Computing (2004)
Digital Object Identifier (DOI) 10.1007/s00607-004-0056-9

Computing
Printed in Austria

Robust Spherical Parameterization of Triangular Meshes

A. Sheffer, Vancouver, **C. Gotsman,** Haifa, and **N. Dyn,** Tel Aviv

Received March 18, 2003; revised October 29, 2003
Published online: March 18, 2004

Abstract

Parameterization of 3D mesh data is important for many graphics and mesh processing applications, in particular for texture mapping, remeshing and morphing. Closed, manifold, genus-0 meshes are topologically equivalent to a sphere, hence this is the natural parameter domain for them. Parameterizing a 3D triangle mesh onto the 3D sphere means assigning a 3D position on the unit sphere to each of the mesh vertices, such that the spherical triangles induced by the mesh connectivity do not overlap. This is called a *spherical triangulation*. In this paper we formulate a set of necessary and sufficient conditions on the spherical angles of the spherical triangles for them to form a spherical triangulation. We formulate and solve an optimization procedure to produce spherical triangulations which reflect the geometric properties of a given 3D mesh in various ways.

AMS Subject Classifications: 68U05, 68U07, 65D18, 51N05.

Keywords: Parameterization, mesh processing, spherical parametrization, spherical embedding.

1. Introduction

Given a 3D mesh, many computer graphics applications map texture onto it in order to produce more realistic renderings. Since the texture map is typically a 2D image, this operation requires assigning 2D plane coordinates to each of the mesh vertices. If the mesh consists of triangles, the texture pixels are then mapped in a piecewise affine manner from the texture plane to the 3D triangle faces. Since a texture map is topologically equivalent to a disk, the most natural result is obtained when the 3D mesh also has the topology of a disk (i.e., manifold, genus-0 with a boundary). In this case, since many parameterizations exist, it is a major challenge to produce that which best fits the geometry of the 3D mesh, minimizing some measure of *distortion*. Most of the recent works on the subject of parameterization (e.g., [2], [6], [9]) have focused on defining the distortion, and showing how to minimize it. The different disk parameterization methods published may be partitioned into two categories: Those who require that the boundary parameter values are pre-defined and form a convex shape (e.g., [2]) and those who impose no special shape on the boundary (e.g., [6], [9]).

The parameterization problem is more complicated when the mesh does not have the topology of a disk. Many manifold 3D meshes are closed (i.e., have no

boundary), so are topologically equivalent to a sphere, which is fundamentally different from the topology of the texture map. The most straightforward way to work around this is to somehow form an artificial "boundary" and then use the methods designed for disks. A triangular boundary may be formed by removing an arbitrary triangle from a closed mesh. A more elaborate boundary may be formed by cutting along mesh edges [8]. This, however, usually introduces discontinuities where the edges are cut and may sometimes increase the parameterization distortion in comparison to spherical parameterization.

While parameterizing to the plane is the most natural way to perform texture-mapping, this is less natural for other mesh processing operations which also require a parameterization. For applications such as morphing [1] and remeshing [5] it is best to parameterize the mesh over a domain which is topologically equivalent to it. So if the mesh has the topology of a sphere, it is best to use a spherical parameter domain. Parameterizing a 3D mesh over the sphere is equivalent to *embedding* its connectivity graph on the sphere, such that all the resulting spherical polygons do not overlap. A classical result of Steinitz is that a graph may be embedded on the sphere iff it is planar and 3-connected [4]. So a closed genus zero triangulation can always be mapped to a *spherical triangulation*.

The simplest way to map a closed triangulation to the sphere is to cut out one triangle to serve as a boundary, parameterize the open mesh over the unit triangle, and then use the inverse stereo projection to map the disk to the sphere (e.g. [3]). The boundary triangle will contain the north pole of the sphere. The main problem with this method is that most disk parameterization methods, when faced with a triangular boundary (containing only three vertices), tend to cluster all the interior vertices in the center of the triangle, leading to significant distortion in both the disk and spherical parameterizations. Also, the inverse stereo projection usually does not preserve any of the geometric properties of the planar triangulation, so the result is quite distorted.

Another popular method is to cut the mesh into two pieces, each topologically equivalent to a disk, parameterize each over a planar disk with a common boundary, and then map each disk to a hemisphere. The common boundary guarantees that the two hemispheres fit together at the equator. Each disk parameterization will be better than the one described in the previous paragraph, so the result will be less distorted. However, the result will depend strongly on the specific cut used to obtain the two disks, so it can be difficult to optimize. In addition the cut line will show up as a parameterization artifact.

Several methods for direct parameterization on the sphere have been developed. The only one that seems to guarantee a valid spherical triangulation is that of Shapiro and Tal [7]. This method works by simplifying the mesh by vertex removal until only a tetrahedron remains. The tetrahedron is embedded on the sphere, and then the vertices are inserted back one by one, so that the convexity of the shape is preserved throughout the process. While this is quite an efficient process, it is difficult to steer the parameterization towards any specific target, due

to its greedy nature. Another direct embedding method was suggested by Kobbelt et al. [5] and Alexa [1]. It is an iterative procedure, attempting to converge to an embedding by applying local improvement (relaxation) rules. In practice, this method works well in many cases. However, there is no guarantee that it will terminate, and, even if it does, that the resulting embedding will be valid (i.e., contain non-overlapping triangles).

The common denominator of all these methods, as well as most of the disk parameterization methods, is that the algorithms search for the values of the parameters. In this paper we present an algorithm for spherical embedding that extends the approach of Sheffer and de Sturler [9] for disk embeddings. Their algorithm works by searching for the *angle*s of the planar triangles rather than the positions of the triangle vertices in the plane. A set of necessary and sufficient conditions for the angles to form a valid planar triangulation is formulated. These turn out to be mostly linear equalities and inequalities and just one non-linear equality relation involving the sines of the angles. Once the angles have been determined, the triangles are formed by an "unfolding" operation. Thanks to the approach of working with angles, the resulting triangulation is valid and not forced to have a convex boundary.

In this paper, we extend the method of Sheffer and de Sturler to parameterization on the sphere. Here too, we work with angles, except these angles are spherical angles, rather than planar angles. We formulate a set of necessary and sufficient conditions for the angles to form a valid spherical triangulation. Once the angles are determined, the spherical triangulation may be generated.

A preliminary version of this paper was presented at the 4th Israel-Korea Bi-National Conference on Geometric Modeling and Computer Graphics [10].

2. Some Spherical Geometry

In this section we review some of the basics of spherical geometry and trignometry on the unit sphere that we will need in the sequel. See Fig. 1 for an illustration. A spherical triangle is the region enclosed by three great circles on the sphere (a great circle is a circle on the sphere whose center is the origin). Denote the length of the arcs who are the sides of the spherical triangle by a, b and c. These are identical to the planar angles of the wedges defined by the origin and pairs of

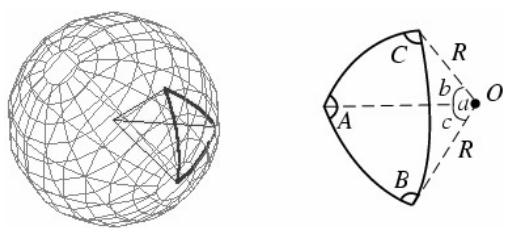

Fig. 1. Spherical geometry

vertices of the triangle. Certainly each of these is less than π. The spherical *defect* of the triangle is $\mathrm{D} = 2\pi-(a+b+c)$. A spherical *angle* is the dihedral angle between the two planes defined by the two great circles, and we denote these by A, B and C. The sum of the spherical angles of a spherical triangle is always more than π and less than 3π. The spherical *excess* of that triangle is $\mathrm{E} = A+B+C-\pi$. The *solid angle* defined by a spherical triangle is the area of the region on the sphere defined by that triangle, and is equal to the excess of the triangle. Hence the sum of all solid angles and the sum of all excesses in a spherical triangulation is 4π. Note that the sum of all spherical angles around any vertex in a spherical triangulation is exactly 2π.

The spherical sine rule is:

$$\frac{\sin A}{\sin a} = \frac{\sin B}{\sin b} = \frac{\sin C}{\sin c}. \tag{1}$$

and the spherical cosine rule is:

$$\begin{aligned} \cos A &= -\cos B \cos C + \sin B \sin C \cos a, \\ \cos B &= -\cos C \cos A + \sin C \sin A \cos b, \\ \cos C &= -\cos A \cos B + \sin A \sin B \cos c. \end{aligned} \tag{2}$$

3. Conditions for Spherical Triangulation

Given a closed manifold genus-0 triangle graph (connectivity) of n vertices, embedded on the sphere, this connectivity forms a valid spherical triangulation iff a set of conditions on the spherical angles and excesses of the embedding hold. By Euler's theorem, the triangulation has $t = 2n-4$ triangles, and $m = 3n-6$ edges.

Following Sheffer and de Sturler [9] we denote the spherical angles of the triangles as α_i^j, $i = 1,\ldots,t$, $j = 0,1,2$ in counter-clockwise order around the face normal. We denote the spherical excess of each triangle as e_i: $i = 1,\ldots,t$. Furthermore, denote by $V^j(k)$ ($j = 0,1,2$) the lists of indices of the triangles whose α^j angles are incident on the kth vertex, by $I_1(l)$ and $I_2(l)$ the indices of the two triangles incident on the lth edge, and by $J_1(l)$ and $J_2(l)$ the indices of the angles opposite the edge in those two triangles. Now the following $8t+n+m$ conditions ((3)–(8)) on $4t$ variables (α^i_j and e_i) are necessary and sufficient for a valid spherical triangulation:

$$\alpha_i^j > 0 \quad i = 1,\ldots,t, \;\; j = 0,1,2 \tag{3}$$

$$e_i > 0 \quad i = 1,\ldots,t \tag{4}$$

$$e_i < 2\alpha_i^j \quad i = 1,\ldots,t, \;\; j = 0,1,2 \tag{5}$$

$$\alpha_i^0 + \alpha_i^1 + \alpha_i^2 - e_i - \pi = 0 \quad i = 1,\ldots,t \tag{6}$$

$$\sum_{j=0}^{2} \sum_{i \in V_j(k)} \alpha_i^j - 2\pi = 0 \quad k = 1,\ldots,n. \tag{7}$$

Conditions (3)–(4) and (6)–(7) are identical to those of the planar case [9] with the addition of the angular excess for each triangle. Condition (5) results from the relationship between spherical angles and area on the sphere. The sine condition described in [9] is replaced by a set of conditions (8) derived from the cosine rule (2). In the plane this rule follows from the fact that the sum of angles in a triangle is π. On the sphere this is not the case, so we must relate the angles within each pair of triangles incident on a common edge. The condition for each such edge is

$$\begin{aligned}&\frac{\cos(\alpha_{I_1(l)}^{J_1(l)}) + \cos(\alpha_{I_1(l)}^{J_1(l)-1})\cos(\alpha_{I_1(l)}^{J_1(l)+1})}{\sin(\alpha_{I_1(l)}^{J_1(l)-1})\sin(\alpha_{I_1(l)}^{J_1(l)+1})} \\ &= \frac{\cos(\alpha_{I_2(l)}^{J_2(l)}) + \cos(\alpha_{I_2(l)}^{J_2(l)-1})\cos(\alpha_{I_2(l)}^{J_2(l)+1})}{\sin(\alpha_{I_2(l)}^{J_2(l)-1})\sin(\alpha_{I_2(l)}^{J_2(l)+1})} \quad l = 1, \ldots, m.\end{aligned} \tag{8}$$

These conditions are necessary and sufficient to guarantee a spherical triangulation. The necessity is obvious following the proof in [9]. To prove the sufficiency, we first prove that the triangles incident on every vertex locally form a valid "ring" of triangles, namely, all triangles have the same orientation and the edges coincide in turn, as in Fig. 2a.

The proof proceeds in the following stages:

1. Any triplet of angles, and an excess which satisfy (3)–(6), form a spherical triangle.
2. Any triplet of angles, an excess, and arc length which satisfy (3)–(8) form a spherical triangle.
3. Any pair of triangles sharing a common edge lie on the sphere, i.e. they agree on the length of the common edge. This follows from (8).
4. The spherical triangles whose angles and excesses satisfy (3)–(8) and have one common vertex form a valid ring of spherical triangles around that vertex. The sine rule condition necessary in [9] is replaced by (8), since if each pair of triangles sharing an edge agrees on its length, then so does an entire ring.

It is then straightforward to see that if the embedding is locally valid everywhere, then it must also be globally valid.

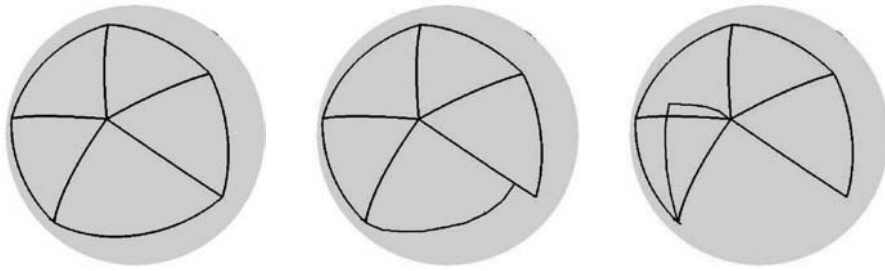

Fig. 2. Local embeddings. (**a**) Valid. (**b**), (**c**) Invalid

Clearly, many different spherical triangulations exist for a given connectivity graph. This can also be seen from the conditions (3)–(8), since the fact that there are $2n - 6$ more variables than equalities implies that there are probably some degrees of freedom in the solution. Hence we can attempt to "mould" the triangulation by specifying target values for the spherical triangle angles and areas. Given these target values, the problem becomes a constrained minimization problem, where we minimize the least-squares distance of the solution values from their target values (β_i^j and e_i'):

$$F(\alpha) = \sum_{i=1}^{t} \sum_{j=0}^{2} (\alpha_i^j - \beta_i^j)^2 + \sum_{i=1}^{t} (e_i - e_i')^2. \tag{9}$$

This cost function allows us to control the shape of the parameterization by optimizing spherical angle and/or area values. For example, if the input connectivity originates in a 3D model, using the (normalized) 3D areas of the model triangles as targets will aim at area preserving parameterization. Similarly, using the original 3D angles as targets will aim at conformal mapping. None of the methods reviewed in the introduction permits such control of parameterization properties. By solving the constrained minimization problem defined above we provide a parameterization method which is both robust and provides user control of the mapping properties.

4. Solving the System

To minimize (9) under constraints (3)–(8), an optimizer for non-linear constrained systems is required. We used the *fmincon* function of MATLAB, which converts the constrained minimization problem into unconstrained minimization using Lagrange multipliers. The unconstrained problem is then solved using a quasi-Newton method combined with line-search. This method requires the gradients of both the cost function and the constraints. Both were computed analytically and supplied to the solver. To speed up the solution we supply an initial guess close to the target, by solving the minimization subject only to the linear constraints (3)–(7). This is a standard linear least squares problem. To further accelerate the solution, we avoid introducing the inequality constraints (3)–(5) unless necessary, since they significantly slow down the solver. Since the solution space is not empty and the initial guess is all positive, those constraints are unlikely to be actually needed. Hence we solve the system first with only equality constraints. If the result contains negative angles or excesses, we introduce the inequality constraints and repeat the solution.

5. Embedding on the Sphere

Once the spherical angles have been determined, it is possible to embed the triangle vertices on the sphere by a recursive preorder traversal of the triangulation connectivity structure as follows. The spherical cosine rule (2) defines the

lengths of the edges of each spherical triangle as a function of the angles. Starting from an arbitrary triangle, we compute the length a of its first edge from the angles

$$a = \arccos\left(\frac{\cos A + \cos B \cos C}{\sin B \sin C}\right). \tag{10}$$

We then embed its first vertex at an arbitrary location v_1 and embed the second vertex at a location v_2 at spherical distance a on the sphere (at an arbitrary orientation) to form its first edge.

The cosine rule (2) is applied again to similarly obtain the other two spherical edge lengths:

$$\begin{aligned} b &= \arccos\left(\frac{\cos B + \cos A \cos C}{\sin A \sin C}\right), \\ c &= \arccos\left(\frac{\cos C + \cos A \cos B}{\sin A \sin B}\right). \end{aligned} \tag{11}$$

Using the planar sine rule, the Euclidean distances between the new vertex v_3 and the two already positioned are:

$$\begin{aligned} l_b &= 2\sin(b/2), \\ l_c &= 2\sin(c/2). \end{aligned}$$

The third vertex coordinates are found by solving for the intersection of three spheres, one of which is the unit sphere, the other two are centered at v_1 and v_2 with radii l_b and l_c, respectively. This is a system of three quadratic equations, which may be solved analytically giving two solutions. The correct of the two is chosen to be that closest to the fourth vertex of a parallelogram whose other three vertices form the base triangle.

This forms the second and third edges of the first triangle. Once the triangle is embedded on the sphere, recursively embed the two new triangles incident on the second and third edges (starting by applying (11) to the spherical angles of the new triangle). The embedding is complete once all triangles have been traversed, and is unique up to global rotation on the sphere.

Similarly to the 2D case [9] the unfolding can accumulate a numerical error, which increases with the number of triangles to unfold. We are examining ways to reduce this error. One way could be to generate more than one unfolding sequence and then use an averaging procedure to moderate the error.

6. Some Examples

Figure 3 shows some spherical embeddings of a cylindrical mesh (Fig. 3a), as obtained from our algorithm, using a variety of target spherical angles and areas.

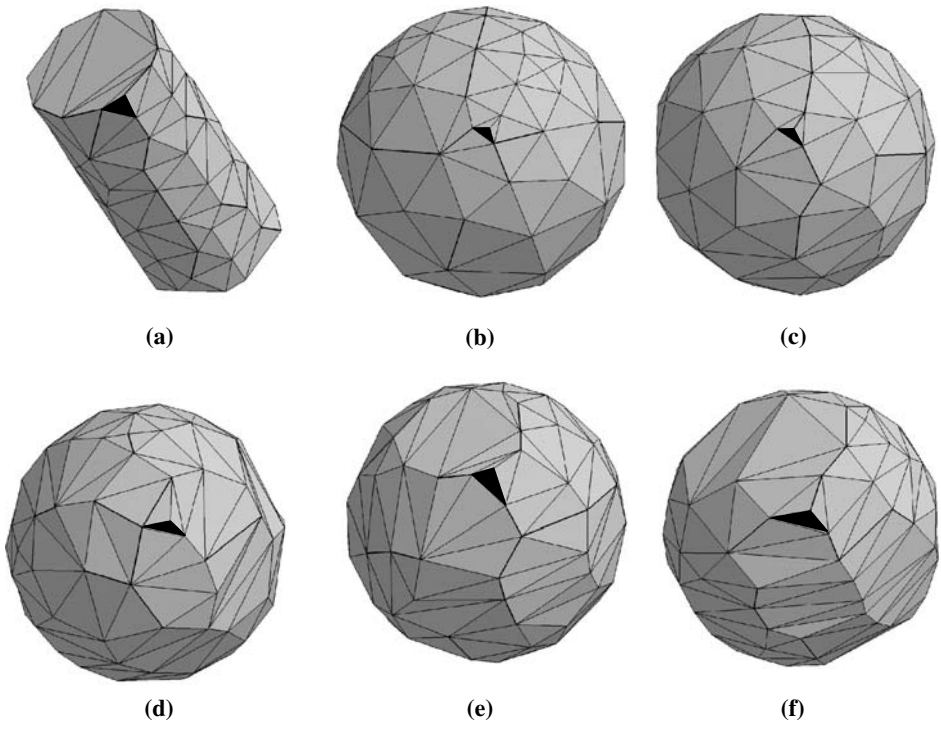

Fig. 3. Spherical embeddings. (**a**) Cylinder mesh. (**b**) Embedding of Alexa [1]. (**c**) Our embedding aiming for equal angles. (**d**) Our embedding aiming for equal areas. (**e**) Our embedding aiming for the original angles. (**f**) Our embedding aiming for the original areas. The black triangles correspond in each of the meshes

Figure 3b is the output from Alexa's algorithm [1], which aims for equal angles. When our algorithm is told to aim for this effect too, we get a very similar result (Fig. 3c). When equal areas are required, skinny triangles start to appear (Fig. 3d). When the embedding is required to reflect the geometric properties of the original cylindrical mesh, the similarities are evident. In Fig. 3e, where the original angles are targeted, the black triangle stays right-angled. In Fig 2f, where the original areas are targeted, the areas of the triangles corresponding to those on the two bases are much larger than the others.

7. Discussion and Conclusion

We have formulated a set of necessary and sufficient conditions for the spherical angles of a triangulation to form a valid spherical triangulation. We use this to generate spherical embeddings with desirable properties by the use of an appropriate cost function.

The conditions we formulated are not minimal, in the sense that there exists some redundancy in them. Some of them may be eliminated without changing the set of solutions. We are not yet sure how to do this.

The numerical procedure we use today to solve the system is quite slow and not practical for meshes containing more than a few hundred vertices. The main challenge we currently face is finding an efficient numerical solution for mid-scale to large meshes. We expect a combination of better tuned numerical techniques combined with the use of mesh hierarchies to significantly accelerate the solution.

References

[1] Alexa, M.: Merging polyhedral shapes with scattered features. The Visual Computer *16*, 26–37 (2000).
[2] Floater, M. S.: Parameterization and smooth approximation of surface triangulation. Computer Aided Geometric Design *14*, 231–250 (1997).
[3] Haker, S., Angenent, S., Tannenbaum, A., Kikinis, R., Sapiro, G., Halle, M.: Conformal surface parameterization for texture mapping. IEEE Transactions on Visualization and Computer Graphics *6*, 181–189 (2000).
[4] Grunbaum, B.: Convex polytopes. Interscience Publishers 1967.
[5] Kobbelt, L.P., Vorsatz, J., Labisk, U., Seidel, H.-P.: A shrink-wrapping approach to remeshing polygonal surfaces. Proceedings of Eurographics 1999.
[6] Levy, B., Petitjean, S.: Least squares conformal maps for automatic texture atlas generation. Proceedings of ACM SIGGRAPH 2002.
[7] Shapiro, A., Tal, A.: Polygon realization for shape transformation. The Visual Computer *14*, 429–444, (1998).
[8] Sheffer, A.: Spanning tree seams for reducing parameterization distortion of triangulated surfaces. Proceedings of Shape Modelling International, pp 61–66, 2002.
[9] Sheffer, A., de Sturler, E.: Parameterization of faceted surfaces for meshing using angle based flattening. Engineering with Computers *17*, 326–337 (2001).
[10] Sheffer, A, Gotsman, C., Dyn, N.: Robust spherical parameterization of triangular meshes. 4th Israel-Korea Bi-National Conference on Geometric Modeling and Computer Graphics, pp. 94–99, 2003.

C. Gotsman
Center for Graphics and
Geometric Computing
Department of Computer Science
Technion-Israel Institute of Technology
Haifa 32000
Israel
e-mail: gotsman@cs.technion.ac.il

N. Dyn
School of Mathematical Sciences
Faculty of Exact Sciences
Tel Aviv University
P.O. Box 39040
Tel Aviv 69978
Israel
e-mail: niradyn@math.tau.ac.il

A. Sheffer
Department of Computer Science
University of British Columbia
201-2366 Main Mall
Vancouver, V6T 1Z4
Canada
e-mail: sheffa@cs.ubc.ca