the neighbours. In such a case the domain of subrange data type is *smaller or greater,* but perhaps it should be *exactly equal* to the area of the polygon.

In the case of the polygon system the above described subrange data type can be used in the same way as the original polyfill function works if for example inside a certain given area we would colour the whole forest or fish-pond. If the function does not find the desired point during the filling procedure inside the first polygon then the counting will continue at the following polygon skipping the empty domains between the polygons. According to the above mentioned, subrange data type is in practice a 2D domain which *consists of more parts,* may be *quite irregular* with *rugged edges* and *holes,* moreover *inside holes* may be possibly one or more *other polygons.*

Second Experiment

We modified the well-known structure of Quad-tree and R-tree for another experiment series so it will be suitable for effective location (see Appendix B), because without modifying, the leaf generating procedure will *fill completely the given object with different size leafs.* If a monitoring command stops the filling procedure at the point which we wanted to localize and *summarize the area of generated leafs* then this amount — as a generalisation of our previous attempts — *gives unambiguous position number.* We tested such a version also when 2D binary tree was used instead of quad-tree. According to our experiences in both cases relatively much time is required for solution. For this reason we tried the inorder traversal instead of preorder one. At such filling method the average runtime and sheet number length was at minimum.

Third Experiment

It proved to be the simplest and quickest solution when the object — which serves as domain — was *sequentially decomposed into triangle* and we paid attention in which triangle could we find the point (see Appendix C1 and

C2). *The position is the sum of the areas of previous triangles completed with a position value* determined by the pixel in the last triangle where the point is situated. It is worth to mention for the sake of the polygon type that can be seen in Appendix C and would be expedient to work out experimentally another decomposition method which decomposes the polygon into *general size rectangles* instead of triangles. Applying such a method the figure in Appendix C would cover 9 rectangles instead of 34 triangles.

Fourth Experiment

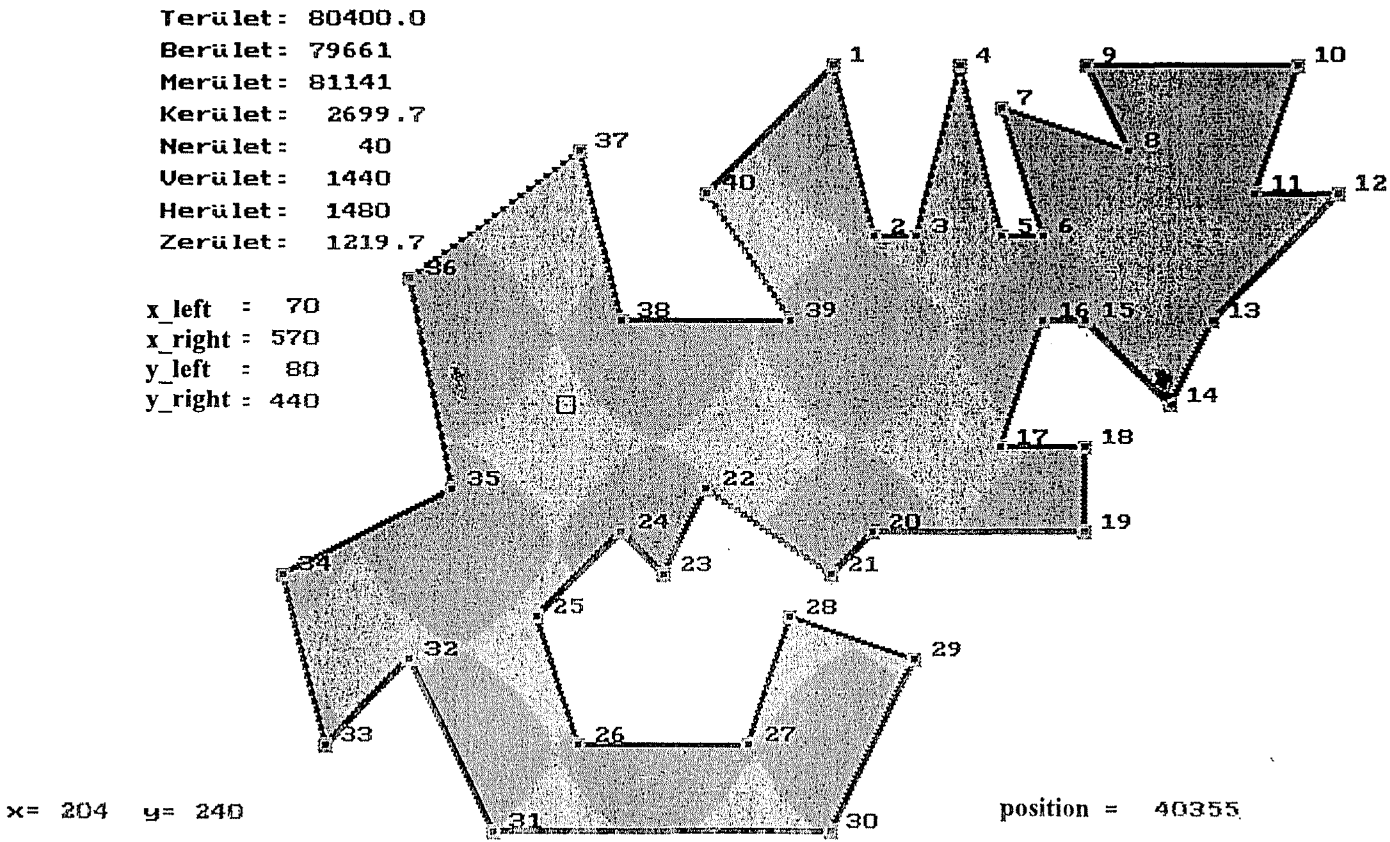
If an object (parcels with identical agricultural land-use or owner) consists of more parts moreover inside the individual polygons can be found so-called 'lakes' and inside this so-called 'island' which borders are formed by polygons we work with 'generalized polygons'. In Appendix D examples can be found.

We feel that the realization of *redundancy-free geometric data compression* turns from pipe-dream into reality and hereby will possibility to develop highly structured position systems.

Appendix

A1

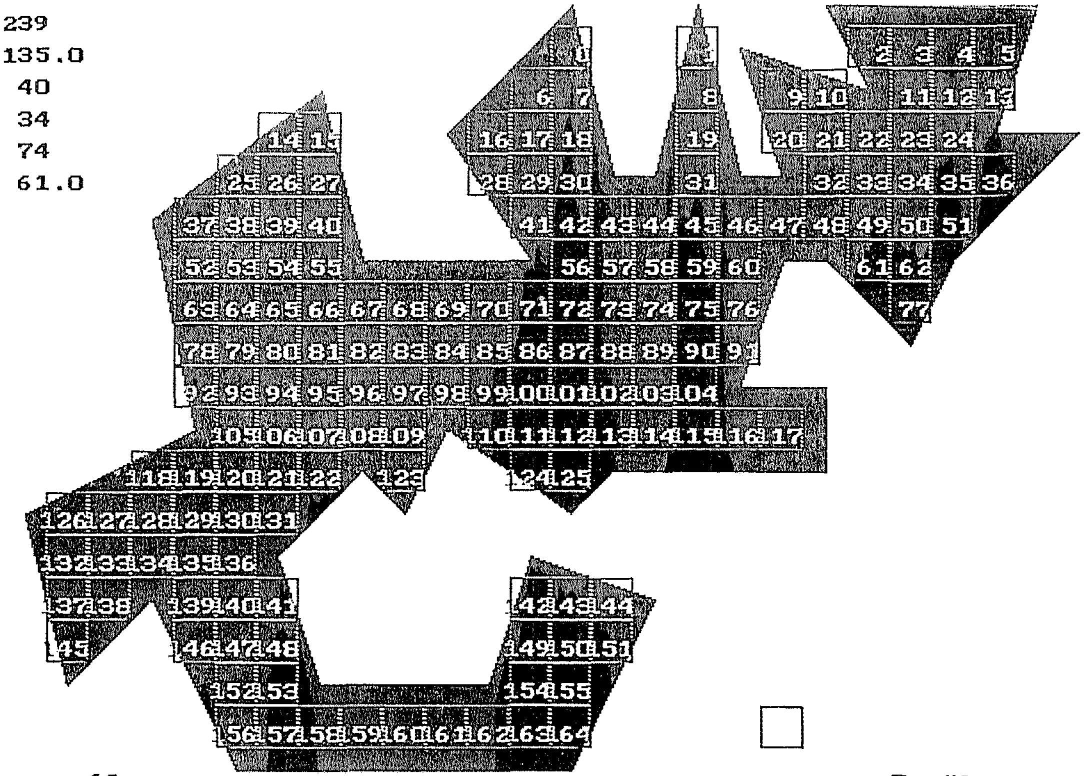
Polygon Reference System
(P-R-S)
instead of coordinates: location



Polygon Reference System
(P-R-S)
instead of coordinates: location

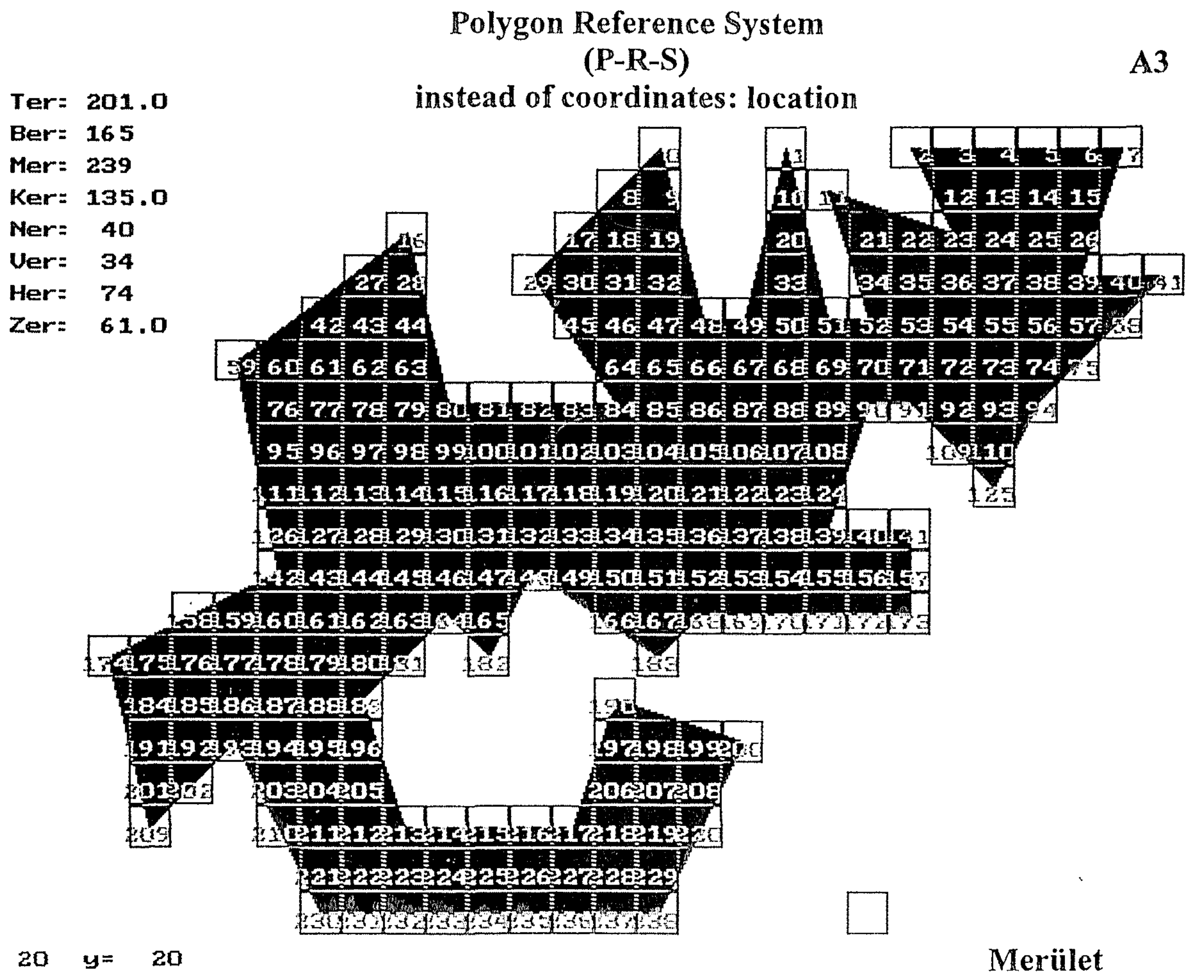
A2

Ter: 201.0
Ber: 165
Mer: 239
Ker: 135.0
Ner: 40
Ver: 34
Her: 74
Zer: 61.0

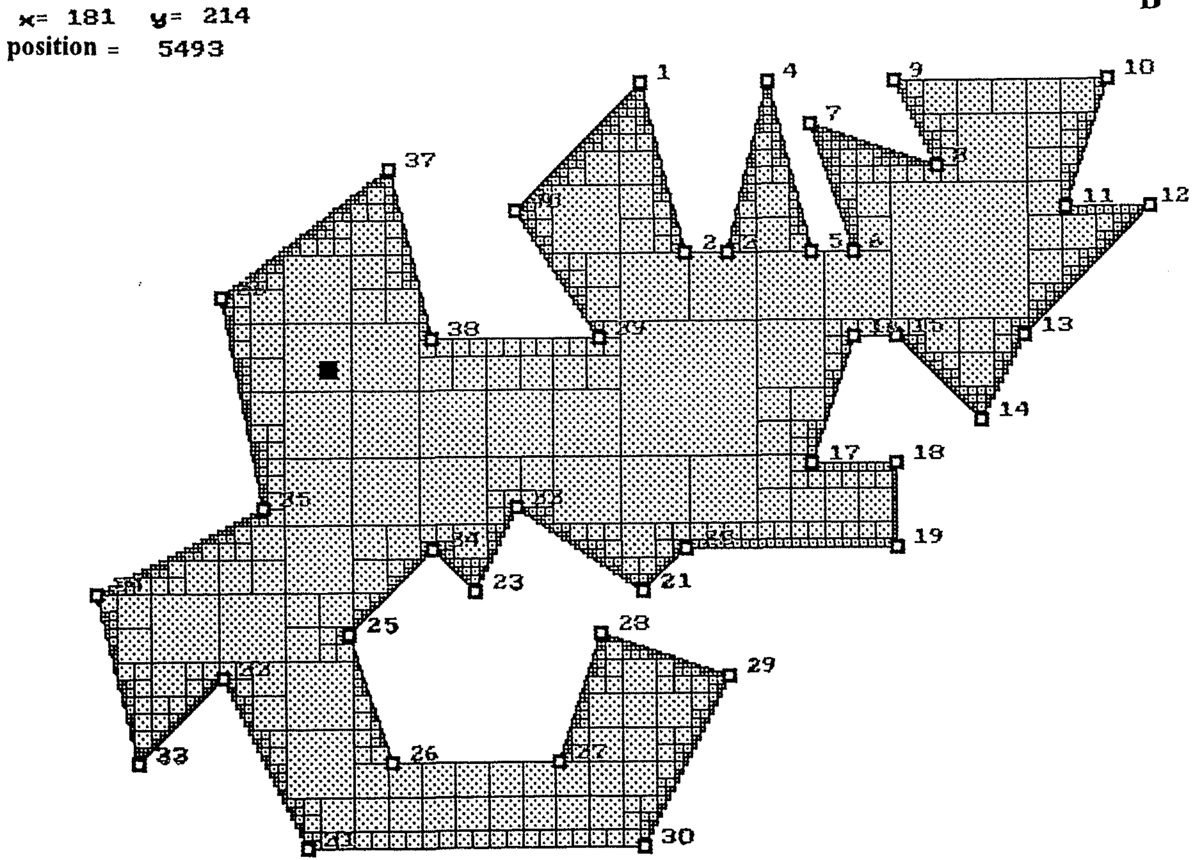


x= 20 y= 19

Berület


Polygon Reference System
(P-R-S)
instead of coordinates: location
A3
Ter: 201.0
Ber: 165
Mer: 239
Ker: 135.0
Ner: 40
Ver: 34
Her: 74
Zer: 61.0
20 y= 20
Merület

B


x= 181 y= 214
position = 5493
1
4
7
9
10
8
11
12
2
5
13
14
17
18
19
21
23
25
28
29
26
30
33
37
38

C

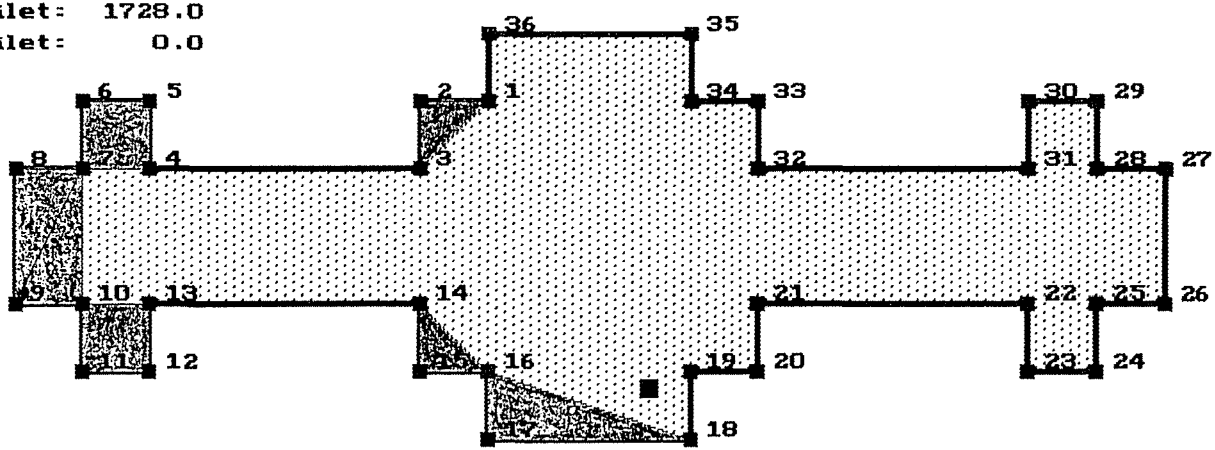
Data of POLYGON:

Terület: 55296.0
Berület: 54433
Merület: 56161
Kerület: 1728.0
Nerület: 36
Verület: 1692.0
Herület: 1728.0
Zerület: 0.0

Polygon Reference System
(P-R-S)
LOCUS instead of Coordinates
position are within BERÜLET

Data of TRIANGLE no.
16-18-19
Terület= 1536.0
Berület= 1457
Herület= 160

Data of DIAGONAL no.
16-19
Verület= 95

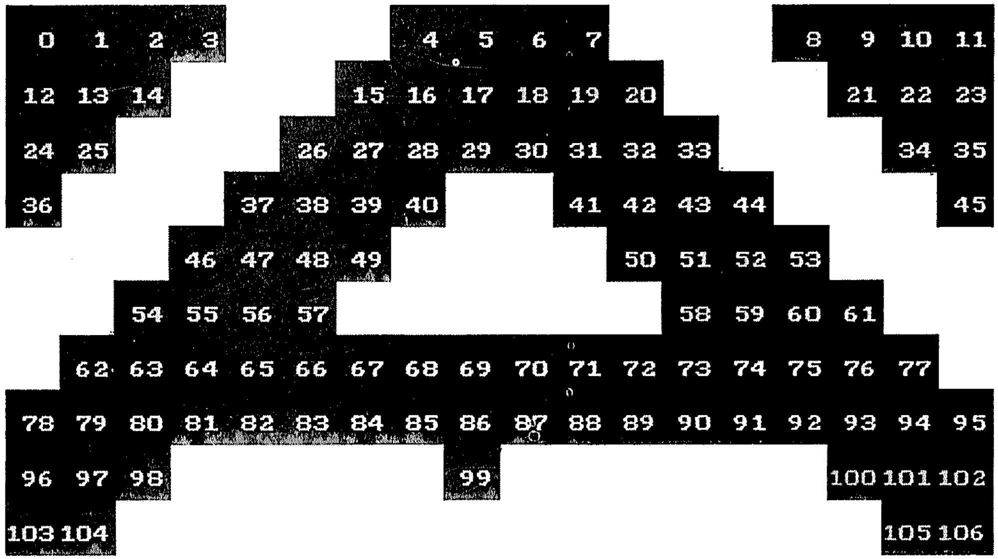


COORDINATES:
x= 348 y= 312
position = 7112

D

Polygon Reference System
(P-R-S)
instead of coordinates: position

Ter: 69.0
Ber: 35
Mer: 107
Her: 72
Ner: 25
Ver: 47
Ker: 85.255
Zer: 13.255



x= 21 y= 11 position := 106

References

SAMET, H.– WEBBER, R. E.: Using Quadtrees to Represent Polygonal Maps. *IEEE Transactions* 1983. CH. 1981–1/83, pp. 127–132.

SAMET, H.: The Quadtree and Related Hierarchical Data Structures. *Computing Surveys,* Vol. 16. No. 2, 1984.

SAMET, H. – WEBBER, R. E.: Storing a Collection of Polygons Using Quadtrees. *ACM Transactions on Graphics,* Vol. 4, No. 3, July 1985. pp. 182–222.

ROSENFELD, A.: Digital Straight Line Segments. *IEEE Transactions on Computer,* Vol. 23, 1974.

ROSENFELD, A.: Picture Languages. Academic Press, New York – San Francisco – London, 1979. Chapter 2. Digital geometry, pp. 7–39.

BURTON, F. W. – KOLLIAS, V. J. – KOLLIAS, J. G.: A General Pascal Program for Map Overlay of Quadtrees and Related Problems. *The Computer Journal,* London, Vol. 30, No. 4, 1987.

KÁDÁR, I.: Natural Dimensions of the Position. (In Hungarian), *Geodézia és Kartográfia,* 1992, pp. 417–424.

KÁDÁR, I. – PAPP, E.: Subrange Data Type Applied to Definition Virtual Coordinate Systems. Presented paper on *'Third Seminar European Land Information System '94',* Delft, The Netherlands, September 12–13, 1994.

MIZSEI, J. – KÁDÁR, I. – ZALEZSÁK, T.: Geodetic and Cartographic Operations among Rastermaps Organized in Tree-Structures (In Hungarian), *Geodézia és Kartográfia,* Vol. 41, No. 1, 1989, pp. 39–53.

KÁRTESZI, F.: Szemléletes geometria, Gondolat Kiadó, Budapest, 1966.