Faculty of Mathematical Sciences

University of Twente

University for Technical and Social Sciences

P.O. Box 217
7500 AE Enschede
The Netherlands
Phone: +31-53-4893400
Fax: +31-53-4893114
Email: memo@math.utwente.nl

Memorandum No. 1478

An orientation on surface reconstruction

G. Kloosterman and R.M.J. van Damme

December 1998

ISSN 0169-2690

An Orientation on Surface Reconstruction

Gertjan Kloosterman Ruud van Damme

December 14, 1998

Abstract

When reconstructing a surface from irregularly spaced data, sampled from a closed surface in 3D, we need to decide how to identify a good triangulation. As a measure of quality we consider various differential geometrical properties, such as integral Gaussian curvature, integral mean curvature and area. We furthermore study a non-functional approach, which is based on a mapping procedure. A locally optimal triangulation is then identified as a fixed point under the map. The optimisation methods all require an initial triangulation as a starting point. To find an initial triangulation, we look at growing and shrinking approaches.

Key words: Polyhedral metrics, triangulations, surface reconstruction, curvature.

1991 Mathematics Subject Classification: $65Y25, 53C42$.

Contents

Chapter 1

Introduction

In September 1998 I joined the Faculty of Mathematical Sciences of the University of Twente to start a research project with the topic surface reconstruction. For this project a proposal was sent to two research funds and in awaiting the outcome of the refereeing process, the research was to start September 15, and by December 15 the preliminary work should be concluded. After this period of time it should be known whether the work could be continued.

The result of these three months work are presented in this report. We were able to test a multitude of ideas, unfortunately not all giving good results. Hopefully this will provide a starting point for the researcher that is to continue this work.

This report will not be a mathematical work in the sense that all results and outcomes are rearranged to make for nice results, but will attempt to describe the chronological order in which ideas emerged and conclusions were drawn.

Chapter 2

Tight Triangulations

The first work that was performed was to re-implement the local tight optimiser that uses Lawson's procedure in 3D. The procedure was implemented as efficient as possible to allow for larger data sets to be considered. This efficiency is also required if we are to implement an initial triangulation algorithm based on this criterion.

For the background information on tight triangulations one should consult the literature [vDA95, AvD95]. We will just shortly discuss the implementation issues.

Every vertex of a triangulated or simplicial polyhedron, can be classified as one of three types. To explain the types we require the concept of a supporting plane. A vertex is said to have a local supporting plane if we can construct a plane containing the vertex, such that all other vertices connected to the vertex lie to one side of that plane.

We furthermore assume we are only dealing with orientable surfaces, thus we can assign an orientation to the edges incident on every vertex. We can now identify the following three types of vertices

- **Convex.** A vertex is classified as being convex (or concave) if every two consecutive edges incident on the vertex form a supporting plane of that vertex.
- **Saddle.** A vertex is classified as a saddle if there exists no supporting plane.
- **Mixed.** A vertex is mixed if it is neither convex nor saddle, thus it has a supporting plane, but there exist two successive edges incident on the vertex, that divide the set of adjacent vertices.

Each of these vertices can be assigned a certain energy corresponding to their integral absolute Gaussian curvature. But let us first introduce the

discretisation of ordinary Gaussian curvature. The Gaussian curvature of a point on a smooth surface, corresponds to the product of the maximal and minimal curvature at that point [dC76], *i.e.*

$$K = \kappa_1 \cdot \kappa_2,$$

Making the equivalent classifications as before, we can conclude that if both κ_1 and κ_2 are positive (negative) we have a convex (concave) point, and if their signs are opposite, the point is saddle. Notice the absence of a mixed point classification in the smooth case.

2.1 Discretised Curvature

Though K is really only defined for smooth surfaces, we can discretise the *integral* curvature. First we note that the curvature is located at the vertices. Assume now we have that $x_1, \ldots, x_N (x_{N+1} = x_1)$ are the coordinates of the ordered set of neighbouring vertices of v, and x_0 are the coordinates of v, then the Gaussian curvature at v can be discretised as:

$$K(v) = 2\pi - \sum_{j=1}^{N_v} \alpha_j, \quad \alpha_j = \angle(x_j - x_0, x_{j+1} - x_0). \tag{2.1}$$

It is known that for any closed orientable surface we have the following relation for the Gaussian curvature:

$$\int K dS = 2\pi\chi,$$

where χ is known as the Euler-Poincaré characteristic of the surface, which depends only on the topological type of that surface. From this we can draw the conclusion that minimising integral K will have no effect. Instead, we could integrate absolute K, thus minimise $\int |K| dS$. This does not correspond to minimising the absolute value of $K(v)$ in Equation 2.1! Instead we need to consider the decomposition of K into its positive and negative part, thus $K(v) = K^+(v) - K^-(v)$, and we wish to minimise

$$\hat{K} = \sum_{i=1}^{N} \hat{K}(v) = \sum_{i=1}^{N} (K^+(v) + K^-(v)) \tag{2.2}$$

Minimising this functional is known as optimising with respect to the tight triangulation criterion.

Unfortunately this optimisation criterion has some undesirable results, as it seems to create long thin triangles. This already happens on a relatively well sampled object, which will serve as our further illustration throughout this document, referenced to as the head data set. See Figure 2.1.

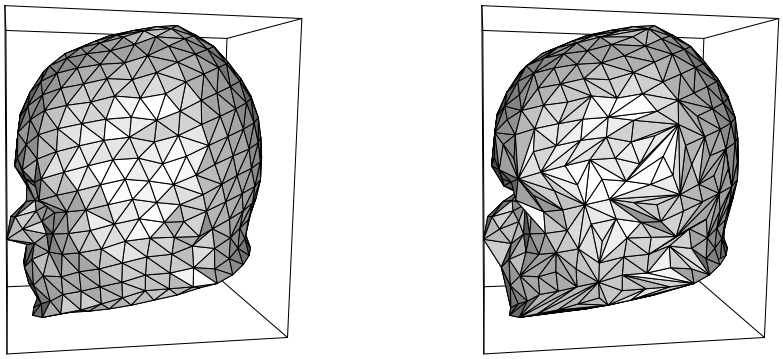

Figure 2.1: The scalp before (left) and after (right) applying the tight criterion

2.2 Shrinking from the Hull

It can be argued however, that the tight criterion is a good criterion, but although the local optimum is bad, the global optimum may still be good. As a result we also tried two shrinkers, explained shortly, from the hull based on the tight criterion, these two shrinkers are restricted to surfaces of genus O.

The starting point for both shrinkers is the convex hull of the object. Both shrinkers sequentially select a point that is not yet on the boundary surface, and a triangle on the boundary surface. The point is then added to the boundary by replacing this selected triangle by the other three possible triangles formed by the four points involved. After this the local optimisation algorithm is run to get as near to the global optimum as possible. The difference in strategy lies in the point–facet pair that is selected at every step. The strategies are the following two:

- Select that vertex–facet pair that has minimum real distance.
- Select that vertex–facet pair that has minimum curvature increase.

Both strategies turned out to give very poor results, with curvatures rating as high as $16 \times 4\pi$, whereas the initial triangulation was only $3.7 \times 4\pi$. The minimum possible curvature for any surface of genus O is 4π.

2.3 The Demix Approach

Though the results seem bad, the algorithm does return correct results on convex data, that is, it returns the convex triangulation if the data is convex. Whether this convex reconstruction is the best triangulation is questionable, but that discussion is postponed till later in this report.

We furthermore know that mixed vertices do not exist on a smooth surface, and that their curvature computation is slightly artificial. It can therefore be argued that mixed vertices are bad. One direct possible implementation that was therefore executed was the immediate 'demixing' of the mixed vertices. This means that all edges that are connected to a mixed vertex that do not lie on a possible spanning plane for that vertex are liable for flipping. We do not allow flips that create self intersections, or that create a mixed vertex elsewhere.

After this demixing routine has finished, the local optimisation procedure is called, and performs flips that decrease the integral absolute Gaussian curvature, under the constraint that no additional mixed vertices are created.

The first results of this approach applied to the head data set, after having it 'optimised' by the tight criterion, were promising. It turned out however that the constrained local optima were just good by pure chance. This could be concluded after we wrote a penalised tight criterion, in which the energy of a mixed vertex was multiplied by a reasonably large constant. This version of the tight criterion gave fewer mixed vertices, but poorly looking triangulations just the same.

We also embedded the demixing criterion as well as the penalised close variant in the shrinking method, to no avail. The resulting triangulations took an enormous time to compute, and were very bad.

After these experiments, it can be said that the tight criterion either is not good or, that it is nearly impossible to reach the global optimum and that all local optima are very bad. Since we have not succeeded in proving global optimality for any of the bad solutions found, nothing can be said with certainty. What can be said though is, that in practice these methods are useless, since we can not hope to find anything better than local optima in a reasonable amount of time.

Chapter 3

Area and NP–hardness

We furthermore briefly considered implementing the area criterion, as proposed by O'Rourke [O'R81]. However, one short look at the article showed that even in the simple case he studied long thin triangles where already locally optimal. A recent study, performed by Kistemaker in her final work for graduating at the faculty of Mathematical Sciences in Twente, shows that the area criterion gives erroneous results in the globally optimal case for data sampled from two slices. See figure 3.1

In his article O'Rourke assumes that finding the minimal area polyhedron through a set of data is an NP–hard problem, since it seems to be a generalisation of the two dimensional Travelling Salesman Problem. Though this statement is very likely to be true for a general data set, there are some comments that can be made regarding his conjecture.

3.1 Area and NP–hardness

The first comment on the NP–hardness claim for the area criterion is that distance, used in the Travelling Salesman Problem in 2D, does not properly generalise to area, used in the minimal area polyhedron problem in 3D. Indeed, assume we have half a circle and a point above it, as in Figure 3.2. The minimal length curve through this data is obviously the convex hull, since the shortest distance between two points is a straight line, and we require two straight lines to make the data lie on a closed curve.

Now assume however that we rotate this data along the axis through the point and through the middle of the circular arc. This will lead to data consisting of half a sphere and a point above it. The convex hull is also the rotation of the convex hull we obtained in 2D, *i.e.* a cone with the point as apex connected to the half sphere. This is however *not* the minimum area

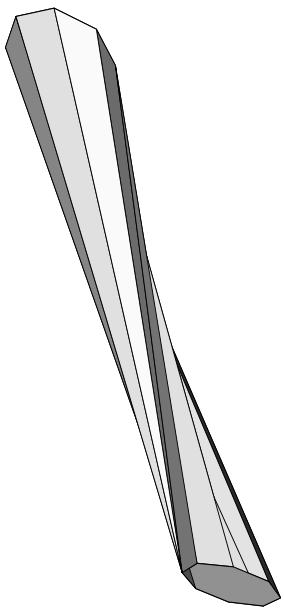

Figure 3.1: Two slices skewed cilinder

surface through the data. To motivate this, assume that the half sphere has radius R, and that the cone has base radius r, we then need to close the rest with a disc with inner radius r and outer radius R. The total area of this object being with the point h above the half sphere

$$A(r) = \pi r h + \pi(R^2 - r^2) + 2\pi R^2$$

From which it can be seen, that the minimum of this function is not necessarily the case, where $r = R$.

The second comment on the conjecture is that although it is quite likely an NP–hard problem, this is not the case for the special data sets we consider in the surface reconstruction problem. Equivalently in the Travelling Salesman Problem the difficulty lies in being able to deal with any data set, no matter how bizarre. In surface reconstruction we can not hope to get any good results unless the data is well sampled, thus we do not have the same difficulties,

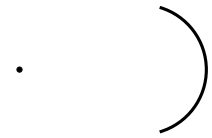

Figure 3.2: Convex data

and therefore the NP–hardness of this class of problems is not —in general— a restriction.

3.2 Further Results on NP–Hardness

We have given the problem of NP–hardness a considerable amount of thought. The problem in proving NP–hardness arises from the fact that all complexity results have been proven for essentially two dimensional problems. Though it would seem the complexity only increases with the dimension, it also increases the complexity of the proof.

The problem already appeared in the previous section, where it seems that minimising length of a polygon can be generalised to minimising area of a polyhedron, this generalisation is not a proper one. Equivalently so with curvature, the 2D discretisation of curvature, is not equivalent to the 3D discretisation of curvature. Even if NP-hardness for the 2D case could be proven (which is a problem in itself) it is not trivial to use this result to obtain a complexity statement for the 3D case.

As another example, we consider the sculpting process introduced by Boissonnat [Boi84]. The sculpting process seems to be a generalisation of finding a Hamilton cycle in the Delaunay triangulation. Deciding whether a graph contains a Hamilton cycle is an NP-complete problem. Whether this is true for the Delaunay triangulation is unknown, though it was proven by Dillencourt [Dil87] that it is possible that a non-degenerate 2D Delaunay triangulation does not contain a Hamiltonian cycle. It is therefore likely that for the generalised problem there exist tetrahedralisations that do not contain a Hamilton polyhedron.

Chapter 4

Triangulation of Starlike Objects

A starlike object is defined as an object that contains a point from which the whole boundary surface is visible. Here visible means that any line from this point to a point on the boundary surface does not intersect the boundary surface elsewhere.

A quick way to find a reasonable triangulation of these objects is by first obtaining the convex hull of the object. Then project all internal points on the nearest facet. This projection is unique. Next compute the barycenter of the hull, which lies in the interior. All points are now visible from this barycenter. It is now possible to project the data onto a sphere of arbitrary radius, with the barycenter as center of this sphere.

The sphere can easily be triangulated, by its convex triangulation. This convex triangulation now is also a non self–intersection triangulation for the original data. The method works pretty good for the head data set.

An immediate extension of this idea to objects that are non–starlike is the following. We can define a distance function that is defined as the sum over all minimal distances from the internal points to the current boundary description. As the next point to include in the boundary description of the object to be triangulated would be that point that reduces this distance the most.

It turns out however, that just inserting one point does not suffice in the case of fairly complex data. Unfortunately this introduces some considerable amount of computational complexity, already in the 2D case.

Chapter 5

Stretched Triangulations

In this chapter we propose a method that maps triangulations of genus O objects to the sphere. The proposed mapping procedure considers edges as pieces of elastic band. Allowing vertices to move due to the tensions in the edges will eventually lead to a starlike polyhedron, which can be projected onto the unit sphere. This mapping represents a possible homeomorphism from the sphere to the surface. Any triangulation of the original data coincides with a triangulation of the mapped points on the sphere and visa versa. We replace the original triangulation with the convex triangulation of the spherically mapped points, which is the best approximation to the spherical surface, and as such, if the homeomorphism is good, a good approximation to the original surface.

5.1 The Sphere Map

Our objective in this section is to develop an algorithm that maps points given a triangulation to the unit sphere. This mapping should preserve relative distances as good as possible in a certain sense over the triangulation. As small edges give rise to small relative distances, these should be better preserved than long relative distances. This map can be seen as a discrete approximation to the diffeomorphism from the unit sphere to the surface we are seeking.

Short edges in the triangulation should have small distance on the unit sphere, and will thus be connected in the convex triangulation of the unit sphere. Therefore, long edges will then tend to disappear. This will lead to an approximation of our minimum maximum edge length triangulation.

The idea behind the algorithm is as follows. Assume we have an elastic balloon and we span this balloon over the vertices, according to the initial

triangulation. If we now were to release the vertices, this balloon will deform to a convex shape. Eventually it will have reshaped such that the shape becomes starlike from the barycenter, *i.e.* all facets of the triangulation are visible from the barycenter. We can then project the facets radially onto the unit sphere, creating a triangulation of points on the unit sphere that is not self intersecting, if the original triangulation was not self intersecting. The elastic deformation process is the topic of discussion in Section 5.1.1.

We can then redistribute these points over the sphere according to our preservation criterion, and retriangulate. This triangulation is over the vertices $\hat{v}_i$, which is the projection of the vertices v_i of the original triangulation on the unit sphere. We now replace the original triangulation by the spherical triangulation. Thus if the edge $(\hat{v}_i, \hat{v}_j)$ is an edge in the convex spherical triangulation, then (v_i, v_j) is an edge in the retriangulation of the original data. Redistribution and retriangulation are the topic of Section 5.1.2.

5.1.1 Simulating Elastic Deformations

The key idea in finding our approximation to a diffeomorphism from the initial triangulation to the sphere, is finding a simulation of a physical process that behaves as the diffeomorphism. This is in our case the behaviour of a balloon. We will however not follow the physical balloon, but a mathematical model that is based on it, since it is only our intent to find *a* diffeomorphism from the triangulation to the sphere, not *the* diffeomorphism.

Our elastic deformation process, consists of two stages:

- First we allow free movement of all vertices that are not on the convex hull of the triangulated object.
- In the second stage we allow free movement of all vertices.

The reason for separating the two processes is to allow the barycenter to move to the interior of the object if it is not already there. The barycenter of the points is used to re-scale the data in the second step, and to add a distance scaling to the stresses on the vertices. We will now discuss this more precisely.

Stress on Vertices

Instead of simulating a balloon, that contracts by area, we will simulate a model where the vertices are connected by springs with zero rest length. All edges are assumed to have unit force in the initial triangulation. We take

the following formula for the resulting force acting on vertex v_i.

$$\mathbf{F}(v_i) = \sum_{j|e_{ij}=1} \kappa_{ij}\mathbf{u}_{ij}. \tag{5.1}$$

Here the κ_{ij} are the spring constants, and $\mathbf{u}_{ij}$ is the relative length of the edge in the current triangulation. This relativity can be expressed in numerous ways, we can eg. divide by the square root of the area, or by the sum of lengths of adjacent edges. We have chosen to divide by the total edge length of the triangulation. Thus

$$\mathbf{u}_{ij} = \frac{\mathbf{x}_j - \mathbf{x}_i}{\sum_{k\ell|e_{k\ell}=1} \|\mathbf{x}_k - \mathbf{x}_\ell\|}. \tag{5.2}$$

The point $\mathbf{x}_i$ denotes the location of the vertex v_i in space. By meeting our demand of unit force contributed by a given edge in the initial triangulation, we have to choose the κ_{ij} as the inverse of the length of the initial relative edge length. Thus

$$\kappa_{ij} = \frac{1}{\mathbf{u}^0_{ij}}, \tag{5.3}$$

where the superscript 0 indicates that we take $\mathbf{u}$ as in the initial triangulation.

Tightening to the Hull

The first part of the process now consists of repeatedly determine all the stresses, and the resulting direction of the free vertices. Then we take a small step in the direction of decreasing force. The largest length of any step is at most δ times the smallest real edge length, where δ is a small number. The reason for this small step is twofold:

- The creation of self intersections is prevented.
- It damps the step, thereby preventing oscillations.

Thus we have the following Jacobi step algorithm:

Distance Scaling

In the next step, we allow *all* vertices to move. If we do this and follow the same algorithm as previously, two things happen. First of all the polyhedron becomes smaller and smaller, thus it changes scale. And secondly the polyhedron can become very elongated, which is undesirable, since the contraction to the sphere will then become bad.

Algorithm 1 Elastic Tightening to the Hull

$\kappa_{ij} = \frac{1}{\|\mathbf{u}_{ij}^0\|}$
repeat
  $\mathbf{F}_i = \mathbf{F}(v_i)$
  $M = \max_i \|\mathbf{F}_i\|$
  $m = \min_{ij} \|\mathbf{u}_{ij}\|$
  if $M > \delta m$ **then**
    $f = \delta \frac{m}{M}$
  else
    $f = 1$
  end if
  $\mathbf{x}_i = \mathbf{x}_i + f \cdot \mathbf{F}_i$
until $M < \epsilon m$

To overcome both problems we use scaling. For the first problem we inflate the object from the barycenter after every step that is taken, such that the minimum distance to the barycenter is always the same.

For the second problem we scale the stresses. The idea is that the further a vertex is from the barycenter, the less outward pressure it would have if it were lying on a balloon, and thus the more it is free to move. Thus if $m_B = \min_i \|\mathbf{x}_i - \mathbf{x}_B\|$, where $\mathbf{x}_B$ denotes the current barycenter, we have the following expression for the force exerted on a vertex v_i.

$$\hat{\mathbf{F}}(v_i) = \frac{\|\mathbf{x}_i - \mathbf{x}_B\|^p}{m_B} \sum_{j|e_{ij}=1} \kappa_{ij}\mathbf{u}_{ij}, \tag{5.4}$$

where $p \geq 0$. For $p = 0$ we obtain our initial schedule. Since the distance in the numerator is always larger than m_B, far away distances are more freely moved for p increasing. Multiplying by a constant for distance is no use, since movements are all relative.

We then have as before the κ_{ij} the Jacobi step algorithm as in Algorithm 2. The resulting object is starlike and can be projected on the unit sphere without causing self intersections.

5.1.2 Redistributing and Retriangulating Data on the Sphere

We now managed to map our initial triangulation onto the sphere, without causing self intersections. Since relative length is not necessarily preserved under a homeomorphism, there is still some freedom as to where points should

Algorithm 2 Elastic Tightening

Require: κ_{ij} as in Algorithm 1
  repeat
    $\mathbf{F}_i = \hat{\mathbf{F}}(v_i)$
    $M = \max_i \|\mathbf{F}_i\|$
    $m = \min_{ij} \|\mathbf{u}_{ij}\|$
    if $M > \delta m$ **then**
      $f = \delta \frac{m}{M}$
    else
      $f = 1$
    end if
    $\mathbf{x}_i = \mathbf{x}_i + f \cdot \mathbf{F}_i$
    Rescale $\mathbf{x}_i$
  until The polyhedron is starlike

be located on the sphere. We can follow the same reasoning as above, with the additional condition that points are restricted to the sphere.

We can now change our stress constants κ_{ij} such that edges that are relatively small are better preserved than long edges. One possible way of doing this, is by noting that all $\kappa_{ij} \geq 1$, and $\kappa_{ij} \geq \kappa_{i'j'}$, if $\|\mathbf{u}^0_{ij}\| \leq \|\mathbf{u}^0_{i'j'}\|$. Furthermore shorter edges should contract faster than long edges, and this can be achieved by taking $\hat{\kappa}_{ij} = \kappa^q_{ij}$, where $q \geq 1$.

At the end of every Jacobi step, we rescale the points back to the unit sphere. There is no distance rescaling, since all points have distance 1 to the center of the unit sphere. We can terminate the iterations, when the actual change in position of the vertices becomes small.

The retriangulation of the object is now trivial, we can just take the convex triangulation. This triangulation of the sphere is now taken as the new triangulation for the initial points. Thus if the point $\mathbf{x}_i$ is mapped onto the spherical point $\hat{\mathbf{x}}_i$, and if $(\hat{\mathbf{x}}_i, \hat{\mathbf{x}}_j)$ is an edge in the convex triangulation of the sphere, then we have that $(\mathbf{x}_i, \mathbf{x}_j)$ is an edge in the new triangulation of the original data. The process could be seen as the deformation of the convex triangulation by pushing vertices to their original positions.

We can now state the redistribution algorithm as in Algorithm 3.

The total algorithm can be run several times to hopefully find a fixed point of the map. All instances we tested converged to such a fixed point, though these points are not unique.

Algorithm 3 Redistribution of the Spherical Points

$\hat{\kappa}_{ij} = \kappa_{ij}^q$
repeat
  $\mathbf{F}_i = \sum_{j|e_{ij}=1} \hat{\kappa}_{ij} \mathbf{u}_{ij}$
  $M = \max_i \|\mathbf{F}_i\|$
  $m = \min_{ij} \|\mathbf{u}_{ij}\|$
  if $M > \delta\ m$ **then**
    $f = \delta \frac{m}{M}$
  else
    $f = 1$
  end if
  $\mathbf{y}_i = \mathbf{x}_i$
  $\mathbf{x}_i = \mathbf{x}_i + f \cdot \mathbf{F}_i$
  $\mathbf{x}_i = \frac{\mathbf{x}_i}{\|\mathbf{x}_i\|}$
until $\|\mathbf{x}_i - \mathbf{y}_i\| < \epsilon_R$

5.2 Example: A Head

As an illustration of our method, we show how it performs on a (bad) initial triangulation of the head. In Figure 5.1 we show the results after running the algorithm several times.

In the first picture at the top left corner, the head is shown which is input to the algorithm. The triangles at the cheek and ear locations are ill placed, and moreover the nose is poorly triangulated. It especially contains a flat section towards the cheek and chin.

In the top right corner, the result is shown after one complete step of the algorithm. It is clear that the bad triangles near the nose are all swapped away, as is the case with the other initially very poor locations. Repeating this process eventually leads to a triangulation which is a fixed point of this algorithm.

5.3 Conclusions

We have introduced a mapping from a surface to the sphere, based on a simulation of elastic deformation processes. This map can then be used to retriangulate the set of original data. The number of parameters available in the simulation process is quite large, and instead of presenting one possible way of obtaining the sphere map, we have in fact introduced a class of methods. Which parameter setting works best for which type of problems

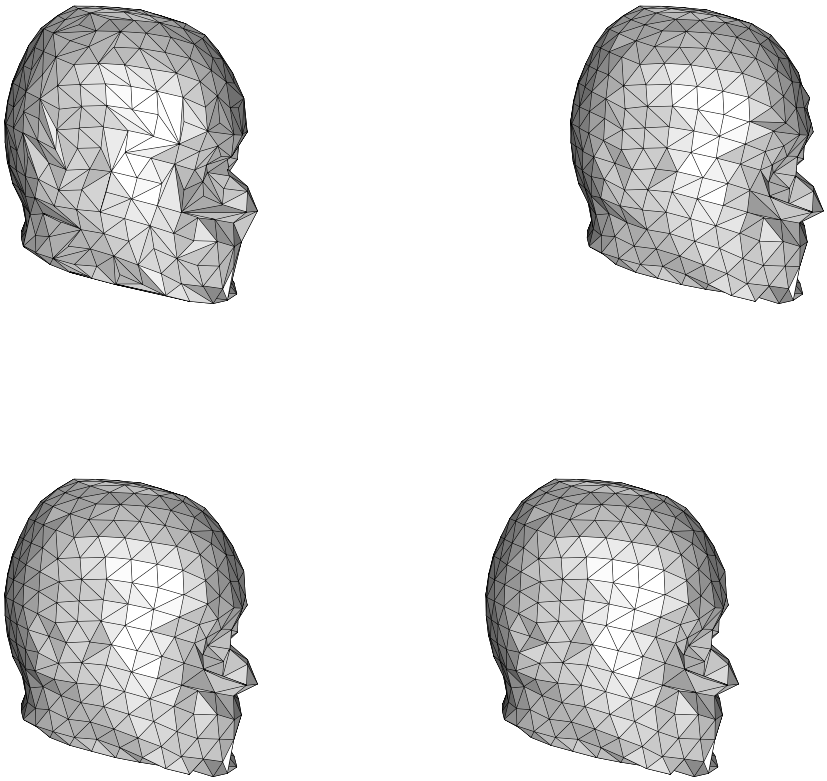

Figure 5.1: Successive stages of bloating

is an open problem. Also some parameter settings may not converge to a starlike object (in a reasonable amount of time) or the resulting spherical triangulation may give self intersections when applied to the initial data.

We have in fact changed the surface reconstruction problem from defining optimality criteria on an arbitrary surface, to finding the best possible mapping of points to the unit sphere. Such a mapping must exist, since the objects we consider are topologically homeomorphic to it.

Actually creating the triangulation has then become an easy task, since the best approximation to the sphere is the convex triangulation of the sample points on it. The triangulation on the original data can then be seen as relocating the points to their original positions in three dimensional space, but keeping the triangulation intact.

Chapter 6

Growing Initial Triangulations

An optimisation strategy being available, we require initial triangulations to complete the surface reconstruction process. One possible way to try and create an initial triangulation is by growing: the concept is simple, in every step we add one triangle to the surface description in a strictly local optimal sense. This process eventually terminates after the surface is closed.

Implementing this idea requires several technical concepts. First of all we require a description of the current triangulation boundary. This triangulation boundary, consists of an ordered list of edges, where the ordering is specified by the orientation of the triangulation. Every edge on the boundary has one triangle adjacent, it requires a second triangle to become internal. The boundary description is not restricted to one single cycle, the actual boundary description can consist of a number of these cycles. There is however, only one connected interior at every step in the growing process. This is necessary to avoid orientation conflicts that could occur if we had several interiors, and would also make the joining of these separate triangulations more complex.

6.1 Configurations

There are now several configurations that can occur when a new triangle is created.

- The new triangle consists of one edge e (and thus two vertices) on the boundary, and one vertex v which is not connected to the triangulation. This configuration is called a *grow*. It removes the edge $e = (v_1, v_2)$ from the boundary, and places two new edges in its place (v_1, v) and (v, v_2).

- The new triangle is spanned by exactly two consecutive edges on one boundary cycle, *i.e.* (v_1, v_2) and (v_2, v_3). This configuration is called a *shortcut*. It removes the two edges from the boundary and replaces it with one single edge (v_1, v_3). If vertex v_2 no longer lies on any boundary cycle, then it is placed on the list of internal vertices. Internal vertices can not be part of any new triangles created later in the growing process.

- The new triangle is spanned by three consecutive edges on one boundary cycle. This is called a *collapse*. All three edges are removed from the boundary. All three vertices are liable for being put on the list of internal vertices, depending upon their appearance on other boundary cycles. The three edges formed a boundary cycle, and this cycle has thus disappeared. If no more boundary cycles would remain after a collapse, the growing has finished, and we are left with a closed surface.

- The new triangle is spanned by one edge e on one boundary cycle, and a vertex on another boundary cycle. This configuration is called a *join*. One should be precautious in joining boundaries, the only time a join should be made is when there is a reasonably long path through the triangulation from the first boundary to the other boundary. The edge is removed, and two new edges are created such that the two boundaries are joined to form one.

- The new triangle is spanned by one edge e and a vertex on the same boundary cycle. This is called a *split*. Again one should be cautious in creating splits. One boundary cycle is cut into two by creating two edges. The orientation dictates the manner in which the fragments are created.

Any other configuration than the ones mentioned above are invalid.

6.2 Growing Criteria

The criteria we tested using this grower are

- Minimal edge length.

- Maximal angle.

The former of the two does not suffice as a selection rule alone. It is quite simple to create self intersections this way. So we require an additional

rule that states that the angle formed by the new triangle created and the triangle opposite the edge should not be too small.

The version where the best was chosen among any edge in the triangulation seemed to give worse results than the version in which every vertex needed to be closed one by one.

Though the minimal edge length increase seemed to work reasonably for the data set of the head, it locks for the holes data set as presented by Hoppe [ea92].

The maximal angle criterion is worse, since the first few triangles have good aspect ratio, but finally closing the triangulation results in very thin triangles.

6.3 Maximising Minimal Angle

Optimising with respect to the minimal angle, *i.e.* Delaunay 2D over polyhedra is very bad. See Figure 6.1.

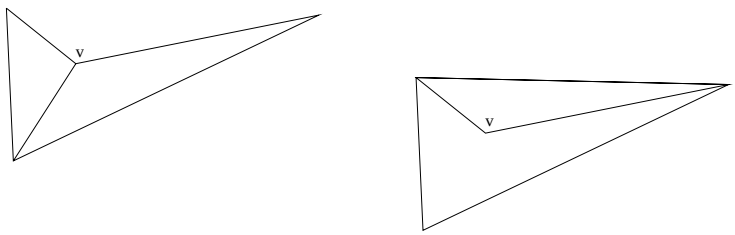

Figure 6.1: Angle bad: v just above plane.

Chapter 7

Absolute Mean Curvature

Absolute mean curvature turns out to be a good criterion. It is basically equivalent to minimising the maximum curvature over the triangulation. This can be seen from the fact that we have only one principal curvature, the other being 0.

What we wish to minimise is H absolute:

$$\int_{\Omega}(|\kappa_1 + \kappa_2|)dS, \tag{7.1}$$

but even rather

$$\int_{\Omega}(|\kappa_1| + |\kappa_2|)dS, \tag{7.2}$$

It turns out we have the latter term if we optimise over the triangulation, this is due to the following reasoning:

- Vertices have no contribution in the integral, since they have zero measure.
- Points inside triangles have no contribution, since both principle curvatures are zero.
- Edges have contribution, but only of one principal curvature, the other being zero. Moreover the two principal curvatures are orthogonal.

This leads to the following discretisation over the edges, assume ρ is the radius of the cylinder used to smooth it.

$$|H(e_{ij})| = \frac{\alpha}{\rho}\rho\|\mathbf{x}_j - \mathbf{x}_j\| \tag{7.3}$$

Which leads to

$$|H(eij)| = \alpha\|\mathbf{x}_j - \mathbf{x}_j\|.$$

Since only one of the two principal curvature is non zero, this is equivalent to Equation 7.2 and also

$$\int_{\Omega} \max(|\kappa_1|, |\kappa_2|) dS. \tag{7.4}$$

This criterion gives very promising results, see Figure 7.1

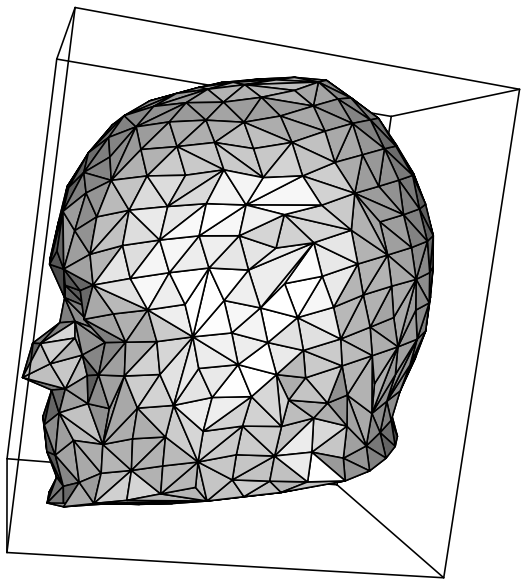

Figure 7.1: Mean Curvature Minimised

Chapter 8

Conclusions

We can conclude that the minimising Gaussian curvature *i.e.* the tight criterion, is not very applicable to general datasets. This is due to the fact that seemingly most local optima are bad, and we can not hope to find the global optimum. Not even after using a modified approach. The Area criterion has been shown to lead to poor results even in the globally optimal case. The only promising functional criterion seems to be minimising absolute mean curvature.

The non-functional criterion is the sphere mapping procedure. This also seems to give reasonable results, but it requires a triangulation as input that is not too bad. Furthermore there are quite a few parameters in the method that need to be chosen.

As a last we have considered growing triangulations. The method we pursued seems to give good results for the head data set, but it seems to lock for data sets that are sampled irregularly.

Bibliography

[AvD95] L. Alboul and R. van Damme. Polyhedral metrics in surface reconstruction: tight triangulations. Memorandum no. 1275, University of Twente, Faculty of Mathematical Sciences, 1995.

[Boi84] J.D. Boissonnat. Geometric structure for 3-dimensional shape representation. *ACM Trans. on Graphics*, 3:266–286, 1984.

[dC76] M. P. do Carmo. *Differential Geometry of Curves and Surfaces*. Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1976.

[Dil87] M.B. Dillencourt. A non-Hamiltonian, nondegenerate Delaunay triangulation. *Information Processing Letters*, 25:149–151, 1987.

[ea92] H. Hoppe et al. Surface reconstruction from unorganized points. *Computer Graphics Proceedings*, 26:71–78, 1992.

[O'R81] J. O'Rourke. Triangulation of minimal area as 3D object models. In *Proc. of the Intern. Joint Conf. on AI 81*, pages 664–666, Vancouver, 1981.

[vDA95] R. van Damme and L. Alboul. Tight triangulations. In M. Dæhlen, T. Lyche, and L. L. Schumaker, editors, *Mathematical Methods for Curves and Surfaces*, pages 517–526, Nashville TN, 1995. Vanderbilt University Press.