6th International Conference on Applied Informatics
Eger, Hungary, January 27–31, 2004.

A Slicing Algorithm for Triangular Meshes

Márta Szilvási-Nagy[a], Ildikó Szabó[b]

[a]Department of Geometry, Budapest University of Technology and Economics
e-mail: szilvasi@math.bme.hu

[b]Department of Geometry, Budapest University of Technology and Economics
e-mail: szabo@math.bme.hu

Abstract

In this paper a slicing procedure for extracting geometric informations about triangular meshes is presented. In manufacturing sculptured surfaces are often represented by triangular meshes. Errors in the mesh as gaps, holes and invalid triangles lead in planar sections to defective contour lines used as tool path in milling or rapid prototyping. The slicing procedure works on a polyhedral data structure containing topological informations which are transferred to the polygonal plane sections. Then an effective two dimensional algorithm is detecting invalid triangles and deliver new three dimensional informations for repairing the mesh. Discrete curvature values at mesh points are also computed and analysed with respect to shape characterization of the mesh.

AMS Subject Classification: 68U07 [Computer-aided design], 65D18 [Computer graphics and computational geometry]

Key Words and Phrases: Triangular mesh slicing, discrete curvature, polyhedral data structure

1. Introduction

The numerical description of three dimensional objects is frequently generated by triangulating the surface of the object. The output of the triangulation is a mesh at the end of the design process, normally converted to a STL file format. In rapid prototyping, which generates the physical model layer-by-layer, the STL model is scliced with a series of parallel planes. Tool path construction in milling of a STL model may work also with plane sections, and requires closed contour lines. The slicing of a corrupted triangular mesh leads to shape errors and invalid layer contours. Methods for repairing corrupted polygonal loops are given in [8].

In a valid triangulation neighbouring faces have a common edge, each edge is shared by at most two faces, the faces meeting at a common vertex form a closed, non-overlapping cycle, and no triangle has an intersection with the interior of any other triangle. Numerical errors and errors in triangulation methods along sharp edges, in high curvature regions, at matching and trimming lines of surfaces, and not appropriate treshold values in the modeling process cause holes, gaps, overlapping triangles and too many and too small triangles in the mesh. A huge number of papers are dealing with mesh simplification, repairing and smoothing algorithms, e.g. [2, 3]. A short survey is given also in [9].

2. Computing a planar section of a STL model

An STL file consists of numerical data of a set of triangles. Each triangle is represented by coordinates of its normal vector oriented consequently outwards and by coordinates of its three vertices listed according to positive orientation of the triangle. In our notation the input data are

$$\begin{array}{lll} n_x^{(i)} & n_y^{(i)} & n_z^{(i)} \\ p_{1x}^{(i)} & p_{1y}^{(i)} & p_{1z}^{(i)} \\ p_{2x}^{(i)} & p_{2y}^{(i)} & p_{2z}^{(i)} \\ p_{3x}^{(i)} & p_{3y}^{(i)} & p_{3z}^{(i)}, \quad i = 1 \ldots, N. \end{array}$$

The number of triangles N varies from model to model, and depends also from treshold values in numerical calculations during the modeling process. Composite surfaces may be represented by several hundred thousands of triangles, what means four times as many real numbers. In order to get out topological informations from STL data we construct an edge based polyhedral data structure on the triangular mesh. The lists of vertices and faces are generated during reading the file. The list of vertices contains each point exactly once by identifying points within an appropriate distance δ. For an actual input point of the STL file the distance is computed from each point already stored in the vertex list. This procedure requires $O(N^2)$ time, but filters out null edges shorter than δ at the same time. At the end of reading the file the lists of vertices and faces are constructed with the following records exepted those which are denoted with $*$.

$$\begin{array}{lll} \mathbf{V} & : & \{V_i \longrightarrow (x, y, z)_i\}, \quad i = 1, \ldots N_v, \\ \mathbf{F} & : & \{F_j \longrightarrow (norm, V1, V2, V3, estart, att^*)_j\}, \quad j = 1, \ldots N_f, \\ & & V1, V2, V3 \in \{1..N_v\}, \quad estart \in \{1..N_e\}. \end{array}$$

The list of oriented half edges is also set up parallely, the number of which is three times the number of the faces, i.e. $N_e = 3N_f$.

$$\mathbf{E} \quad : \quad \{E_k \longrightarrow (Vs, Ve, face, nexte, linke^*)_k\}, \quad k = 1, \ldots N_e,$$
$$Vs, Ve \in \{1..N_v\}, \quad face \in \{1..N_f\}, \quad nexte \in \{1..N_e\}.$$

The physically different vertices of the mesh are stored by their coordinates in the vertex list, and are specified by their indices as pointers in the face and edge lists. Non degenerate triangles are stored in the face list. Each face is represented by the pointers to the normal vector, to the vertices, to a starting edge and an integer value which is a characteristic attribute established later. The edges in the edge list are determined by their starting and end points, the containing face, the next edge in the edge cycle of the face and the linked edge, i.e. the side of the neighbouring triangle which is the same segment with opposite orientation, if exists.

The missing data in the lists of faces and edges denoted by $*$ are determined by scanning all edges for identical end points. If a half edge has on a neighbouring face its other half, both are linked together by their indices which are written as pointers $linke \in \{1..N_e\}$ in the records of both edges. Boundary edges are linked to -1. The attribute att of the j^{th} face is the number of its boundary edges. Inner faces have no boundary edge, isolated triangles have three.

A simple shape characterization of the mesh is to specify so called feature edges according to the dihedral angle of their containing faces.

In our first example the feature edges are collected at dihedral angles which are approximately $\pi/2$ or very sharp. In Fig. 1 the feature and boundary edges of a STL model are shown. Invalid triangles occur along matching lines of the curved surface segments and at the round end on the left hand side.

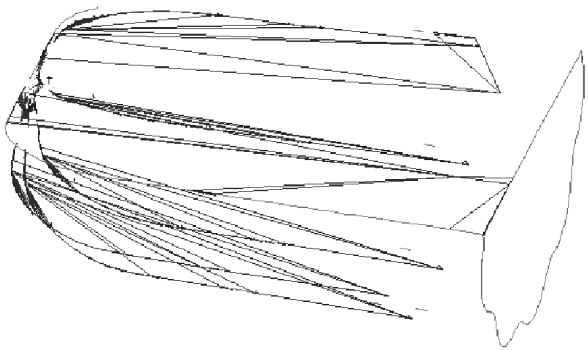


Figure 1: Feature and boundary edges of a STL model.

Computing the intersection of the mesh with a plane is carried out on the edge list. As inner edges are twice in the edge list, only the edges for which $k > linke_k$,

$k = 1, \ldots, N_e$ are intersected with the given plane. For a point of intersection M on the half edge E_k the following record is constructed.

$$M \longrightarrow (m, k, f1, f2),$$

where m is pointing to the coordinates in the point list $\{M_l\}$, $l = 1 \ldots, N_m$ computed in the plane, k is the pointer to the intersected edge, $f1$ and $f2$ are the pointers to the faces containing this edge, where $f1 = E_k \rightarrow face$ and $f2 = E_k \rightarrow linke \rightarrow face$ (Fig 2). In the case of boundary edges one of $f1$ and $f2$ equals -1. Of course, M is lying also on the edge E_{linke_k} if E_k is an inner edge, and this linked edge leads to the same data. The sides of the line of intersection are straight line segments joining two points from the list $\{M_l\}$ which are lying in a common face. This step is made by a sorting procedure according to $f1$ and $f2$. The next step is the collection of chains of sides using the face pointers again. A valid polygonal intersection consists of a closed outer loop which may contain disjunct inner loops.

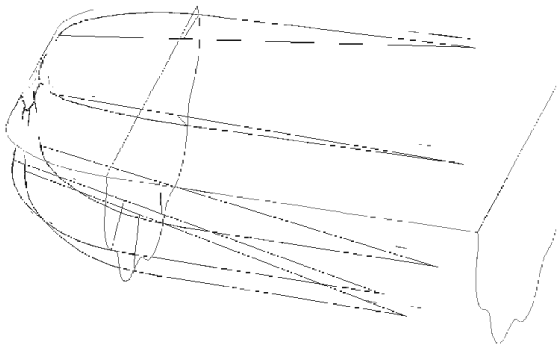


Figure 2: Plane section of the mesh shown in Fig. 1.

In Fig 2 a planar section of the STL model in Fig 1 is shown. The invalid segments are due to errors in the mesh.

If the outer loop is not closed, then boundary edges have been intersected, and there is a gap in the mesh. The faces bordering the gap are marked for matching in the mesh repairing process. Single segments and short open chains are due to invalid triangles which are marked for deletion from the mesh. In this way new three dimensional informations are obtained from scanning two dimensional data. Flat and high curvature regions are also detected. Moreover, computing the first order difference of discrete curvature values at vertices of the polygonal line as in [5] gives information about regions of the mesh for smoothing processes. Tiresome and time consuming 3D algorithms can be avoided on the base of such informations.

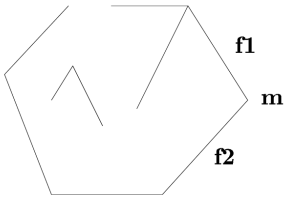


Figure 3: Geometric informations and errors in a plane section.

3. Mesh investigation by computing discrete curvature values and discrete geodesic lines

Discrete curvature values are frequently used in mesh smoothing processes. If the triangular mesh is a good approximation of a sculptured surface model, then discrete Gaussian and mean curvatures approximate well those of the surface, and discrete geodesic lines show the shape faithfully. The theory of discrete curvatures and shortest geodesic lines was developed for polyhedral surfaces by the school of Aleksandrov [1]. An extension of this theory with discrete straightest geodesics is presented in [7].

The straightest geodesic on a polyhedral surface is the path of a moving point, for each point of which the sum of the angles on the left hand and right hand sides measured on the faces are equal (Fig 4). In interior points of a face this geodesic is locally a straight line, across an edge two angles sum up on the opposite sides, at a vertex it halves the total vertex angle. Straightest geodesics provide unique solution of the initial value problem, and can be extended uniquely on a polyhedral surface.

A straightest geodesic starting from a point in an arbitrary direction on a sphere is a big circle with the radius of the sphere. We have investigated two different triangulations of a sphere, and generated straightest geodesics in different initial directions. Due to numerical inaccuracy, the solutions are no exact circles, but on the mesh of nearly uniform triangles (Fig 5(a)) the deviation from a circle is smaller than in the triangulation made by longitudes and parallel circles (Fig 5(b)). Moreover, geodesics are rocking more while changing the initial direction on the second sphere.

In the computation of discrete curvature values we have followed the principles in [6]. The mean curvature value κ_H at a point $\mathbf{x}_i$ of an arbitrary mesh is determined by the formula

$$\kappa_H(\mathbf{x}_i) = \frac{1}{2} \left| \mathbf{K}(\mathbf{x}_i) \right| .$$

Here $\mathbf{K}(\mathbf{x}_i)$ is the curvature normal

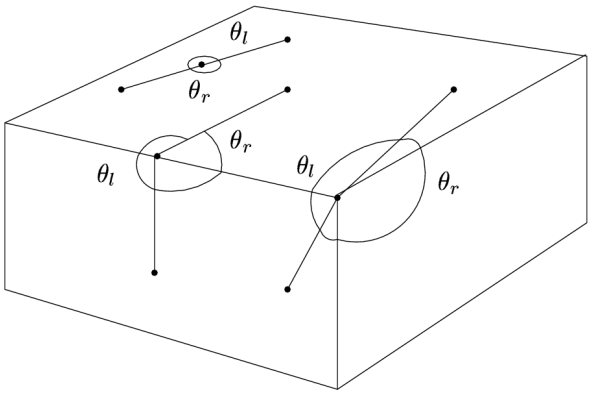


Figure 4: Straightest geodesic on polyhedral surface.

$$\mathbf{K}(\mathbf{x}_i) = \frac{1}{2A_{mixed}} \sum_{j \in N_1(i)} (\cot \alpha_{ij} + \cot \beta_{ij})(\mathbf{x}_i - \mathbf{x}_j),$$

where the mixed area A_{mixed} is computed from the Voronoi area of the vertex $\mathbf{x}_i$.

The discrete Gaussian curvature κ_G is given by

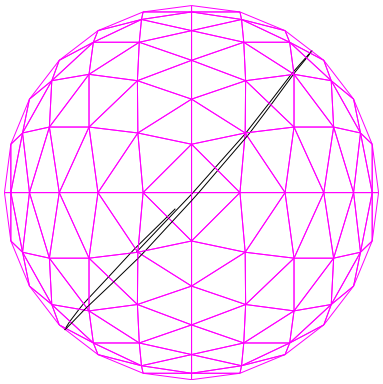
$$\kappa_G(\mathbf{x}_i) = \left(2\pi - \sum_{j=1}^{\#f} \theta_j \right) / A_{mixed},$$

where θ_j is the angle of the j^{th} face at the vertex $\mathbf{x}_i$, and $\#f$ denotes the number of faces around this vertex.(Fig 6(a), 6(b))

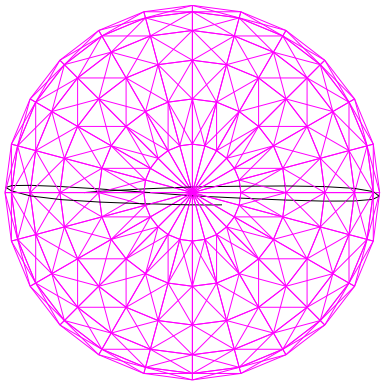
The two different triangulations of the sphere result different curvature values. In the more uniform mesh (Fig 5(a)) the curvatures at the different mesh points are the same, but in the second case (Fig 5(b)) their values are changing from point to point.

The example of a cylinder shows the influence of triangulation on the discrete curvature values more explicitely. Along the boundary circle of the upper face the Gaussian curvature should be the same at each point. As the area values of the triangles on the upper face are very different, so are also the computed discrete Gaussian curvatures. Smaller triangles cause higher curvatures. In Fig 7 vertices with smaller curvature values are marked with squares, bigger values are marked with stars.

Estimating surface normals have a key role in many problems, e.g. smoothing, denoising, visualizing. A simple estimation is to take the everage of the normals of the surrounding triangles at each point. A weighted average uses the incident

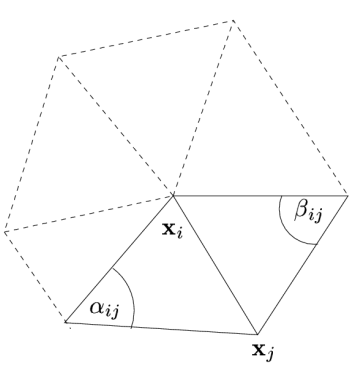


(a) Straightest geodesic on a nearly uniform triangulation.

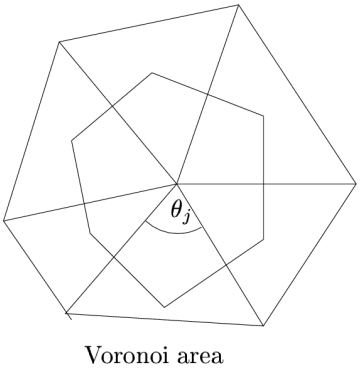


(b) Error in a straightest geodesic.

Figure 5:



(a) Data of discrete curvatures.



(b) Voronoi neighbourhood of a vertex.

Figure 6:

angle of each face at a vertex as the weights, in an other method the weigths are the area values of triangles around the vertex. We have computed the estimated normal vectors by all three methods at the vertices of the cylinder, and have drawn the results in Fig 8. The three segments at each point demonstrate the difference of the three methods.

Unfortunately, STL files generated by CAD systems are computed with different and unknown algorithms as outputs of a black-box system. Our examples show that such triangular meshes are in general bad approximations in the above mentioned sense. However, shapes of STL models show the shapes of approximated surfaces

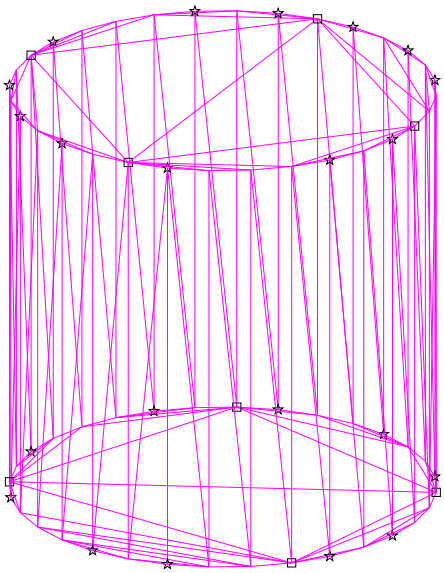


Figure 7: The curvature values are different at the marked vertices.

clearly. The most usable informations about triangulated surfaces can be obtained by feature lines composed from feature edges.

Results and techniques from differential geometry can be transferred to high-quality meshes. A construction of fair triangle meshes is presented in [4] made by a subdivision scheme that is optimal with respect to some discretized curvature energy functional.

4. Conclusions

In the presented method for computing plane sections of triangular meshes we have first constructed a polyhedral data structure on the mesh in order to detect gaps and holes due to numerical and algorithmic errors in the triangulation. Then we have intersected the mesh with a series of planes. The representation of a planar section consisting of polygonal lines and loops contains three dimensional informations, and provides new informations about the mesh which are usable for detecting invalid triangles and defected regions. We have analized discrete geodesic lines and curvature values on STL models of spheres and cylinders. Finally, we have concluded that these values characterize in fact the polyhedral surface of the mesh, and are not usable for the characterization of the approximated surface.

The slicing algorithm is implemented in Java language, the computations of discrete curvatures have been made by a symbolical program package.

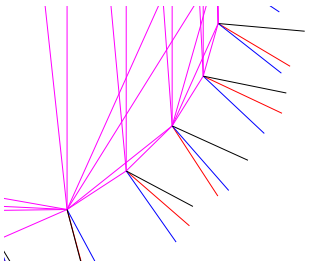


Figure 8: Different estimated surface normals at the vertices of the cylinder shown in Fig 7.

References

[1] Aleksandrov, A.D. and Zalgaller, V.A.: *Intrinsic Geometry of Surfaces*, Vol. 15 of Translation of Mathematical Monograph, AMS Rhode Island, USA, 1967.

[2] Hamann, B.: A data reduction scheme for triangulated surfaces, *Computer Aided Geometric Design* 11 (1994) 197-214.

[3] Hoppe, H.: Efficient implementation of progressive meshes, *Comput. & Graphics* 22 (1998) 27-36.

[4] Kobbelt, L.P.: Discrete fairing and variational subdivision for freeform surface design, *The Visual Computer* 16 (2000) 142-158.

[5] Lin, G.H., Wong, Y.S., Zhang, Y.F. and Loh, H.T.: Adaptive fairing of digitized point data with discrete curvature, *Computer-aided Design* 34 (2002) 309-320.

[6] Meyer, M., Desbrun, M., Schröder, P. and Barr, A.H.: *Discrete Differential-Geometry Operators for Triangulated 2-Manifolds*, in Visualization and Mathematics, Hege, H-Ch. and Polthier, K. (Eds.), Springer 2003, 35-57.

[7] Polthier, K. and Schmies, M.: *Straightest geodesics on polyhedral surfaces*, in Visualization and Mathematics, Hege, H-Ch. and Polthier, K. (Eds.), Springer 1997, 135-150.

[8] Park, Sang C.: Sculptured surface machining using triangular mesh slicing, *Computer-aided Design* 36 (2004) 279-288.

[9] Szilvási-Nagy, M. and Mátyási, Gy.: Analysis of STL files, *Mathematical and Computer Modelling* 38 (2003) 945-960.

Postal addresses

Department of Geometry
Budapest University of Technology and Economics
H-1521 Budapest, Egry József u. 1. H. 22.
Hungary