

ENCODING AND DECODING OF PLANAR MAPS THROUGH CONFORMING DELAUNAY TRIANGULATIONS

Edward Verbree

Delft University of Technology, Research Institute OTB, Section GIS technology, Jaffalaan 9, 2628 BX, Delft,
e.verbree@otb.tudelft.nl

Working Group II/3, II/6

KEY WORDS: Planer Maps, Conforming Delaunay Triangulations

ABSTRACT:

This paper describes a method to represent a Planar Map (PM) through a Conforming Delaunay Triangulation (CDT) with applications in a server-client environment. At the server a CDT of the edges of the PM is determined. As the PM is now embedded by the CDT it is sufficient to send to the client the list of coordinates of the CDT nodes and an efficient encoded bitmap of the corresponding PM-CDT edges. The client determines a Delaunay Triangulation (DT) of the received list of coordinates of the CDT nodes. The DT at the client side is – in principle – equivalent to the CDT at the server side. The edges of the PM are found within this DT by the decoding of the bitmap of the corresponding PM-CDT edges.

1. INTRODUCTION

1.1 Support of Planar Maps by the OGC

One of the main principles in distributed Geographical Information Systems (GIS) is the server-client architecture. According to the Open Geospatial Consortium (OGC) a Web Map Server (WMS) allows clients to retrieve map portrayals (images) of the data features through the Internet. In a similar way access to the features is made possible by a Web Feature Server (WFS). The requested features from a specified layer and within a certain extent are encoded in the Geographical Markup Language (GML) at the server. The client has to understand the semantics of the GML to process the features, i.e. know to handle point, lines and polygon simple feature types. The necessary logic is with respect to these simple feature geometries not too hard, as the coordinates are part of the simple feature definition. The coding and the understanding of Planar Maps is however more difficult to achieve.

Planar Maps are fundamental structures in computational geometry. They are used to represent the subdivision of the plane into regions and have numerous applications. Planar maps by themselves can be used to represent geographic maps. They also serve as fundamental structures on which more involved geometric data structures are constructed (Flato, 2000). A

Within GIS most often each region is paired to an identifier that is used to link the geometry to a set of attribute values. Within GML Planar Maps do not have their own normative schema yet, but they can be considered as special types of polygon coverages.

Coverages are described as follows (OGC_03-105r1, par 19.1, URL): “Coverages support mapping from a spatio-temporal domain to attribute values where attribute types are common to all geographic positions within the spatio-temporal domain. A spatio-temporal domain consists of a collection of direct positions in a coordinate space. Examples of coverages include rasters, triangulated irregular networks, point coverages, and polygon coverages. Coverages are the prevailing data

structures in a number of application areas, such as remote sensing, meteorology, and bathymetric, elevation, soil, and vegetation mapping.”

“The information describing a coverage is represented in one of the two ways: a) as a set of discrete location-value pairs, or b) as a description of the spatio-temporal domain (multi-geometry, grid) and a description of the set of values from the range, together with a method or rule (which may be implicit) that assigns a value from the range set to each position within the domain. The first method only applies to domains that are partitioned into discrete components. This representation may be realised in GML as a homogeneous feature collection (i.e. all the features have the same set of properties), where the set of location from the features compose the domain (as gml_location may refer to any geometry, not just points).”

If we restrict the composition of the domain such that it embeds the space into non-overlapping adjacent regions, where each region has a value (i.e. an identifier) the polygon coverage could be used to represent planar maps. There is however a draw back regarding to these polygon coverages: for coverages whose domain is composed of a large set of locations this explicit representation may be bulky. But, more important, the GML coverage encoding is at the moment not addressed within the implementation of GML.

Instead of using the polygon coverage, GML offers the topology primitive *face*. A planar map is represented by a set of non-overlapping, adjacent faces, where each face is defined by its boundary, which consists of a list of connected and directed edges. This option has the disadvantage that the client has to validate the received set of faces to be a planar map or not, as there is no option to guarantee that the faces represent together a planar map.

1.2 Representation of Planar Maps

The Computational Geometry Algorithms Library (CGAL) gives a definition of planar maps: “Planar maps are embeddings of topological maps into the plane. A planar map

subdivides the plane into vertices, edges, and faces” (CGAL Basic Library Manual – 2D Planar Maps, URL).

This definition deals with topological maps: “A *topological map* is a graph that consists of vertices, edges, faces and an incidence relation on them. Each edge is represented by two halfedges with opposite orientations. A face of the topological map is defined by the ordered circular sequences (inner and outer) of halfedges along its boundary”. “For a topological map, its *Double Connected Edge List (DCEL)* representation consists of a connected list of halfedges for every Connected Component of the Boundary (CCB) of every face in the subdivision, with additional incidence information that enables us to traverse the subdivision” (CGAL Basic Library Manual – Topological Map, URL), see Figure 1.

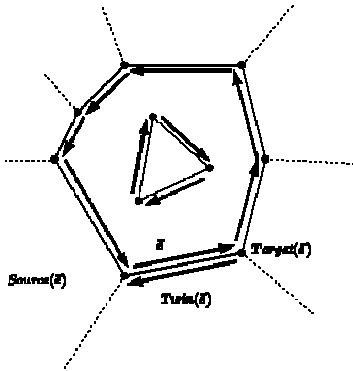


Figure 1: For each halfedge the DCEL stores a pointer to its twin halfedge and to the next halfedge around its incident face

If the non-overlapping adjacent polygons of a Planar Map are all triangles, then we deal with a Triangulated Irregular Network. A TIN can be regarded as a special case of a topological map, according to the definition: “A *triangulation* is a 2-dimensional simplicial complex which is pure connected and without singularities. Thus a triangulation can be viewed as a collection of triangular faces, such that two faces either have an empty intersection or share an edge or a vertex” (CGAL Basic Library Manual – 2D Triangulations, URL). For both the planar map and the TIN a DCEL representation can be used, but in many cases, i.e. in CGAL, as the triangulation is a set of triangular faces with constant-size complexity, the triangulations are not implemented as a layer on top of a planar map. CGAL uses instead a proper internal representation of triangulations. The basic elements of the representation are vertices and faces. Each triangular face gives access to its three incident vertices and to its three adjacent faces. Each vertex gives access to one of its incident faces and through that face to the circular list of its incident faces.

1.3 Representing Planar Maps through Conforming Delaunay Triangulations

Both a Planar Map (PM) and a Triangulation consist of a collection of non-overlapping, adjacent region / faces. In the Triangulation however these faces are just simple triangles. It should be possible to represent each region of the PM by one or more triangles, or the other way around, one or more adjacent triangles of the Triangulation represent a face of the PM.

In this encoding process it should be guaranteed that each face of the Planar Map has its counterpart in one or more triangles of

the Triangulation. Applying a Conforming Delaunay Triangulation can ensure this requirement.

Most Triangulation implementations take as their input a *Planar Straight Line Graph (PSLG)*. A PSLG is a set of vertices and segments. A segment is an edge that must be represented by a sequence of contiguous edges in the final mesh. By definition, a PSLG is required to contain both endpoints of every segment it contains, and a segment may intersect (touch) vertices and other segments only at its endpoints, see Figure 2.

Planar Map partitions the embedding space into non-overlapping adjacent regions. Such maps are planar subdivision induced by planar embeddings of graphs. The embedding of a node of the graph is called a vertex, and the embedding of an arc is called an edge (de Berg, 2000).

If a Planar Map is restricted to embeddings where every edge is a straight-line segment, then a this PM is a special case of a PSLG. In this case a PSLG triangulation algorithm could be applied to mesh a Planar Map.

There exists a huge amount of references on Delaunay Triangulations; here the definitions as given by Jonathan Shewchuk are stated (Triangle, URL):

A *Delaunay triangulation (DT)* of a vertex set is a triangulation of the vertex set with the property that no vertex in the vertex set falls in the interior of the circumcircle (circle that passes through all three vertices) of any triangle in the triangulation.

A *Conforming Delaunay triangulation (CDT)* of a PSLG is a true Delaunay triangulation in which each PSLG segment may have been subdivided into several subsegments by the insertion of additional vertices, called *Steiner points*. Steiner points are necessary to allow the segments to exist in the mesh while maintaining the Delaunay property, see Figure 3.

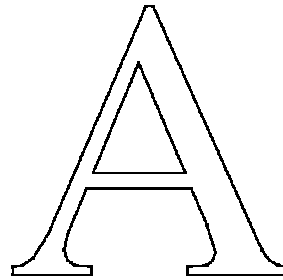


Figure 2: PSLG

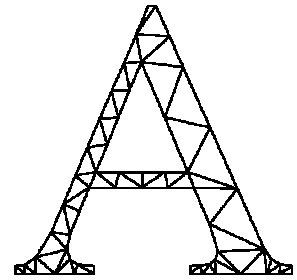


Figure 3: CDT

Where and how to add the Steiner points is the focus of extensive research with respect to refinement algorithms for triangular mesh generation (Shewchuk, 2001).

Each original, but now split, edge of the PSLG is then represented by one or more edges of the CDT. The original nodes of the PSLG plus the added Steiner points are input for the Delaunay triangulation algorithm of both the encoding and the decoding process.

1.4 Outline of this paper

Section 2 addresses the encoding of a Planar Map through a Conforming Delaunay Triangulation. Section 3 describes the

decoding process to restore the Planar Map. Section 4 shows some experimental results. Section 5 concludes with some conclusions and recommendations.

2. ENCODING THE PLANAR MAP THROUGH CONFORMING DELAUNAY TRIANGULATION

2.1 Encoding of the Planar Map

A possible way of encoding a Planar Map (PM) through a Conforming Delaunay Triangulation (CDT) will be explained by some figures.

Figure 4 shows an example of a Planar Map. Six faces are embedded in the space bounded by the bordered rectangle: the faces “M”, “A”, and “P” and the ‘inner-faces’ “A-”, “P-”. The remaining sixth face around “M”, “A”, and “P” is denoted with “O-”. Each face is defined by the ordered circular sequences (inner and outer) of straight edges along its boundary.

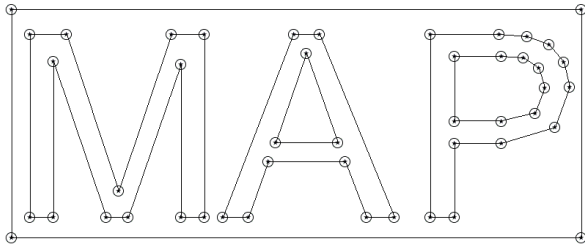


Figure 4: Planer Map at the server

In Figure 5 the Conforming Delaunay Triangulation (CDT) of this Planar Map (PM) is shown. Note that the PM edges are split by Steiner points if this is necessary to obey the Delaunay circum circle criterion. See also the explanation of the Triangle algorithm of Jonathan Shewshuk (Triangle, URL).

Each face of the PM is now represented by one or more of the triangles of the CDT.

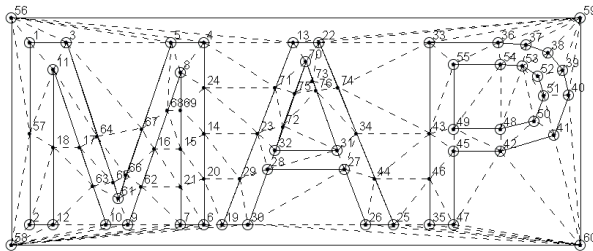


Figure 5: Conforming Delaunay Triangulation at the server

Figure 6 shows in more detail “A-”, or PM_F(32,31,70), which is represented by CDT triangles CDT_F(32,31,72), CDT_F(72,21,76), CDT_F(72,76,75), CDT_F(75,76,73), and CDT_F(73,70,75).

2.2 Relating the edges of the Planar Map and the edges of the Conforming Delaunay Triangulation

Each edge of the PM is represented by one or more of edges of the CDT.

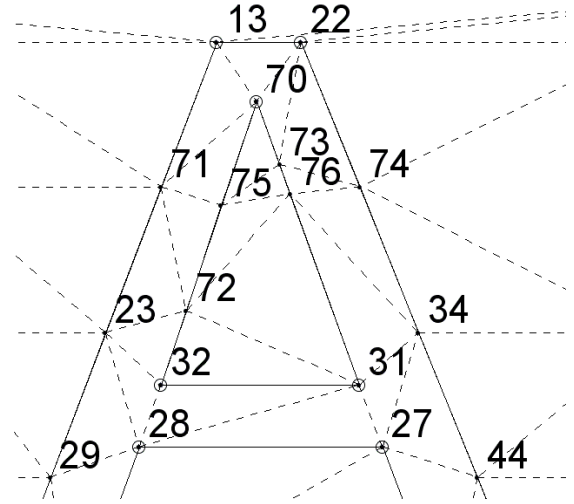


Figure 6: Relation between the edges of the Planar Map and the Conforming Delaunay Triangulation

Figure 6 shows the PM edges of “A-”: PM_E(31,32), PM_E(31,70), and PM_E(32,70). PM_E(31,32) is represented by the edge CDT_E(31,32); PM_E(31,70) is represented by edges CDT_E(31,76), CDT_E(73,76), and CDT_E(70,73); PM_E(32,70) is represented by edges CDT_E(70,75), CDT_E(72,75) and CDT_E(32,72).

The dashed CDT edges are non PM edges.

The nodes CDT_N(76), CDT_N(73), CDT_N(75), and CDT_N(72) are the Steiner points to obey the Conforming Delaunay criterion.

2.3 Encoding the PM_Edges

The encoding relates the edges of the CDT with the edges of the PM. The CDT_edges are sorted according to their from_node and then to their to_node, where to_node > from_node. So only half of the CDT_Edges are stored. If a CDT_Edge represents (a part of) a PM_Edge it is marked as “True”, else as “False”.

For example the CDT_Edges from CDT_N(1) are CDT_E(1,3), CDT_E(1,11), CDT_E(1,56), and CDT_E(1,57). CDT_E(1,3) represents PM_E(1,3) and is thus marked as “True”. CDT_E(1,11) has no representing PM edge, so it is marked as “False”. CDT_E(1,56) is also marked as “False”, and as CDT_E(1,57) is part of MP_E(1,2) it is marked as “True”.

As the CDT_Edges are sorted, these edges are stored more efficient as a kind of a selection edge bitmap. CDT_E(1,3) is set to “1”, CDT_E(1,11) is set to “0”, CDT_E(1,56) is set to “0”, and CDT_E(1,57) is set to “1”.

The full coding bitmap that can be compressed in a very efficient way, of the CDT_Edges of the example Planar Map “MAP” is:

```
1001101000010100100001010100010001110010001
1100100100010101000010101001010000011000000
0010010100010100100010000010000010011110000
11011010001000100010001000100010000000010000
0100011000110101110110110011001000100011001
000100
```

For each face one identification point has to be determined. This seed point has to be inside the face and will be used to restore the PM_Faces within the decoding process. These seed points are not shown in the figures.

The server sent the CDT_Nodes, the selection edge bitmap, the list of seed points at request to the client.

3. DECODING TO THE PLANAR MAP THROUGH DELAUNAY TRIANGULATION

3.1 Reconstructing the Conforming Delaunay Triangulation

The CDT_Nodes, the selection edge bitmap and the list of seed points is received and processed by the client to reproduce the original Planar Map. The first step in the decoding is the creation of an identical Triangulation as created within the encoding step.

We will use the following, neat, property: as the encoded Conforming Delaunay Triangulation obeys the Delaunay criterion, any Delaunay Triangulator at the client side will produce, in principle, the identical result, see Figure 7:

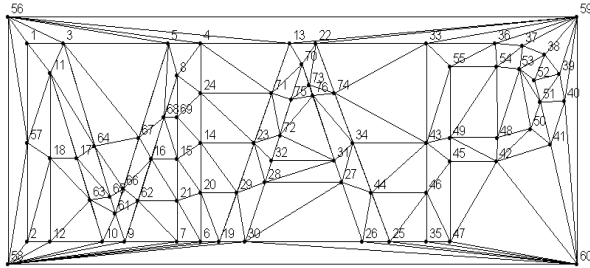


Figure 7: Delaunay Triangulation at the client

There are, however, some considerations to the choice of the applied Delaunay Triangulator to make. First off all, the Node identifiers of the reproduced DT should be identical to the original CDT. Otherwise the sorted list of DT_Edges at the client does not correspond with the sorted list of CDT_Edges at the server.

But, more important, if four or more nodes are at the same circum circle, the Delaunay criterion is free to choose which nodes to connect and a different set of DT_Edges is produced. One way to deal with that problem is to apply the same triangulation algorithm at the client side as use at the server side. A more fundamental approach is however to apply a kind of a unique Delaunay Triangulation in the sense that the same set of DT_Edges is produced at both sides.

3.2 Decoding the PM_Edges

Once the sorted list of DT_Edges is determined, the coding bitmap is set at all DT_Edges to indicate the PM_Edges, see Figure 8.

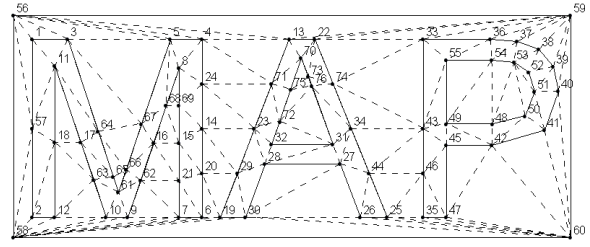


Figure 8: Decoded PM_Edges at the client

3.3 Decoding the PM_Faces

Once all PM_Edges are detected the PM_Faces are restored by the Seed_Points. Each Seed_Point is located inside a DT_Face. This DT_Face is given the identification value of the corresponding face of the PM. The DT_Edges of this DT_Face are checked whether or not to be set by the coding bitmap. If not, the adjacent DT_Face will get the same identification. This flood fill algorithm repeats until all DT_Faces do have their identification values.

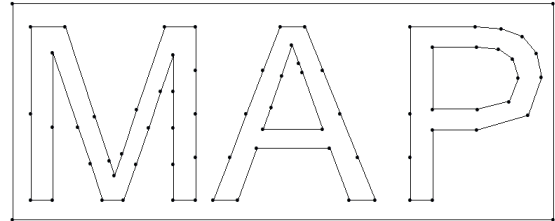


Figure 9: Decoded PM_faces at the client

The union of the DT_Faces with the same identification value will be the original PM_Face, see Figure 9.

4. EXPERIMENTAL RESULTS

The approach of encoding and decoding of a Planar Map, or any Planar Straight Line Graph, is proven to work through a test-implementation with two kinds of Conforming Delaunay Triangulators. The first triangulation is part of ESRI ArcView 3D Analyst (ESRI - 3D Analyst, URL), the second implementation makes use of 'Triangle' provided by Jonathan Shewshuk (Triangle, URL).

4.1 Test-case "the Netherlands"

The method as described in the previous sections is used to encode and decode the edges of the Planar Map of the Netherlands through a Conforming Delaunay Triangulation, see Figure 10.

As shown in the detail, see Figure 11, only a few Steiner Points (i.e. nodes 1314, 1315, 1317 and 1318) had to be added to allow the edges of the Planer Map to exist as edges in the Triangulation while maintaining the Delaunay property.

The detail of the island of "Ameland", Figure 12, shows the conforming TIN edges, representing the PM edges in bold. The remaining TIN edges are dashed.

With respect to the first two nodes this representation is as follows:

- 1 2 true PM-edge
- 1 3 true PM-edge

1 4 false PM-edge
1 26 false PM-edge
1 1496 false PM-edge
2 26 false PM-edge
2 33 true PM-edge
2 1496 false PM-edge
2 1497 false PM-edge

Or more short the coding bitmap or the first two nodes reads:
110000100.

The full coding bitmap is sent in a compressed format together with the coordinate list of the nodes of the Conformal Delaunay Triangulation to the client.

The client performs a “standard” Delaunay Triangulation on these nodes and set the coding bitmap on the TIN edges. As the result all (sub)segments of the original Planar Map are found.

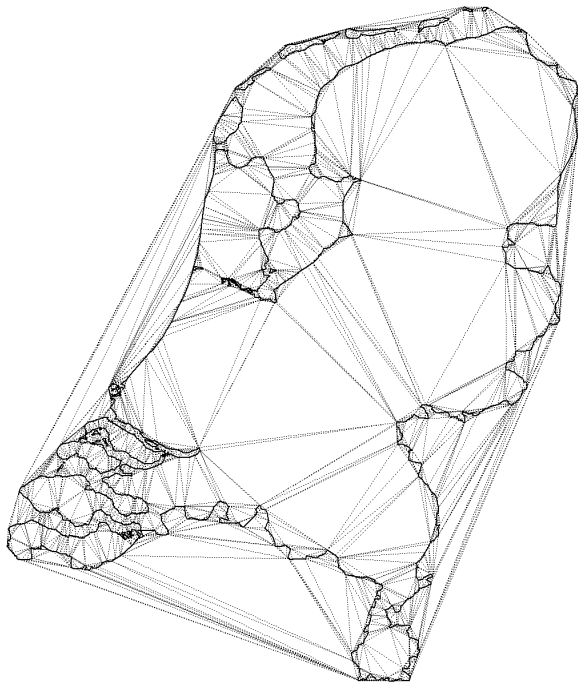


Figure 10: Conforming Delaunay Triangulation of "the Netherlands"

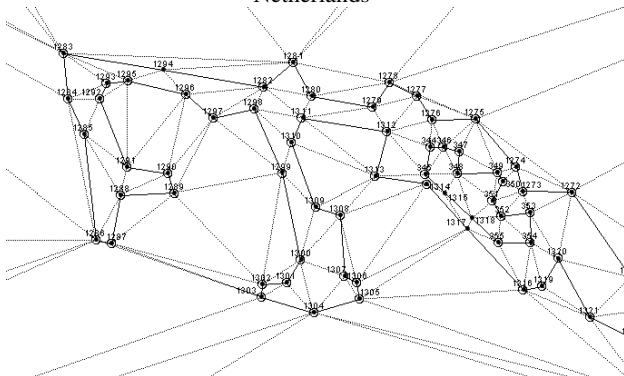


Figure 11: Detail of Conforming Delaunay Triangulation

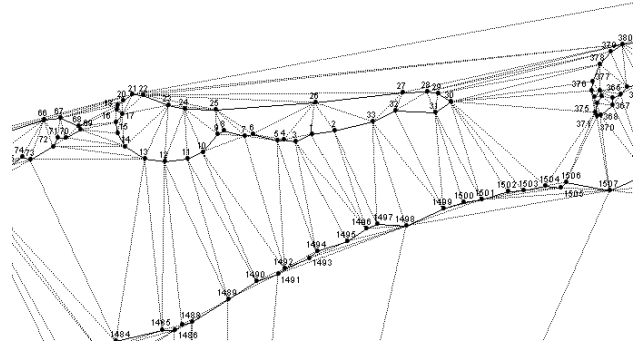


Figure 12: Detail on island "Ameland"



Figure 13: Decoded Edges of Planar Map of "the Netherlands"

5. CONCLUSIONS AND RECOMMENDATIONS

This paper presents a method to encode and decode a Planar Map (PM) by a Conforming Delaunay Triangulation (CDT). The geometry and topology of the CDT could be encoded and sent to a client. The client needs access to some basic Computation Geometry logic, like a standard Delaunay Triangulation to decode the Planar Map.

During the implementation phase it is shown that although a Delaunay Triangulation is unique in respect to the empty circum circle criterion, not each Delaunay Triangulator produces the same unique triangulation given the same point dataset. This ‘failure’ is due to the fact that if four or more points are co-circular they can be triangulated in several ways, all according the Delaunay criterion. As the representation of Planar Maps by Conforming Delaunay Triangulation relies heavily upon the uniqueness of a Delaunay Triangulator more research is necessary with respect to Unique Delaunay Triangulators.

The node numbers of the CDT should be according to the ordering of the list of coordinates of the CDT list as sent by the server. If not, the implicit linkage to the edge bitmap is lost. The decoding of the Delaunay Triangulation to a Planar Map relies also on the coding schema of the edge bitmap of the corresponding CDT-PM edges. As the slightest misinterpretation will cause the setting of the wrong PM edges within the triangulation, most case should be taken to avoid a misread or misinterpretation of this edge bitmap.

To avoid all these problems, it is an option to send the CDT itself to the client. To avoid heavy network load, an efficient compression algorithms, like Edgebreaker (Rossignac, 1999) could be applied.

One could argue about the aim of this method. It is clear that the Planar Map itself can be sent to the client by i.e. its winged edge datastructure. But if we want to support multi-scale representation of spatial data seamless, like progressive web transfer, self-adaptable visualisation and other applications, the TIN datastructure itself promises some fine characteristics.

Despite all these considerations, the encoding of Planar Maps by Conforming Delaunay Triangulations could be extended to the third dimension. Polyhedron boundary representations could be encoded and decoded through a conforming tetrahedronization.

6. ACKNOWLEDGEMENTS

The author wants to acknowledge Peter van Oosterom, Friso Penninga, Sander Oude Elberink and Elfriede Fendel for their contribution to this paper. Their remarks and considerations are highly appreciated, as are the suggestions for improvement of the reviewers of this paper. Remaining errors and other indistinctness are mine.

References:

Berg, de M, M. van Kreveld, M. Overmars, O. Schwarzkopf, 2000. *Computational Geometry: Algorithms and Applications*, second edition. Springer.

CGAL Basic Library Manuals - 2D Planar Maps, http://www.cgal.org/Manual/doc_html/cgal_manual/Planar_map_ref/Chapter_intro.html

CGAL Basic Library Manuals - Topological Maps, http://www.cs.tau.ac.il/CGAL/Manuals/doc_html/basic_lib/Topological_map/Chapter_main.html

CGAL Basic Library Manuals - 2D Triangulations, http://www.cs.tau.ac.il/CGAL/Manuals/doc_html/basic_lib/Triangulation_2/Chapter_main.html

ESRI - 3D Analyst, <http://www.esri.com/software/arcview/extensions/3danalyst/>

Flato, Eyal, Dan Halperin, Iddo Hanniel and Oren Nechushtan, 2000. The design and Implementation of Planar Maps in CGAL, The ACM Journal of Experimental Algorithmics, Volume 5, Article 13.

OGC_03-065r6, Web Coverage Service (WCS), Version 1.0.0

OGC_03-105r1, OpenGIS® Geography Markup Language (GML) Encoding Specification

Rossignac, Jarek, 1999. Edgebreaker: Connectivity compression for triangle meshes, IEEE Transactions on Visualization and Computer Graphics, Vol. 5, No. 1, January - March 1999

Shewchuk, Jonathan Richard. 2002. Delaunay Refinement Algorithms for Triangular Mesh Generation, Computational Geometry: Theory and Applications 22(1-3): 21-74.

Triangle, <http://www.cs.cmu.edu/~quake/triangle.html>