Implementation and Visualization of Discrete Computational Geometry Using Database Managers

Stella Orozco-Ochoa[#1], Oscar Ruiz-Salguero[*2], Carlos A. Cadavid†[3]
[#] Department of Informatics, Universidade de Vigo, Spain
[1]morozcoo@yahoo.com
Universidad EAFIT
[*]Laboratory of CAD CAM CAE,Universidad EAFIT, Colombia
[2]oruiz@eafit.edu.co
[†]Mathematics and Applications Group,Universidad EAFIT, Colombia
[3]ccadavid@eafit.edu.co

***Abstract*—Database manager systems (DBMS) have been traditionally used for handling economic and alpha-numeric information, related to business, government, education, health, urbanism, etc. Making 3D geometry and topology and their related algorithms the raw material and subject of DBMS is rare. Attempts have been made in the specific field of Geographic Information Systems (GIS). However, for historical and economical reasons 3D geometry and topology was appended on top of 2D entities. The mechanism found is usually the addition of 3D information in the form of attributes for, for example, vertical extrusions of 2D entities. The related algorithms usually handle the interrogations and constructs of GIS, even though some GIS systems contain3D geometry and Topology, in the sense of Geometric Modelling. This manuscript presents the usage of DBMS graph capabilities for approximation of hard core computational geometry algorithms. This rarely used approach has the advantage of avoiding degenerate situations, at the price of lower precision. Taping into the vast graph capabilities of DBMSs has the obvious advantage of large algorithm libraries, which in this case we apply to computational geometry. The lower geometric precision of graph-based DBMS algorithms do not hamper their application, in problems of dimensional reduction, mesh parameterization and segmentation, etc. This manuscript is also attractive in that it illustrates the articulation of freeware display systems (e.g. JavaView™) with DBMS for computational geometry applications.**

Keyword - Geometry Databases, Topology Operations, Computational Geometry, Graph Databases, Triangular Manifolds

GLOSSARY

TIN	Triangulated Irregular Network, used in this work to express terrain topography.
DBMS	Database Manager System
GIS	Geographic Information System
PL	Piecewise Linear
$\boldsymbol{M}$	PL 2-manifold triangular surface (also called *mesh*), which is also a planar graph embedded in $\boldsymbol{R^3}$.
$d_M(v_0,v_f)$	Geodesic distance, measured on mesh $\boldsymbol{M}$, between vertices $\boldsymbol{v_0}$ and $\boldsymbol{v_f}$ of $\boldsymbol{M}$.
$E_M(v,w)$	Number of constant length EDGEs of the minimal path between nodes $\boldsymbol{v}$ and $\boldsymbol{w}$ of graph $\boldsymbol{M}$.
$L_M(v,w)$	PL length of the minimal path between nodes $\boldsymbol{v}$ and $\boldsymbol{w}$ of graph $\boldsymbol{M}$.
$\delta(M)$	minimal EDGE length in a triangular mesh $\boldsymbol{M}$.
B-Rep	Boundary Representation (BODY, LUMPs, SHELLs, FACEs, LOOPs, EDGEs, VERTEX) of a solid object in $\boldsymbol{R^3}$. The parametric surfaces that carry the FACEs of a B-Rep are usually smooth ($\boldsymbol{C^1}$, $\boldsymbol{C^2}$), unless the model specifically requires a flat FACE.
Tr-BRep	B-Rep whose FACEs are exclusively triangles. In this case, the FACEs have no holes.
FACE	A connected region on a parametric surface $\boldsymbol{S}(u_1, u_2)\square\boldsymbol{R^3}$ (B-Rep context).
LOOP	Closed piecewise smooth curve in $\boldsymbol{R^3}$ (B-Rep context).
$\boldsymbol{S^2}$	Open unitary disk in $\boldsymbol{R^2}$ centred in (0,0).

I. INTRODUCTION

Algorithms and constructions in Computational Geometry are usually served by dedicated CAD or similar software or libraries. In the particular case of Piecewise Linear SHELLs, LOOPs, etc. (e.g. in triangular meshes), floating point computations are conducted to calculate discrete counterparts of smooth operators (normal vector, derivatives, geodesic curves, etc.). However, it is not frequent the usage of database operations and interrogations on the 3D mesh graph to respond constructive or boolean queries on the mesh. As an example, geodesic curves between vertices v_p and v_q are usually calculated by using the fact that their acceleration is always parallel to the vector locally normal to the mesh. As an example, geodesic curves are rarely approximated by finding a shortest path between nodes v_p and v_q of the mesh graph.

The usage of constructors and queries on the mesh graph has the advantage of avoiding degenerate cases (e.g. the geodesic hitting a mesh VERTEX or flowing along a triangle EDGE). The obvious disadvantage is that discrete graph operators are approximations coarser than the floating point computations on the already approximated PL mesh. However, these approximations are still good enough for mesh operations. As an example, the dimensionality reduction of IsoMap[20] can be used as a mesh parameterization tool and the shortest path approximation of geodesics in IsoMap is good enough to find a parameterization for quasi - developable meshes. Failures of IsoMap are not due to the coarse graph approximation of geodesic curves. Instead,holes or to the non-developable character of the mesh hinder IsoMap performance [15].

In addition to the attractiveness of discrete interrogations and constructions on a 3D mesh graph, there is the additional advantage of these operations being possible as Database operations. For that purpose, the 3D mesh has to be structured as a database. Although this is an apparently straightforward task, the correct database declaration (e.g. in Oracle™) of separate Topology and Geometry of a 3D mesh is not a trivial one, due to a historical bias in favour of (a) 2D entities (node, polyline) and (b)geographical TIN format for3D meshes. The TIN format (i) only serves meshes which represent surfaces $z=f(x,y)$, excluding all meshes that are the boundary $\partial \boldsymbol{B}$ of a solid body $\boldsymbol{B}$, and, (ii) assumes a connectivity (Topology) dictated by proximity of vertex projections on the XY plane and thus rejects the actual connectivity of the 3D mesh.

In response to these opportunities, this manuscript shows the conjunction between database resources (.e.g Oracle™-Spatial, -Graph and -Point Cloud) and visualization tools (e.g. JavaView ™[11]), to implement discrete counterparts of computational geometry floating point constructs. The emphasis is set on the usage of - as far as possible- freeware tools (e.g. JavaView™).

A. Terminology

1) Topology and Geometry in Computational Geometry.

In Computational Geometry, the term *geometries* refers to points $\boldsymbol{p: R^0 \rightarrow R^3}$, curves $\boldsymbol{C: R^1 \rightarrow R^3}$ and surfaces $\boldsymbol{S: R^2 \rightarrow R^3}$. The term *topologies* refer to the connectivity among the three basic ones VERTEX, EDGE, FACE, which are subsets of the geometries point, curve, surface, respectively. The connectivity among topologies is expressed on the terms BODY, LUMP, SHELL, FACE, LOOP, EDGE, VERTEX. The term *Edge* relates to an Oracle™ name, while the term *EDGE* relates to the classic solid modelling topology.

2) Topology and Geometry in Oracle™.

Oracle™ Spatial [3] calls *geometries* the basic types point and polyline. On the other hand, *Oracle™ topologies* may be either: (a) simple topological types (node, edge, face), (b) composite types built by using the simple topological types (node, edge, face) , and (c) attributes of geometrical data (e.g. 'park' applied to a 2D polygonal region or 'street' applied to a polyline).

3) Topology and Geometry in Graphs.

In graphs, the term *topology* refers to the connectivity among graph nodes. The term *geometry* is not directly expressed in graphs. A graph might have a geometric incarnation but geometric information such as location, size, orientation, etc. is not usually considered as inherent graph information.

In this manuscript, we will in general refer to *topology* and *geometry* in the classic sense of Computational Geometry. In specific locations in which the context is explicit (e.g. Oracle™), a clear warning will be issued for the reader.

4) Manifold Condition.

A set $\boldsymbol{M} \subset \boldsymbol{R^3}$is a 2-manifold [18] if for each point $\boldsymbol{p} \in \boldsymbol{M}$ there exists a $\delta \in \boldsymbol{R^+}$ such that for all radius r with $0 < r < \delta$, $\boldsymbol{B(p,r)} \cap \boldsymbol{M}$ is isomorphic to the disk $\boldsymbol{S^2}$. Informally, this definition denotes a closed or *watertight* shell, which presents no self-intersections. It indicates that all neighbourhoods of $\boldsymbol{M}$ are bijectively similar to, or deformable into, flat 0-thickness disks.

5) High Quality Triangular Mesh

This mesh contains quasi-equilateral triangles, whose minimal EDGE length δ satisfies $\delta \leq (\delta_{min} / 2)$, where δ_{min} is the smallest geometric detail of the original BODY boundary (*skin of a solid*) that its sample ***M*** is able to detect or preserve (Nyquist - Shannon Sampling Theorem [10],[16]).

This manuscript is organized as follows: section II examines the state of the art. Section III describes the methods that materialized our work, section IV describes the results obtained. Section V concludes the manuscript and discusses possible new research directions.

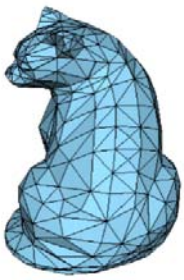
II. STATE OF THE ART

A. Oracle TM Triangulated Irregular Network (TIN) Libraries.

This section explores existing Database and Libraries or Oracle™ (i.e. PL/SQL) in order to construct a connection capacity between 3D geometry Viewers and Spatial databases. It must be noticed that commercial software may be used as viewer (e.g. MicroStation™) , but it is usually expensive and with other emphasis in the final goal. The freeware JavaView™ is used here as an alternative. The final goal is obviously to use the TIN representation to support 3D computational geometry constructs and related interrogation algorithms.

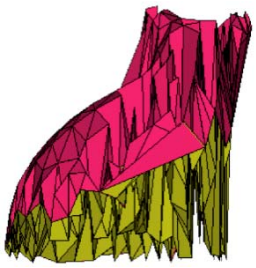
The Oracle™ - TIN library (SDO_TIN_PKG) was used to import the legal 2-manifold CAT model (Fig. 1(a)).A TIN data set is only able to import the geometry (i.e. vertex positions, Fig. 1(b)), and the result is an incorrect object representation (Fig. 1(c)), due to the fact that TIN does not accept importing connectivity information. Instead, TIN establishes an *(x,y)*-Delaunay vertex connectivity, derived from proximity of vertices in the XY plane (i.e. ignoring the z coordinate).In addition, TIN only handles surfaces $z=f(x, y)$ (i.e. having at most a unique z value for a given 2D location (x, y)). These limitations make TIN un-suitable for representing general triangular meshes. As the TIN representation collapses, any subsequent query or derived object algorithm does not proceed.

One concludes that, within a database environment, an alternative to the TIN formalism is required, which allow for the explicit declaration of vertex connectivity. A possibility explored later in this manuscript is the Point Cloud Oracle™ (SDO_PC_PKG) data type.



(a) Geometry (VERTEX set) and Topology (connectivity).

(b) Geometry Alone.



(c) Wrong Topology inferred from VERTEX closest neighbours according to distance projected onto the XY plane.

Fig. 1.Error in the Oracle™ TIN Topology reconstruction (i.e. 2.5D) based on vertex proximity inside the XY plane. *Data set Cat* (public from 3DCafe athttp://3dcafe.com/)

B. Oracle™ Database Topology and Geometry

In Oracle™, a Node has an associated(x, y) pair. An Edge supports 2 or more Nodes. Edges do not include circular arcs. A Face is built by one or more Edge loops. Each Edge may contain the information of incident Faces. There may be *island* entities (Edge loops, Edges, Nodes) inside a Face.

As in a Wing Edge data structure [8], the concepts of NEXT LEFT, PREVIOUS LEFT, NEXT RIGHT, PREVIOUS RIGHT exist in Oracle™. An indirect guarantee for manifold condition exists in the fact that there may be at most *one* NEXT or PREVIOUS Edge for a given Edge, both in the right and left directions (e.g. there can be no 2 NEXT LEFT Edges for a given Edge). This is a 2D world, with a primigenial Face created (with id = -1), and it is the universe on which all Nodes, Edges and Faces are contained. This topology has no geometry associated with it. A *loop Edge* is an Edge whose initial and final Node are the same. A loop Edge may have intermediate nodes. This means that a closed circuit may be represented by using one Edge.

In Oracle™, the name *Topology Geometry* is what Computer Aided Design and Manufacture call a *feature*. For example, a *vehicular roundpoint* or *traffic roundabout*[13] is a feature or *topology geometry*, made by a particular combination of Edges.

C. GIS Databases.

Reference [17] examines the extension to 3D of 2D geometry and topology primitives already present in GIS software and DBMS. An approach is proposed which amends the 2D polyline and XY-plane polygons to assume diverse *z* coordinates. Discussion also touches the usage of CGS modelling applied to GIS entities. This reference does not discuss graph operators for 2D or 3D GIS. Notice that the triangle - based Boundary Representation (a) is general and sound in geometric and topological terms, (b) allows for the most sophisticated scientific simulations (since it is based on 0-, 1-, 2- and 3-simplexes, base of Finite Element Analysis) and (c) has direct application in visualization, additive manufacturing, calculation of mass properties, etc. Triangular B-Reps are naturally representable by graphs, thus accepting tools present in Graph libraries of DBMS.

Reference [5] attempts the extension to 3D of 2D approximations stored of GIS objects, by using object - relational databases. This reference proposes a relational DBMS equivalent to a flat - polygon - based Boundary Representation. However, the 3D model is not implemented. An example query of region proximity / intersection based on bounding boxes is presented. The long-term goal is to implement a 3D model of GIS systems, but it is not discussed in this reference. The reference does not use Graph operators nor exploits the flexibility of 0-, 1-, 2-simplicial complexes for full 3D representation (i.e. quadrangular / triangular meshes).

Reference [19] explores editing and visualization of 3D data in GIS databases. The visualization by commercial packages (e.g. MicroStation^{TM}) is possible, but it is economically expensive and computationally cumbersome. This reference cites VRML as a possible way for visualization. However, VRML does not solve the problem of editing, as it is usually a model - to -image unidirectional solution.

The reference finds that3D primitives not supported by the DBMS. This limitation leads to exotic behaviour, such as impossibility to handle vertical Faces (a problem already encountered in the TIN standard).

D. Graph-based Geometry Computations.

Reference [6] is an example of continuous domain computational algorithms which are approximated by a reasonable graph discrete counterpart. In this particular case, a curve is required in a continuous domain, on which a line integral of a penalty function is minimized. In this particular case, the anisotropic penalty function depends on the position and tangent vector on the curve. The authors show that such a problem has an approximate solution if the function and curve domains are represented by a graph. The optimal curve is the shortest path on the synthesized graph.

Reference [20] presents a graph minimal path variant for the calculation of geodesic curves in an *m*-dimensional manifold $\boldsymbol{M}$ embedded in $\boldsymbol{R^n}$ with $n > m$. This graph - based distance measured on the *m*-manifold is used to assess an isometric map $\boldsymbol{M} \rightarrow \boldsymbol{R^m}$. In the case $n=3$ and $m=2$ this dimensional reduction algorithm constitutes in fact a mesh parameterization [15].

Reference [9} surveys data structures used in GIS. References [21] (Simplified Spatial Model - SSM) and [2] (Urban Data Model -UDM) appear as the earliest data structures that are topologically complete for expressing flat face 3D BODYs and SHELLs. The representation uses the concepts of BODY, SHELL, FACE, LOOP, EDGE and VERTEX (possibly with other names). No mention is made respect to the usage of DBMS to implement such structures. This survey expressly favours Object Oriented over Relational databases, and does not include a discussion on any concrete database. Reference [12] in the survey addresses the time stamps of the database to register the history of the sites and estates. However, no reference is made to actually using the database for the storage of 3D full topological / geometrical information.

E. Conclusions of Literature Review.

The reviewed literature shows that, possibly for historic reasons, Geographic Information Systems consumed spatial services from databases. Therefore, the spatial topology and geometry were oriented to serve 2D versions of features of a GIS system (roads, plots, parcels, rivers, lakes, etc.). Then, features and additional information (taxes, price, people) was hanged to these features, which is naturally stored in a DBMS. Then, a drive for 3D modelling was experienced for 3D spatial databases, but it collided against an already essential 2D structure. As an example, 3D topology cannot be stored in TIN format, since TIN topology is dictated by 2D proximity. This absence of full 3D topological and geometrical representation in DBMS domain is also visible in the absence of computational geometry algorithms actually mounted on DBMS - geometry based software. Our manuscript responds to such void, taping from the DBMS Graph libraries, with the expectation that more CAD, CAM or Reverse Engineering SW may actually be placed in DBMS.

According to the reviewed literature, we find that the following features of our manuscript are attractive for the reader: (a) replacement of the default TIN topology in OracleTM Spatial by the actual mesh topology. This replacement is conducted here by using additional relation declarations of Logical Network of the OracleTM Spatial database. (b) Implementation of the Boundary Representation Wing-Edge data structure [8] by using the spatial database. (c) Replacement of floating-point operations on 3D meshes (e.g. Geodesic curves) with discrete approximations (e.g. Shortest Path) of Graph Operations mounted on a spatial Database. (d) Additional

operations / constructs on the B-Rep Graph mounted on spatial databases (e.g. Minimal Spanning Tree). (e) Usage of JavaViewTM for end-user interaction as either (1) Master, which uses the services of a Database server or (2) Visualization Server, subordinated to a Master Database application.

III. METHODOLOGY

A. Implementation.

The present section reports the implementation and results of module articulation as per Fig. 2. A standard database (e.g. Oracle™) appears as foundation. On top of this database, an abstraction (Oracle™ Spatial) is built, in which the entities have geometric meaning. Likewise, the Graph abstraction to implemented (e.g. Oracle™ Graphs). Both, Oracle™ Spatial and Oracle™ Graphs are provided by Oracle™. On top of them, we built the abstraction of a Boundary Representation(B-Rep) which is a standard formalism for solid and 3D SHELLs description.

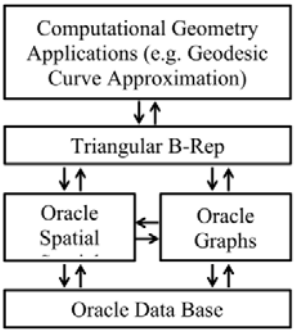


Fig. 2.Approximation of Computational Geometry Algorithms using Spatial Database - based Graph Libraries.

For the purposes of illustration, we have chosen to implement 2 different discrete approximations of Geodesic Curves on the 3D mesh. We do not implement a floating point PL approach (briefly commented in section *I*). Instead, we use the discrete approximation of Geodesic Curves made possible by the B-Rep on top of Oracle™ Spatial and Oracle™ Graphs.

B. Required Topology Constructors and Queries.

The SW component structure shown in Fig. 2 supports the following functionality by a combination of the database, its functions and its additional SW layers.

```
1. initial_vertex(EDGE):          → VERTEX
2. final_vertex(EDGE):            → VERTEX
3. counter_edge(EDGE):            → EDGE
4. owner_face(EDGE):              → FACE
5. owner_edge(VERTEX):            → EDGE
6. edge_sequence(FACE):           → [EDGE]
7. neighbor_face(EDGE,FACE):      → FACE
8. incident_edges(VERTEX):        → [EDGE]
```

The interrogation `owner_edge(VERTEX): → EDGE}` returns any EDGE for which the given VERTEX is the head. Consistently, the interrogation `incident_edges(VERTEX): → [EDGE]` returns a CW- or CCW- ordered sequence of EDGEs, incident to the given VERTEX.

C. B-Rep Database and Interface Architecture.

6) Construction of B-Rep Database and Graph.

Fig. 3 represents 3 different B-Rep graph construction processes, with mixed success results, implemented for the present manuscript. A pre-condition is, of course, the existence of a 3D 2-manifold triangular mesh ***M***. Although it is not a requisite for 2-manifold condition, the used mesh is closed or *watertight* and thus contains no border. A necessary step to declare the 3D mesh ***M*** inside Oracle™ is the importation of Geometry and Topology information.

As established in the Literature Review section, declaring a Topology (i.e. connectivity) in the TIN standard is not possible, ending up in an illegal situation. The TIN option connects a VERTEX $\boldsymbol{v}$ with the neighbour ones that are closest according to their separation with $\boldsymbol{v}$, projected on the XY plane. TIN assumes that $\boldsymbol{M}$ represents a surface $z = f(x, y)$, with f: $\boldsymbol{R^2 \rightarrow R}$ being a function. Therefore, TIN excludes, among others, meshes $\boldsymbol{M}$ which represent the boundary of a solid region. Typically, TIN allows only terrain - like digitisations. Fig. 1(a) shows the *Cat data set*. Fig. 1(b) shows the Geometry (point set) as imported and inserted in Oracle™. Fig. 1(c) displays the defective Topology that Oracle™ TIN infers from the Geometry.

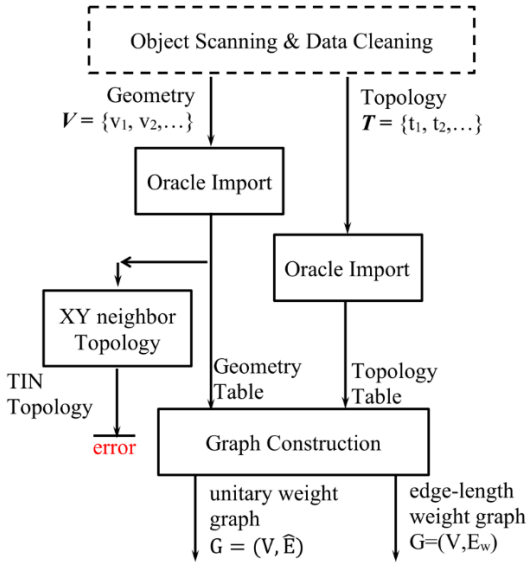


Fig. 3. Construction of Boundary Representation (B-Rep) Graph in a Database Manager. An error occurs using Oracle™ TIN since it replaces the actual Topology by closest (on plane XY) point connectivity.

Declaring an arbitrary Topology in Oracle™ implies to ignore the TIN standard and instead to declare the actual Topology via a Oracle™ Graph. The Geometry (point set) is still declared via Oracle™ Point Cloud. Our implementation compares two graphs: (a) with EDGEs having unitary weight, (b) with EDGEs whose weight is their Euclidean length.

7) Graphical User Interface. JavaView™

Fig. 4 displays an implementation of the Graphic User Interface, using JavaView™. The user interacts with, and receives feedback from, the JavaView™ client. This client passes the queries to a Computational Geometry application. In this report, the particular case of discrete approximation of geodesic curves on the mesh $\boldsymbol{M}$ is conducted. The application communicates the relevant interrogations to a translator Java/Database, which expresses the interrogations in terms of the Graph and Database operations. The database responds with the query / construct result, which is then translated into the data structures of the Computational Geometry Application and into Java commands, for human visualization.

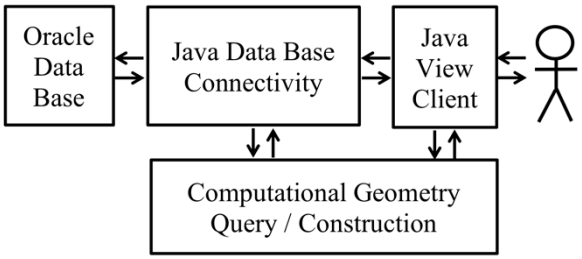


Fig. 4. Structure of a Computational Geometry Application Based on a Graph Database (e.g. Oracle™).

D. Background on Mesh Geodesics.

The purpose of this section is to give a short background on the Computational Geometry problem which serves as an example for our discrete approximation by using the Oracle™ Graph and JavaView™ capabilities. In this case, we consider the problem of finding a discrete approximation for the geodesic curve that joints two points of a surface $\boldsymbol{M}$. The geodesic curve remains on $\boldsymbol{M}$ and has the property of being *straight within* $\boldsymbol{M}$ and thus uses a minimal length for joining these two points on $\boldsymbol{M}$. For the purpose of completeness, we briefly discuss geodesic curves on a PL triangular mesh $\boldsymbol{M}$. For interesting discussions of geodesics, see [34].

PL Geodesic. A PL path α:[a,b] $\rightarrow M$ is a piecewise linear geodesic curve on M if and only if α'(t) is parallel to the vector normal to M, $n(\alpha(t))$, for each t in the interval (a,b). The geodesic curve, *on a surface,* joining two points of that surface is in general not unique (for both PL and smooth surfaces).

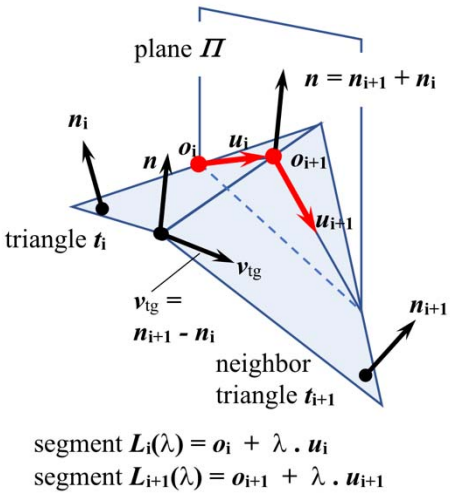


Fig. 5. Transition of geodesic curve between neighbouring triangles t_i and t_{i+1}[14].

Fig. 5 shows the transition of a PL geodesic curve $L_i(\lambda)$ on TRIANGLE t_i as it becomes curve $L_{i+1}(\lambda)$ on TRIANGLE t_{i+1}.Building a PL geodesic curve on M is based on the fact that the geodesic must remain in the plane Π spanned by the vectors n_i and n_{i+1} and containing the point o_{i+1}. The plane Π is normal to TRIANGLEs t_i and t_{i+1}.The vector locally *tangen*t to the geodesic would have the direction $v_{tg} = n_{i+1} - n_i$. The *acceleration* vector of the curve would be locally *normal* to the geodesic and to the surface and would have the direction $n_{i+1} + n_i$. All terms *tangent*, *normal*, *acceleration*, etc., are discrete approximations since the geodesic curve on M is piecewise linear (PL).

There exist particular cases, in which the geodesic hits a VERTEX or aligns itself with an EDGE. Notice that such cases do not appear if the geodesic is approximated by the shortest or lowest cost paths between the two vertices in question.

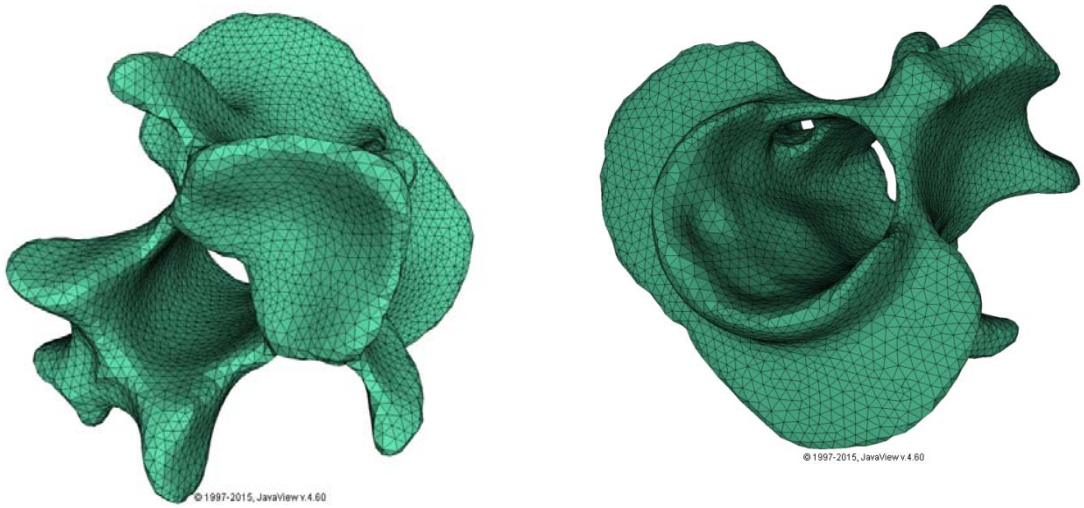
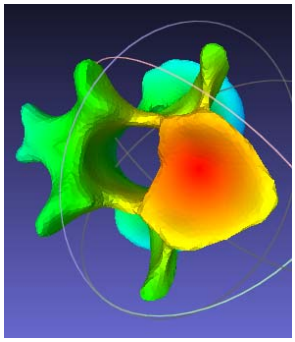
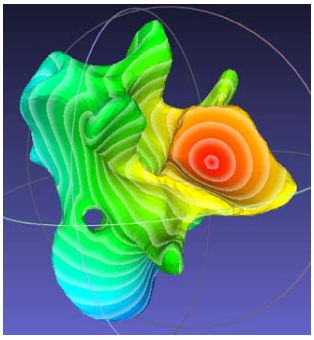


Fig. 6 *Vertebra data set* (Universidad EAFIT, Colombia), with homogeneous triangle EDGE length.

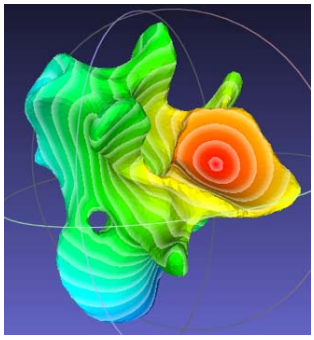
Fig. 6 shows the triangular mesh *Vertebra data set*, containing homogeneous EDGE length. Fig. 7 displays a scalar field $d_M()$ on the mesh M, which corresponds to the geodesic distance from a source or origin VERTEX v_0 on M to every other VERTEX of M. The image is obtained using the MeshLab [1] geodesics tool.



Smooth Geodesic Field.



Contours of Iso-Geodesice Distances. View 1.



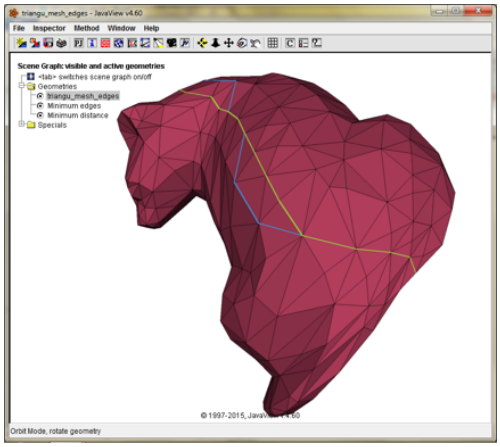
Contours of Iso-Geodesice Distances. View 2.

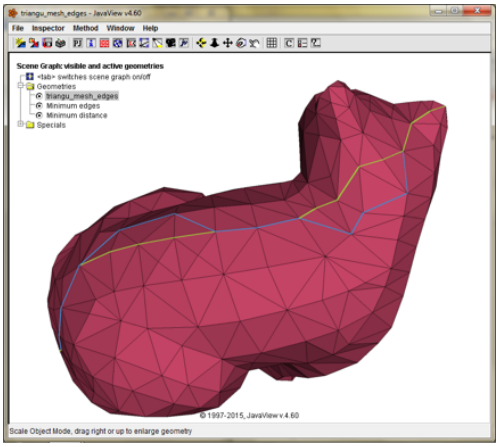
Fig. 7. *Vertebra data set* (Universidad EAFIT, Colombia). Scalar Field G:$\boldsymbol{M} \rightarrow \boldsymbol{R}$ expressing the geodesic distance, on mesh $\boldsymbol{M}$, from a given origin (red spot). Image obtained using MeshLab [1].

IV. Results

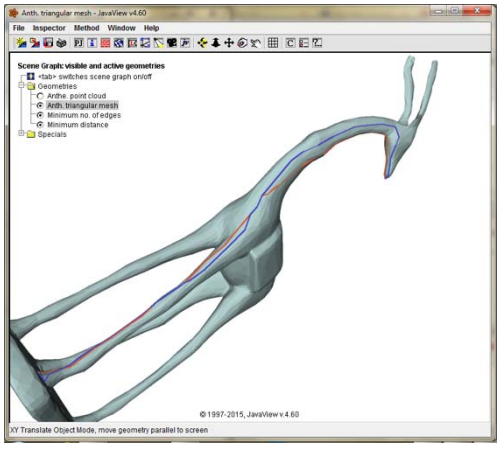
This section will discuss the following results: (a) With low quality meshes, 2 graph approximations of geodesic curves: with minimal number of EDGEs vs. with minimal path length. (b) With good quality meshes, 2 graph approximations of geodesic curves: with minimal number of EDGEs vs. with minimal path length. A contrast will be displayed which uses the MeshLab floating point geodesic scalar field.

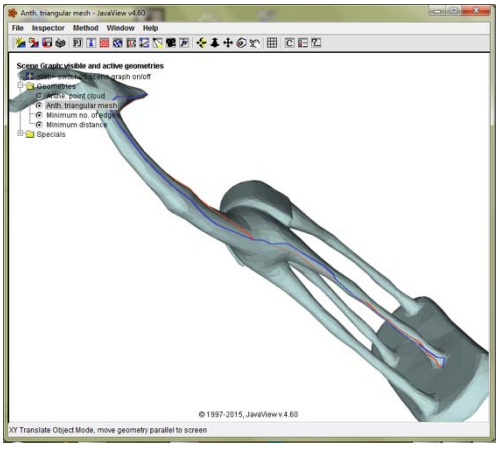
E. Results with Low Quality Triangular Mesh $\boldsymbol{M}$.





Blue: path with minimal number of EDGEs. Green: path with minimal distance.



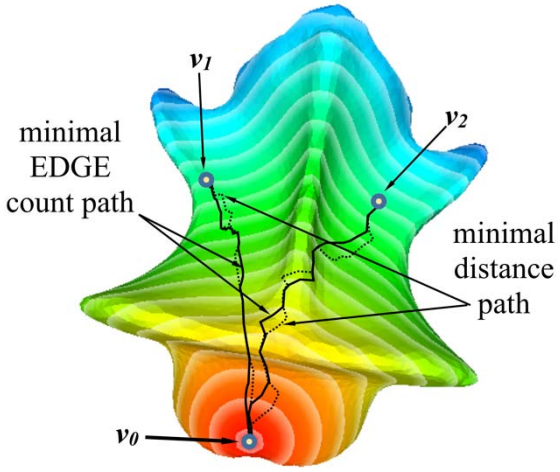


Blue: path with minimal number of EDGEs. Red: path with minimal distance.

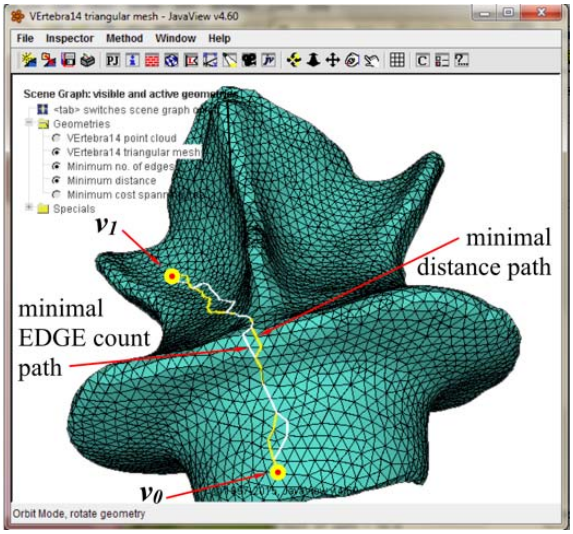
Fig. 8. *Cat* (3DCafe at http://3dcafe.com/) and *Antelope data set*s (Universidad EAFIT, Colombia) low quality mesh *data set*s. Graph geodesic approximations.

Fig. 8 shows two low quality *data set* meshes ***M*** (Cat and *Antelope*). For each *data set*, 2 graph - geodesic approximations are shown: (a) by a path with minimal EDGE count, (b) by a path with minimal distance. The two approximations render different paths. A reasonable expectation is that, as the triangulations have better quality, the two paths will converge to one, and they will converge to the PL curve obtained with the floating point geodesic calculation.

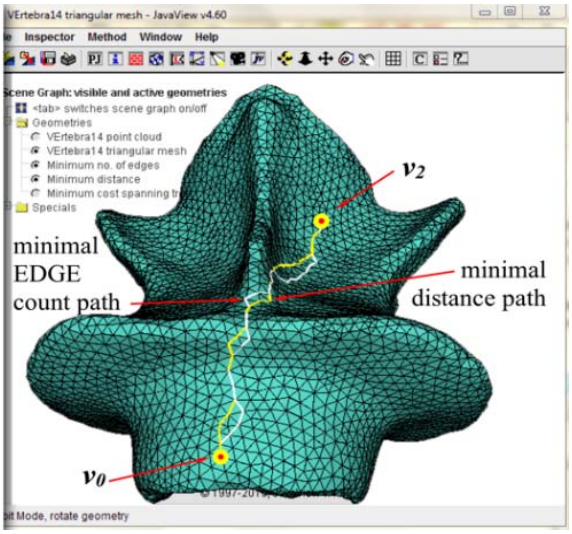
*F. Results with High Quality Triangular Mesh **M**.*



(a) [Geodesic Field from VERTEX v_0. Expected graph minimal paths from VERTEX v_0 to v_1 and v_2.



(b)Results. From v_0 to v_1 Minimal EDGE count vs Minimal length paths.



(c) Results. From v_0 to v_2Minimal EDGE count vs Minimal length paths.

Fig. 9. *Vertebra data set* (Universidad EAFIT, Colombia). Geodesic Curve Approximations. Vertices$v_1$andv_2 chosen so that $d_M(v_0,v_1)$= $d_M(v_0,v_2)$. Curve with Minimal Number of EDGEs vs. Curve with Minimal Cost.

Fig. 9 shows a higher quality *data set* mesh ***M***, called *Vertebra*. Fig. 9(a) shows illustrative paths from VERTEX v_0 to two VERTEXes v_1 and v_2, which are separated from v_0 by the same geodesic distance on mesh ***M***. The geodesic distance $d_M(v_0, v_1)$ on ***M*** between v_0 and v_1 may be estimated by the EDGE count of the path with least EDGEs between v_0 and v_1 on ***M*** (EDGE count $E_M(v_0, v_1)$). Eq. 2 states that $d_M(v_0, v_1)$ may be estimated by the length $L_M(v_0, v_1)$ of the shortest PL path, on ***M***, between v_0 and v_1. Equality is achieved when the triangles of ***M*** become infinitesimal.

$$E_M(v_0, v_1) \approx L_M(v_0, v_1) \tag{1}$$

$$\lim_{\delta(M)\to 0} E_M(v_0, v_1) = \lim_{\delta(M)\to 0} L_M(v_0, v_1) = \lim_{\delta(M)\to 0} d_M(v_0, v_1) \tag{2}$$

If a second vertex v_2 is chosen, such that its geodesic distance to v_0 equals the distance from v_0 to v_1 (Eq. 3), one has the situation of Fig.9(a). Notice that the colour texture in Fig.9(a) corresponds to the scalar field $f_{v\_0,M}(v) = d_M(v_0, v)$, which is the geodesic distance on ***M*** from v_0 to any variable VERTEX $v \in M$.

$$d_M(v_0, v_1) = d_M(v_0, v_2) \tag{3}$$

Under the condition of Eq. 3, for a finitely PL mesh $\boldsymbol{M}$, the minimal EDGE count $E_M(\ )$ paths to $\boldsymbol{v_1}$ and $\boldsymbol{v_2}$ from $\boldsymbol{v_0}$ represent similar distance (Eq. 5). The same similarity holds for the PL minimal distance $L_M(\ )$ to $\boldsymbol{v_1}$ and $\boldsymbol{v_2}$ from $\boldsymbol{v_0}$ (Eq. 5).

$$E_M(v_0, v_1) \approx E_M(v_0, v_2) \tag{4}$$

$$L_M(v_0, v_1) \approx L_M(v_0, v_2) \tag{5}$$

and so,

$$\begin{aligned} \lim_{\delta(M)\to 0} E_M(v_0, v_1) = \lim_{\delta(M)\to 0} L_M(v_0, v_1) = \lim_{\delta(M)\to 0} d_M(v_0, v_1) = \\ = \lim_{\delta(M)\to 0} d_M(v_0, v_2) = \lim_{\delta(M)\to 0} L_M(v_0, v_2) = \lim_{\delta(M)\to 0} E_M(v_0, v_2) \end{aligned} \tag{6}$$

Notice that the $E_M(\ )$ count of minimal path EDGEs does not represent a distance unless multiplied by the length of the EDGE (assumed constant in a High Quality Triangular Mesh). Table I displays the statistics of the two path types (minimal EDGE count vs. minimal length), for both paths $\boldsymbol{v_0}$-$\boldsymbol{v_1}$ and $\boldsymbol{v_0}$-$\boldsymbol{v_2}$.

TABLE I. Accounting on Paths $\boldsymbol{v_0}$ - $\boldsymbol{v_1}$and $\boldsymbol{v_0}$– $\boldsymbol{v_2}$of Fig. 9.

	Minimal EGDE Count (white)		**Minimal Distance (yellow)**	
path	**Num. EDGEs**	**length**	**Num. EDGEs**	Length
v0 - v1	34	107.66	34	102.75
v0 - v2	34	108.03	34	101.58

Homogeneous EDGE length meshes are not a rarity, since tasks such as mesh segmentation and parameterization are very difficult with other meshes. Meshes with non-homogeneous EDGE length are convenient for data reduction, but are very counter-productive for mesh segmentation and parameterization.

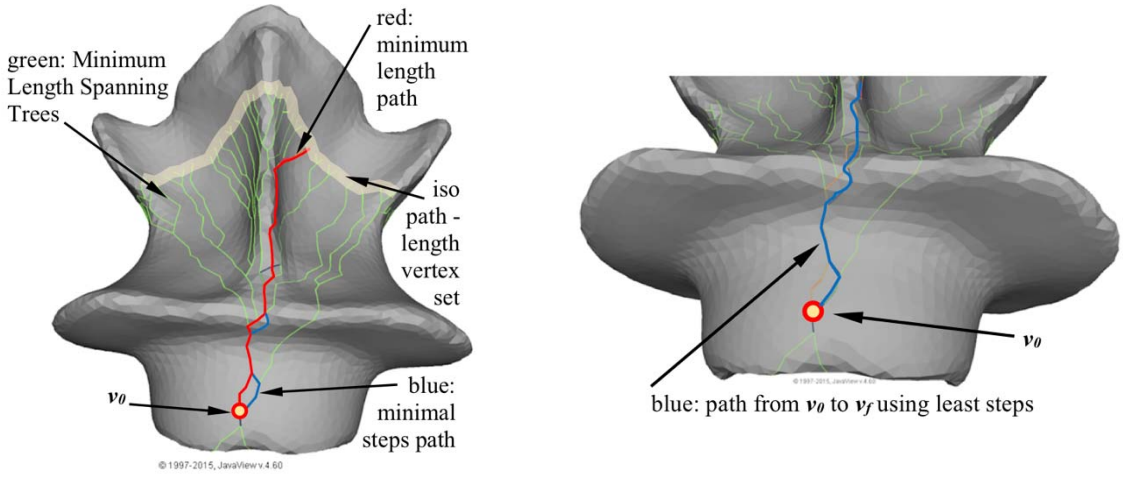


Fig. 10. *Vertebra data set* (Universidad EAFIT, Colombia). Multiple paths of similar length from an origin vertex. Geometry in Oracle™-Point Cloud. Topology and Algorithm in Oracle™-Graph. Visualization in JavaView™.

Fig. 10 displays several nearly iso-length paths (green colour) from an origin node in the *Vertebra data set*. As expected, the locus of the path end vertices approximates the geodesic iso-distance map of Fig. 9. Fig. 10 also shows iso-step paths, which are based on the number of edges traversed (assuming nearly constant edge length). As the triangulation improves in quality (small constant size EDGEs forming equilateral triangles), the iso-length and iso-step paths tend to be the same. The yellow strip represents a vertex set reached by paths which start from vertex $\boldsymbol{v_0}$ and reach vertices whose lengths are in a given interval. Notice the (expected) resemblance with the iso geodesic lengths of Fig. 9(a).

G. Minimal Cost Spanning Tree on $\boldsymbol{M}$.

Fig. 11 presents the Minimal Cost Spanning Tree for the *Vertebra data set*. This result is also achieved using the Oracle™ Point Cloud type (for the geometry data), supplemented with the Oracle™-Graph tables (for the topology or connectivity) and algorithm. The display is executed in JavaView™.

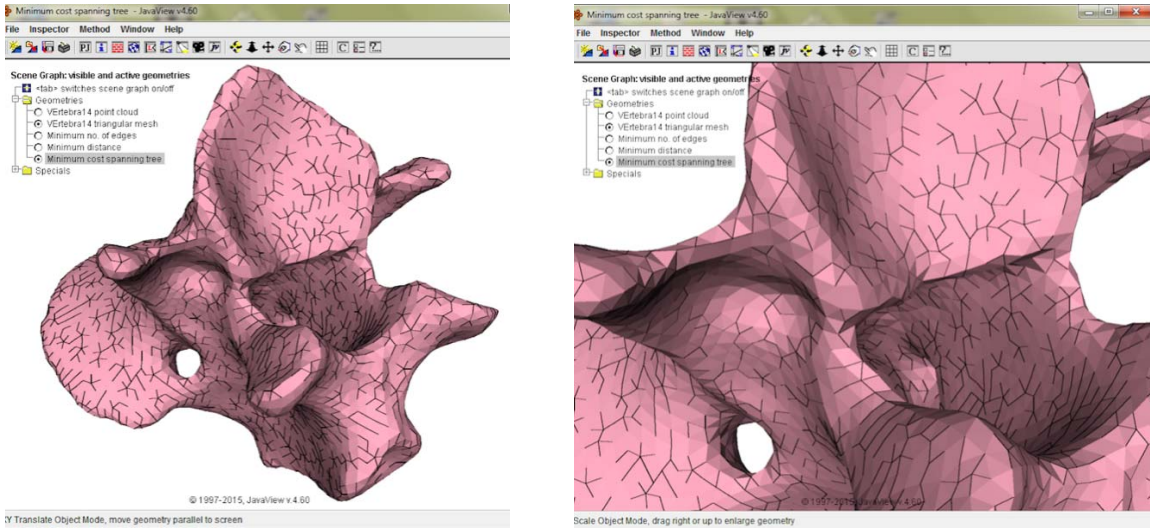


Fig. 11. *Vertebra data set* (Universidad EAFIT, Colombia). Minimal Cost Spanning Tree. Geometry in Oracle™-Point Cloud. Topology and Algorithm in Oracle™-Graph. Visualization in JavaView™.

V. CONCLUSIONS AND FUTURE WORK

This manuscript presented an articulation of database types and algorithmic, for the purpose of treating computational geometry and topology as other usual database information (client, payroll, assets, debt, address, etc.). This work has relevance given that spatial databases, mostly related to GIS, historically deal with 2D entities (parcel, road, block, highway, bay) and only ad-hoc incorporate 3D information.

The manuscript remarks the usefulness of database operators (e.g. graphs) given the fact that many interrogations and constructs in 3D Computational Geometry may be defined or approximated in terms of graph operations. A specific example is presented, of the geodesic field on a manifold triangular mesh $\boldsymbol{M}$, which can be approximated by shortest path interrogations in a high-quality mesh $\boldsymbol{M}$. High-quality meshes are compulsory for mesh segmentation and parameterization. Thus, the application pre-conditions for graph operations do not seem overly demanding.

This manuscript presents an added value of tool articulation, by including the application of JavaView™ to serve as interface with the DBMS and with the Graphic User Interface. The facts that JavaView™ is open software and database managers may be also found as open, indicate the possibility of cost-effective articulation of open tools for large data set computational geometry.

ACKNOWLEDGMENT

The authors wish to thank (a) the Laboratory of CAD CAM CAE of Universidad EAFIT for the *data set*s *Antelope* and *(Cow) Vertebra*, (b) the Dept. of Informatics of University of Vigo – Spain, which hosted part of this manuscript's work by author Stella Orozco – Ochoa and offered her the inspiring graph lectures by Professors Arno Formella and Marta Perez - Rodriguez while Stella Orozco - Ochoa was realizing her Magister degree at U. Vigo, (c) the Internet public site 3DCafe (http://3dcafe.com/) for the *Cat data set*.

REFERENCES

[1] P. Cignoni, M. Callieri, M. Corsini, M. Dellepiane, F. Ganovelli, and G. Ranzuglia,"MeshLab: an Open-Source Mesh Processing Tool," in Proc. Eurographics Italian Chapter Conference, 2008.
[2] V. Coors, "3D-GIS in networking environments," Computers, Environment and Urban Systems, vol. 27, no. 4, pp. 345-357, Jul. 2003.
[3] C. Murray. (2019)Oracle Spatial Developer's Guide 11g release 2 (11.2), Part Number E11830-06. [Online]. Available:https://docs.oracle.com/cd/E18283_01/appdev.112/e11830/toc.htm
[4] K. Crane, C. Weischedel, and M. Wardetzky. "Geodesics in heat," ACM Transactions on Graphics, vol. 32, no. 5, pp. 1-11, Sep. 2013.
[5] G. Groeger, M. Reuter, and L. Pluemer. "Representation of a 3-d city model in spatial object-relational databases," in 20th Congress of Intl. Soc. for Photogrammetry and Remote Sensing, 2004.
[6] J. Kim and J.P. Hespanha. "Discrete approximations to continuous shortest-path: application to minimum-risk path planning for groups of UAVs," in 42nd IEEE International Conference on Decision and Control, 2003.
[7] R. Kothuri and S. Ravada,"Oracle Spatial Geometries," Encyclopedia of GIS, Springer US, pp. 821-826, 2008.
[8] M. Mantyla, An Introduction to Solid Modeling, Rockville, Maryland, USA: Computer Science Press, 1988.
[9] T. Nguyen-Gia, M. Dao, and C. Mai-Van, "A comparative survey of 3D GIS models," in The National Foundation for Science and Technology Development (NAFOSTED) Conference on Information and Computer Science (NICS), 2017.
[10] H. Nyquist, "Certain topics in telegraph transmission theory," Bell System Technical Journal, 1928, reprint as classic paper in Proc. IEEE, vol. 90, no. 2, Feb. 2002.
[11] K. Polthier, S. Khadem-Al-Charieh, E. Preuss, and U.J. Reitebuch. (2019) The JavaView Project. [Online]. Available: http://www.javaview.de/
[12] N. Pelekis, B. Theodoulidis, I. Kopanakis, and Y. Theodoridis, "Literature review of spatio-temporal database models," The Knowledge Engineering Review, vol. 19, no. 3, pp. 235-274, 2004.
[13] D. Plater-Zyberk and Co. (2019) The lexicon of new urbanism. [Online]. Available: https://www.dpz.com/

[14] O. E. Ruiz and C. A. Cadavid. "PL-geodesics on PL-continuous partial meshes," in Proc. 9-th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision WSCG'01, 2001, vol. 1, pp. 110-119.
[15] O. E. Ruiz, D. Mejia, and C. A. Cadavid, "Triangular mesh parameterization with trimmed surfaces," International Journal on Interactive Design and Manufacturing (IJIDeM), pp. 1-14, 2015.
[16] C. Shannon, "Communication in presence of noise," IRE, vol. 37, pp. 10-21, 1949, reprint as classic paper in: Proc. IEEE, vol. 86, no. 2, Feb. 1998.
[17] J. Stoter and P. Van Oosterom, "Incorporating 3D Geo-Objects into a 2D Geo-DBMS," in ACSM-ASPRS 2002 Annual Conference Proceedings, 2002.
[18] M. D. Spivak, Calculus On Manifolds: A Modern Approach To Classical Theorems Of Advanced Calculus, Westview Press, 1971.
[19] J. Stoter and S. Zlatanova, "Visualisation and editing of 3D objects organised in a dbms," in Proc. of the EuroSDR Com V. Workshop on Visualisation and Rendering, 2003, pp. 22-24.
[20] J. Tenenbaum, V. De Silva, and J. Langford, "A global geometric framework for nonlinear dimensionality reduction," Science, vol. 290, no. 5500, pp. 2319-2323, 2000.
[21] S. Zlatanova,"Advances in 3D GIS,"Quarterly Review of Disegno Digitale e Design, vol. 1, no. 4, pp. 24-29, 2002.

AUTHOR PROFILE

Stella Orozco obtained in 2004 the Diploma on Computer Science at EAFIT University, Medellin, Colombia. Maria Stella Orozco conducted a Research Internship (2003) in Fraunhofer Institute of Computer Graphics, Darmstadt Germanyon "Detection of the line of sight" which obtained a Merit mention from U. EAFIT (Supervisors Dr. U. Kretschmer Fraunhofer IGD and Prof. Dr. O. Ruiz U. EAFIT). Stella Orozco earned the DEA in History of Science (Universidad de Santiago de Compostela, Spain2008). In 2014, Stella Orozco obtained the M. Sc. degree in Informatics (Merit Thesis on Data Mining in 2-Manifolds in R^3) from Universidade de Vigo, Spain (supervision of Profs. Drs. Arno Formella, U. Vigo and Oscar Ruiz, U. EAFIT). Her research interests are History of Science, Data Mining, Machine Learning and Spatial Databases.

Oscar Ruiz – Salguero obtained Diplomas in Mechanical Eng. and Computer Science at. U. de los Andes Colombia, and a M.Sc. - Ph.D. (1995) from U. Illinois at Urbana- Champaign, USA. Prof. Ruiz - Salguero has held Visiting Researcher positions at Ford Motor Co. (USA), Fraunhofer Inst. for Computer Graphics (Germany), University of Vigo (Spain), Max Planck Inst. for Informatik (Germany) and Purdue University (USA). He is coordinator of the CAD CAM CAE Laboratory at U. EAFIT (Colombia). His research interests are in Applied Computational Geometry.

Carlos Cadavid obtained a Diploma in Mathematics from Universidad Nacional de Colombia, a M.Sc. from the University of Cincinnati, and a Ph. D. in Mathematics from University of Texas at Austin (1998). Prof. Cadavid is currently a full time Faculty member at the Universidad EAFIT (Colombia). His areas of research are the topology of 4-dimensional manifolds and applications of topology to graphic computation.