Mesh Optimization for Animation Purposes

Oliver Bunsen[*], Georg Fleischmann[*]
Academy of Media Arts, Cologne

Abstract

Three-dimensional scanning devices have proved to be invaluable for obtaining surface data of complex three-dimensional figures and models. In order to allow real-time animation of the scanned models and objects, significant data reduction is required. Several algorithms have been presented in literature to convert the complex 3D data to simpler polygonal approximations of the surface. All of these methods perform a vast reduction of data while keeping the actual surface information within certain error bounds. Unfortunately, none of these algorithms is offering any means of user interaction to incorporate animation requirements into the decimation process. This paper presents a first approach to incorporate the requirements of non-rigid object animation into surface meshing based on data from laser range scanning systems.

1 Introduction

The animation of complex objects which are modeled by polygonal surface meshes can be seen as geometric transformation of groups of mesh nodes. A common approach for modeling objects of this type is to create a triangle mesh based on scanned 3D-data obtained from a solid model of the object. Suitable triangle meshes have to meet two major requirements. The first goal is to come up with a storage efficient surface representation that approximates the laserscan-data-set within a predefined error range. The second goal is to provide properties related to the subsequent animation processes. These properties include triangle shape, mesh node position, mesh node density and required edges between certain nodes. All of these structural properties determine the range of feasible animation of the object's surface in advance. It is intuitively clear that an inappropriate mesh will disallow certain deformations of the surface.

The field of polygon decimation of large models obtained from 3D-scanning devices has been intensively studied in literature. These methods generate efficient approximations of the original data from laser range scanning systems that fulfill the requirements of rigid object computer animation. The requirements of non-rigid animation that is based on temporal surface deformation are not taken into account. The main contributions in the field of polygon reduction have come from [1], [2], [3], [4], [5].

[*] Dept. of Computer Science / AV-Media, Peter-Welter-Platz 2, 50676 Cologne, Germany, e-mail: {obunsen, fleischmann}@khm.de

An animation oriented meshing approach for the special field of virtual actor animation has been presented in [6]. This work uses a generic face mesh which is mapped on human head scan data. Human faces all share approximately the same location of mouth, nose, eyes and main face muscles. Since cartoon figures have very exaggerated individual face proportions, this generic approach produces poor results. Especially for the subtle field of modeling convincing 3D characters, it remains a challenging task to come up with an animation based remeshing algorithm working on data from laser range scanning systems.

The first animation meshes for cartoon-like virtual actors have been manually created using "trial and error". Our software tools merely provided an interface for placing individual nodes on the actor's scanned surface. The operator was able to link any two nodes with an edge. Several attempts had to be carried out with this time consuming method until a suitable surface mesh contained the potential to exhibit the desired deformations with a minimal total number of mesh nodes.

After several failed attempts to generate optimal meshes, the animator generally acquires an intuitive knowledge about how to create an optimal mesh. A formalization of this knowledge is the first step towards an automatic and therefore time saving approach of the generation of meshes for animation.

This paper contains a formalization of the structural properties of a mesh. A software concept for computer supported animation-mesh design is developed based on this formalization and results of its implementation will be discussed.

2 Requirements of Animation to Mesh Generation

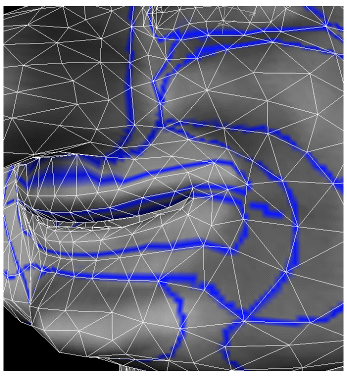
Figure 1: Expression "Grin" deforms the surface mesh with implicit animation lines.

An optimal mesh respects a number of implicit lines that are approximated by nodes and their connecting edges. Two types of lines appear in a mesh. The first type describes the defining and most strongly transformed surface points of the animation. The transformation of neighboring mesh nodes appears to depend on the animation line nodes and its strength declines with their distance. Other animation lines describe skin-like folds that result from nearby deformations (Figure 1).

We will call these lines *animation lines* and use them as the key concept for the formal description of the optimal mesh structure. The animation lines are intended to form a simple and intuitive formal description of the animation requirements to the mesh. With this concept we will now postulate the optimal properties of an animation mesh:

1. The mesh elements are triangles. Triangles offer the highest degree of flexibility to approximate a curved surface and to adapt on geometric constraints. Triangles are the actually supported type of polygons for accelerated graphics hardware that allows powerful realtime rendering of objects.

2. The triangles should be as equiangular as possible, so that the mesh offers a high degree of possible deformation before any two edges intersect.
3. The triangle mesh approximates the surface described by scan data with a limited number of nodes. Limited because of performance considerations.
4. The animation lines induce a partition of the surface into regions-of-interest. The approximation error of the surface in a region depends on its importance (e.g. mouth is more important than rear skull).
5. The surface of some of these regions may either be heavily enlarged or receive very detailed deformation during animation. Additional nodes have to be placed into these regions.
6. The mesh will have to contain holes (mouth, eyes) and interfaces so that it can be connected with other partial meshes.
7. A system of animation lines will have to be approximated by sequences of edges (Figure 3).
8. Whenever possible, triangles have to be placed in successive layers along animation lines. This mesh structure allows smooth surface deformation for skin-like elastic surface behavior.

The relative importance of the single requirements depends on the application and may change in special cases.

3 Preparation of Animation-Oriented Meshing

A meshing algorithm for a non-interactive generation of a figure’s animation mesh would have to take as input the figure’s shape information and an abstract description of the figure’s animated expressions. In practice, it is impossible to abstractely define the figure’s entire animation and subsequently create an optimized animation mesh. Subtle deformations of a figure’s surface can only be expressed based on well defined geometric transformations of individual mesh nodes along the time axis. An animation optimized mesh already has to be available before it can come to the description of subtle surface animation.

The operator has to span this information gap at the mesh creation stage. Our observation that the properties of a mesh that is optimized for animation can be described by means of animation lines is leading to an interactive approach for mesh creation. The concept of animation lines and induced regions-of-interest is quite intuitive. The operator sketches the lines on the figure's scanned surface. The animation lines also serve to define the *regions*. Several lines intersect and enclose regions. The regions are automatically identified. The operator assigns relative node density values to each of the regions. At this stage, all necessary external information for the algorithmic creation of a first mesh is provided. An algorithm performs the meshing of each region individually. The operator can alter some mesh parameters and redo a region's mesh. Interactively, an experienced operator is quickly able to create an optimum mesh. The following paragraphs describe the approach in more detail.

3.1 The Domain

The meshing domain consists of the 3D laser scan data. Due to the scan geometry the data is organized on a 512 times 512 unfold surface grid of a cylinder. In other words, the data set is a discrete 2D-function that maps two cylindrical coordinates - scanning angle and the vertical laser position - to a floating point depth value. The data points can be characterized as 2.5D-pixels. Due to the nature of the scanning data, 2D meshing algorithms may produce working surface meshes.

3.2 Processing Animation Lines

The user interface for sketching the animation lines displays the scanned figure's texture. This allows the operator to navigate on the surface. A 3D-view of the object is offering additional control of the line's positions. The lines are curves in 3D on the object's surface.

The decomposition of the domain into regions that are defined by intersecting animation lines can be performed algorithmically. The lines are split into line *segments* that connect either two intersection points, one intersection point and an animation line's endpoint or two endpoints. Several line-segments may have common endpoints. A region's data structure contains references to a cyclic sequence of line segments and possible internal segments (Figures 2,3).

For a single connected system of line segments, the region-analysis is straightforward. The detection of additional internal line segments inside a region may require more advanced topological analysis or user interaction.

Throughout the process, the data structures will have to be carefully designed in order to clearly manage segments shared by two regions and segment's endpoints that are shared by arbitrarily many regions.

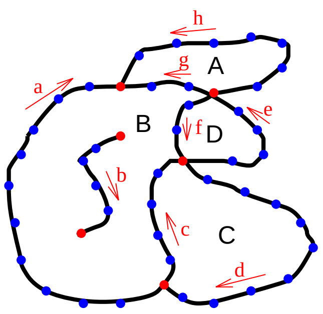

Figure 2: System of initialized segments a,..,h forming the regions A,...,D.

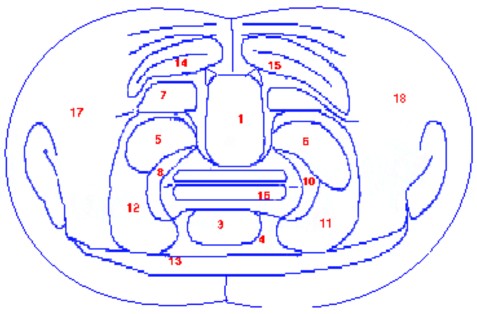

Figure 3: Regions in a figure's face. Some regions have internal segments.

3.3 Surface Meshing - Boundary Initialization

The basic step of meshing a region is to initialize its boundary with nodes and connecting edges. For the meshing, each line segment is replaced by a sequence of edges fulfilling the following requirements:

1. The segments' endpoints are the endnodes of the sequence of edges. Shared endpoints of neighbored segments cause shared endnodes of their sequences of edges.
2. The sequence of edges approximates the segment-curve sufficiently well (adaptive boundary initialization).
3. The edge length on a segment will have to conform to the typical edge length parameters for the future meshes in both neighbored regions. The edge length for internal segments will have to conform to the future surounding mesh. Otherwise, sudden strong changes in edge length will lead to malformed triangles in the animation mesh.

The simple approach for boundary initialization is to place the nodes equidistantly with respect to a discrete arc length measure in the 512x512 domain. The total arc length of the segment is calculated as a sum of the cartesian distances between subsequent curve pixels in the discrete 2.5D-domain. The total arc length is divided by the intended distance between subsequent nodes. In order to include the segment's endpoints, the intended distance is reduced to the next value that allows equidistant edges over the segment-curve.

The goal of adaptive meshing (node density is a function of the local level of detail) includes adaptive boundary initialization. A better approximation of the boundary segments can be achieved by increasing the node density in presence of high curvature [9].

Definition 1: **Curvature of a Line in 3D.**

Let $f: \mathrm{I} = [a, b] \rightarrow \mathbb{R}^3$ be parameterized to arc length $s \in \mathrm{I}$ and the second derivative exists and $s_1 \in \mathrm{I}$, then $k(s_1) = \left| f''(s_1) \right|$ is the curvature of f at s_1.

Coming up with a curvature based method of node placement on a segment-curve is less straightforward than one might expect. The optimization problem is very constrained. Based on curvature information, the position of a relatively small number of nodes has to be altered, while keeping the edge length within bounds. Any attempt of placing nodes on the curve's extreme points will violate the need of conforming edge lengths. Coming up with a modified density function to place the nodes is inadequate for short segments with less than 10 nodes. An optimal solution is still open.

4 Surface Meshing with an Advancing Front

4.1 Basic Algorithm

The meshing of the regions is performed by an *advancing front* algorithm [7] that is known to produce meshes with successive layers of triangles. The initialized region's boundary forms the initial front. In every iteration a new triangle is constructed by adding two new edges on some front edge. An update operation for the front replaces the front edge with the newly created triangle edges. The unmeshed region has shrunk and may break down into multiple shrinking regions that are defined by independent *components* of the front. Finally, the unmeshed region will disappear like a melting ice block. The meshed region remains.

The front C is defined by a set of connected, directed, simple, planar and subsequent edges. Multiple edges may share common nodes. The successor and predecessor of an edge is uniquely determined (Figure 4):

Definition 2: **Unique Successor of an Edge.**

Given three edges E, F, G with a common node E.dest = G.orig = F.orig,
succ(E) = F is uniquely determined iff

$$angle(E.orig, F.orig, F.dest) = \max_{K \in \{F,G,H\}} angle(E.dest, K.orig, K.dest).$$

The reachability of an edge by a sequence of successive edges is an equivalence relation. The corresponding equivalence class is defining the components of a front.

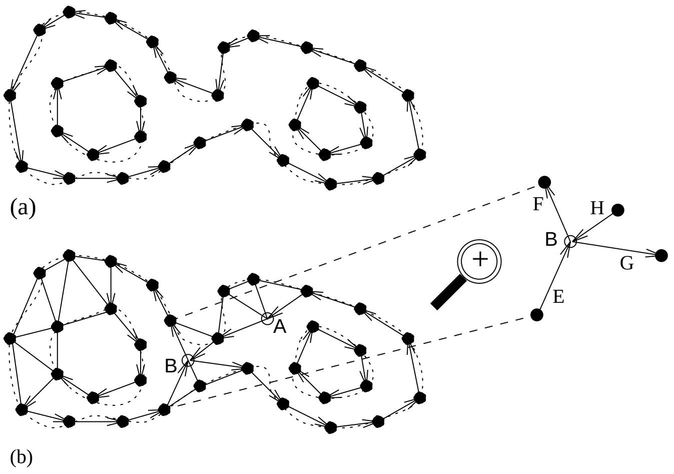

Figure 4: (a) Multiply connected region with initialized front made up from three components. (b) 12 iterations have taken place. Two nodes A,B have been added to the domain. Two components form two subregions. F is the successing edge of E.

By convention, the unmeshed region can be found left of an edge. This allows to maintain one formalization of the unmeshed region, even when it breaks down to multiple shrinking regions in the meshing process. At all stages the unmeshed region is the union of all interiors *interior* C(E) of all components C(E) of the front C (Figure 4).

In an iteration step of the advancing front algorithm a new edge is selected at random. Two candidate nodes are considered for the creation of a new triangle. The *new candidate* node is introduced to form a new triangle of ideal equiangular shape. The *existing candidate* node is chosen from the set of existing front nodes to offer an alternative triangle that knits well with the existing mesh.

An efficient strategy for determining a suitable existing node and for prefering one of both candidate triangles in order to globally optimize mesh quality is based on the *constrained Delaunay triangulation* (CDT) [10] of the unmeshed region. The constrained Delaunay triangulation of the current unmeshed region respects the boundary and fills the region with triangles that respect the *modified empty circumcircle criteria* (equivalent for constrained Delaunay triangulation).

Definition 3: **Modified Circumcircle Criteria.**

The triangles' circumcircles do not contain any other boundary nodes than the vertices of the triangles. If the circumcircle contains another node, then the triangle is separated from that node by a required boundary edge (constraint).

In an iteration step, the CDT-triangle node opposite to the selected edge is chosen to be the existing candidate. The empty circumcircle criteria globally maximizes the minimum angles in a triangulation. This way, the best choice is made from the set of existing nodes that still forms a well shaped triangle. The new candidate node R that can produce an ideally equiangular triangle is only preferred if this new triangle would still exist in the CDT of the unmeshed region after adding R. This is an efficient (Figure 5) strategy for globally optimizing mesh quality. Ideal triangles will only be produced if they do not cause malformed triangles in later iteration steps.

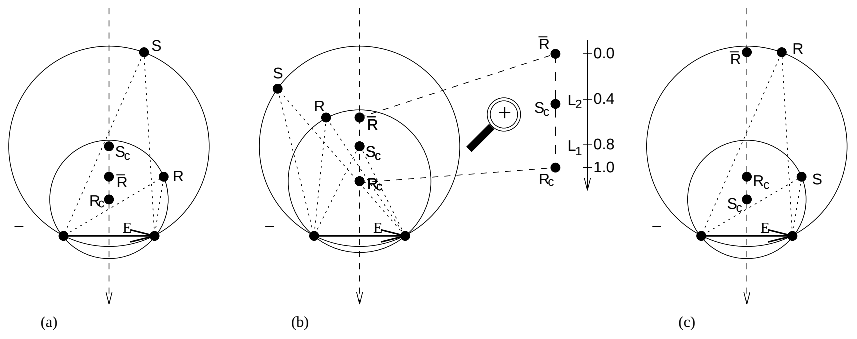

Figure 5: FARESTAM [7] proposes the following Delaunay based selection strategy with an additional candidate node. Determine $\overline{R}, \overline{S}$, the projection of R and S on the bisector of E. (a) choose R in case $S_c < \overline{R}$ (S_c is the circumcenter of edge E and node S). (b) choose S if $S_c > R_c$. (c) for $S_c \in [\overline{R}, R_c]$, a smoother gradation of triangle sizes can be achieved by choosing R if $R < S_c < L_1$, S_c if $L_1 < S_c < L_2$ and S if $L_2 < S_c$. Experimental results lead to $L_1 = 0.8$ and $L_2 = 0.4$.

4.2 Extension of the 2D Advancing Front Algorithm

In a simple approach, all geometric calculations for the advancing front method take place in the discrete 512x512 domain. Later the mesh nodes are mapped onto the object's surface.

Four extensions to the simple meshing algorithm in 2D have been formulated and investigated. One extension deals with the creation of well formed triangles in 3D. Two extensions also tackle the goal of adaptive meshing. User interaction is the last extension.

4.2.1 Adaptive Approach: Meshing in Transformation Space

Adaptive meshing is guided by the local surface curvature. The overhead to numerically calculate the local surface curvature of the object in order to come up with a modified metrics for all geometric calculations within the advancing front algorithm appeared to be unsupportable. Attempts have been made instead to globally transform the 512x512 domain into a curvature corrected 512x512 domain. The idea is to perform meshing in the transformed domain and to map the nodes back in order to increase node density in areas of high curvature. A simple discrete approximation of the maximums surface curvature is calculated by means of the angles formed by the local pixel and two neighbor pixels.

4.2.2 3D-Extension: Rotative Construction

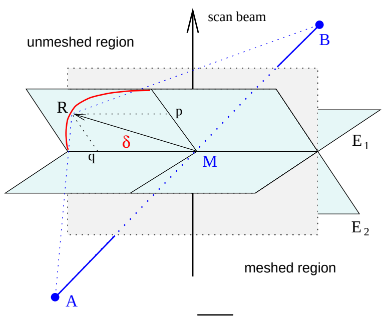

Figure 6: The edge $\overline{AB}$ is visited. A new candidate node is constructed by rotating an ideal triangle until the surface is reached.

The meshing results of the simple 2D approach are mapped on the 2.5D surface. In steep areas with surface normals very different to the laserscan beam (e.g. nose, chin), the simple mapping is causing heavy distortions in edge lengths and triangle shape.

In order to improve triangle shape in these areas, a numerical method to calculate the new candidate node R is introduced.

An ideal triangle is constructed vertically (with respect to scan direction) on top of the selected edge. It is rotated around the edge until the distance of R to the surface is minimized (Figure 6). The position of R can only be calculated numerically because the surface is explicitly given as a set of data samples.

The geometric calculations for this operation are straightforward.

4.2.3 Rotative Construction with adapting Edge Length

Whenever the center of a rotatively constructed triangle has a distance to the surface above a certain value, the new candidate R is deleted and the rotation will be repeated with a new triangle with reduced edge length.

4.2.4 User Interaction

The operator supervises the mesh creation process. With each run, the random selection strategy in the advancing front iteration is producing slightly different meshes. The operator may repeat the meshing until the result is most satisfactory. Besides, he may alter the region's edge length parameter.

In a post processing step, he is free to manipulate the position of single nodes to achieve further improvements.

5 Results

The simple 2D approach is already capable of producing a working face mesh with respected animation lines. However, twodimensional meshing fails in steep areas (see Figure 7).
For meshing in transformation space it has not been feasible to come up with a sound curvature based global transformation. The distortions in length and angle - both measures for triangle quality - are too heavy with a transformation from one discrete rectangular 512x512 domain into another.

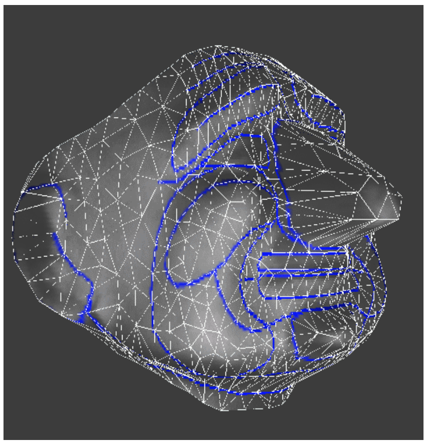
Figure 7: Mesh generated with 2D-meshing.

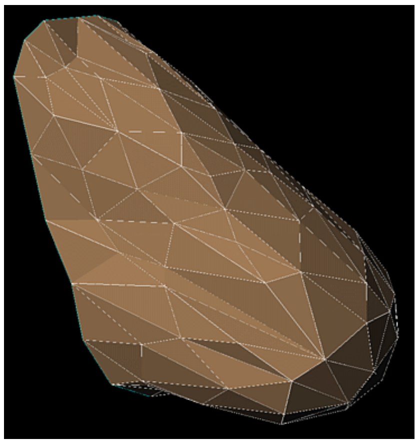
Figure 8: Rotative construction of new candidates can improve mesh quality on the flanks.

Rotative construction with adapting edge length increases node density in areas with high surface curvature and offers a further slight improvement of the surface approximation (Figure 8).

6 Conclusion

The test implementations have shown that the idea of an interactive meshing software based on the concept of animation lines is a promising approach for the efficient generation of animation meshes. Future work will have to include three fields: (1) The initialization of closely neighbored segments has to interact. (2) A true 3D meshing algorithm can significantly improve mesh quality. (3) The coarse discrete domain has to be replaced.

In special cases, when the segments that define a region are closely neighbored, the regular initialization can make it virtually impossible to create well shaped triangles in the domain. A simple solution is to split the segments edges and to perform meshing at half the intended edge length. A alternative way that does not cut down the mesh's geometric scale is to perform interdependent initialization on neighbored segments. It remains to investigate whether a modified Laplace smoothing [11] method can solve this task. The smoothing is applied to the segment’s initial nodes. The node relaxation is limited to positions on the underlying segment-curve.

The rotative construction of new candidate nodes and the boundary initialization are the only 3D extensions to the 2D advancing front algorithm. The decision between the candidates and the Delaunay triangulation of the meshed and unmeshed region are based on 2D metrics and may cause malformed triangles. CHEW [8] describes a formulation of the empty circumcircle criteria for curved surfaces. His algorithm for adaptive constrained Delaunay surface meshing performs a topological decomposition of the domain. Advancing front methods perform an elementwise decomposition that may cause poorer results in small regions with strong geometrical constraints.

Finally, the discrete 512x512 domain is too coarse to define small triangles in steep areas. It is necessary to interpolate a continuous domain. A discrete representation of animation lines does not allow proper discrete curvature analysis. A B-spline representation of the segments assures scale invariance and allows analytic curvature estimation for boundary initialization.

Future iterative methods of mesh optimization will automatically adapt the underlying animation mesh to the operator’s successive refinements of abstract animation expressions.

7 References

[1] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, W. Stuetzle, "Surface Reconstruction from Unorganised Points", Computer Graphics, SIGGRAPH ‘92 Proceedings, pp. 71-78, 1992

[2] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, W. Stuetzle, "Mesh Optimization", Computer Graphics, SIGGRAPH ‘93 Proceedings, pp. 19-26, 1993

[3] G. Turk, "Re-Tiling Polygonal Surfaces", Computer Graphics, SIGGRAPH ’92 Proceedings, pp. 55-64, 1992

[4] W. J. Schroeder, J. A. Zarge, W. E. Lorensen, "Decimation of Triangle Meshes", Computer Graphics, SIGGRAPH ’92 Proceedings, pp. 65-70, 1992

[5] M. J. De Haemer, M. J. Zyda, "Simplification of Objects Rendered by Polygonal Approximations", Comput. & Graphics, Vol. 15, No. 2, pp. 175-184, 1991

[6] Y. Lee, D. Terzopoulos, K. Waters, "Realistic Modeling for Facial Animation", Computer Graphics, SIGGRAPH ’95 Proceedings, 1995

[7] S. Farestam, "Generating Antisotropic Unstructured Grids for Two Dimensional Manifolds", Dissertation No. 819, CERFACS, Toulouse, 1993

[8] L. P. Chew, "Guaranteed-Quality Mesh Generation for Curved Surfaces", Computational Geometry, Vol. 9, No. 5, 1993

[9] S. Abramowski and M. Müller, "Geometrisches Modellieren", Number 75 in Reihe Informatik, BI-Wiss.-Verlag, 1991

[10] Robert J. Renka., "A Constrained Two-Dimensional Triangulation and the Solution of Closest Node Problems in the Presence of Barriers", Transactions on Mathematical Software, 22(1):1-8, March 1996

[11] Marshall Bern and Brian Eppstein, "Computing in Euclidean Space", chapter Mesh Generation and Optimal Triangulation, pages 23-90, World Scientific, 1992.