Smoothing an Overlay Grid to Minimize Linear Distortion in Texture Mapping

ALLA SHEFFER
Technion
and
ERIC DE STURLER
Department of Computer Science, University of Illinois at Urbana-Champaign

Texture is an essential component of computer generated models. For a texture mapping procedure to be effective it has to generate continuous textures and cause only small mapping distortion. The *Angle Based Flattening (ABF)* parameterization method is guaranteed to provide a continuous (no foldovers) mapping. It also minimizes the angular distortion of the parameterization, including locating the optimal planar domain boundary. However, since it concentrates on minimizing the angular distortion of the mapping, it can introduce relatively large linear distortion.

In this paper we introduce a new procedure for reducing length distortion of an existing parameterization and apply it to ABF results. The linear distortion reduction is added as a second step in a texture mapping computation. The new method is based on computing a mapping from the plane to itself which has length distortion very similar to that of the ABF parameterization. By applying the inverse mapping to the result of the initial parameterization, we obtain a new parameterization with low length distortion. We notice that the procedure for computing the inverse mapping can be applied to any other (convenient) mapping from the three-dimensional surface to the plane in order to improve it.

The mapping in the plane is computed by applying weighted Laplacian smoothing to a Cartesian grid covering the planar domain of the initial mapping. Both the mapping and its inverse are provably continuous. Since angle preserving (conformal) mappings, such as ABF, locally preserve distances as well, the planar mapping has small local deformation. As a result, the inverse mapping does not significantly increase the angular distortion.

The combined texture mapping procedure provides a mapping with low distance and angular distortion, which is guaranteed to be continuous.

Categories and Subject Descriptors: I.3.7 [**Computer Graphics**]: Three-Dimensional Graphics and Realism—*Color, shading, shadowing, and texture*; I.3.5 [**Computer Graphics**]: Computational Geometry and Object Modeling; G.1.6 [**Numerical Analysis**]: Optimization—*Constrained optimization*; J.6 [**Computer Aided-Engineering**]

General Terms: Algorithms

Additional Key Words and Phrases: texture mapping, parameterization, triangulation, smoothing.

1. INTRODUCTION

Texture is an essential component of computer generated images. To create a texture on a three-dimensional surface model, a texture mapping procedure is used. Texture mapping provides a projection

The work reported here was supported, in part, by the Center for Process Simulation and Design (NSF DMS 98-73945), the Center for Simulation of Advanced Rockets (DOE LLNL B341494), and Sandia National Laboratories (SNL 16806).

Authors' addresses: A. Sheffer, Computer Science Department, Technion, Technion City, Haifa, 32000, Israel; email: sheffa@cs.technion.ac.il; E. de Sturler, Department of Computer Science, University of Illinois at Urbana Champaign, Urbana, Il, 61801; e-mail: sturler@cs.uiuc.edu.

function from a three-dimensional surface to a two-dimensional texture pattern. Such a mapping corresponds to a two-dimensional parameterization of the three-dimensional surface. To minimize the distortion of the texture, the parameterization function has to preserve the surface metric structures as much as possible.

Parameterization of three-dimensional surfaces has many other applications as well, including finite-element surface meshing [Sheffer and de Sturler 2000a; Marcum and Gaiter 1999], surface reconstruction [Floater 1997; Hormann and Greiner 2000], multiresolution analysis [Eck et al. 1995], formation of ship hulls, generation of clothing patterns [McCartney et al. 1999], and metal forming.

When the surface is represented using an analytic description, this description can often be used to provide the parameterization. However, many computer graphics models are represented by a triangular tessellation and the analytic representation of the surface is often not available. A triangulated mesh is the "natural" description of surfaces constructed from scattered data such as input from laser range scanners or sampling on a regular three-dimensional grid.

An algorithm for two-dimensional parameterization or *flattening* of tessellated surfaces first constructs a two-dimensional mesh with a connectivity equivalent to that of the three-dimensional surface. Then, a parametric function is defined between the two-dimensional mesh facets and their three-dimensional counterparts.

Several approaches for parameterization of tessellated surfaces have been suggested. In McCartney et al. [1999] the authors suggest a heuristic approach for triangulation flattening. The method is based on optimal local positioning of projected nodes, based on a sequential addition of the nodes. Similar ideas were suggested in Samek et al. [1986] and Bennis et al. [1991]. The methods are efficient and produce good results for nearly planar surfaces. However, they do not guarantee the preservation of the metric structures in the two-dimensional mesh or mapping continuity. Discontinuities in the mapping arise when the mesh folds, that is mesh faces get inverted.

Several works, for example Azariadis and Aspragathos [2001], limit the parameterization problem to the case of nearly-developable surfaces, where they are able to provide optimal results, without risking generation of discontinuities.

Eck et al. [1995] suggest the use of harmonic maps [Eells and Sampson 1964; Eells and Lemaire 1988] to generate the two-dimensional projection of the three-dimensional mesh. The algorithm produces approximations of good quality, and provides an accurate mapping function. Nonetheless, it has a major disadvantage in that it requires the boundary of the two-dimensional mesh to be predefined and convex. Another drawback is that the method does not guarantee mapping continuity (i.e. it may generate inverted elements).

Floater [1997] describes a parameterization method which computes the positions of the nodes in the flat mesh using the solution of a linear system based on convex combinations. Floater generalized earlier work by Tutte [1960, 1963] to derive a method that sets nodes as convex combinations of their neighbors while aiming to preserve local shape characteristics. The method also guarantees that the resulting mapping is continuous. However, similarly to the algorithm in Eck et al. [1995], the method requires the boundary of the two-dimensional mesh domain to be predefined and convex.

Hormann and Greiner [2000] use Floater's algorithm as a starting point for a highly non-linear local optimization algorithm which computes the positions for both interior and boundary nodes based on local shape preservation criteria. The method is very promising, but it is not clear whether the procedure is guaranteed to converge to a valid solution.

Marcum [Marcum and Gaiter 1999] introduces the use of finite-element techniques to compute the locations of the flat mesh nodes. The method computes the boundary as part of the solution, using an iterative procedure that alternatingly computes the boundary while keeping the interior fixed and vice versa. However, no continuity guarantees are given for this method.

A lot of research on computing parameterizations of tessellated surfaces has been done in the context of computer graphics, since parameterization is required to generate non-distorted texture mappings [Foley et al. 1992]. Maillot et al. [1993] presented one of the first works in the context of texture mapping that formally defined a distortion functional and proceeded to minimize it. Zigelman et al. [2001] provide a method for flattening surfaces by using multi-dimensional scaling. It computes the two-dimensional domain boundaries as part of the solution. The method does not prevent foldovers. Levy and Mallet [1998] suggest a method that provides user control on the spread of the distortion across the flattened mesh. The method uses some of the formulations introduced in Floater [1997], and as a result has similar limitations. In a more recent work, Levy [2001], suggested a procedure that allows the user to constrain a number of point locations within the parameterization instead of defining the boundary. The method provides impressive results, but can sometimes generate foldovers.

Several recent works [Sander et al. 2001; Praun et al. 2000] subdivide the three-dimensional surface into patches for which low distortion mapping is computed. This leads to generation of texture seams, which the authors try to hide using various techniques. Sorkine et al. [2002] suggest a combination of subdivision with local optimization, which guarantees a distortion bound, but can generate long texture seams.

Sheffer and de Sturler [2000a] introduced a quasi-conformal mapping method. The method is based on the observation that a triangulated planar mesh is fully defined by the mesh angles up to global scaling, rotation, and translation. The authors formulate the parameterization problem in terms of the flat mesh angles and solve it in the angle space. They use a set of constraints on angles which define a valid (continuous) planar mesh. They proceed to minimize the angular distortion of the parameterization, subject to the defined constraints. Since the problem is formulated solely in terms of the planar angles, the new method was named *Angle Based Flattening* (*ABF*). The ABF algorithm does not require the two-dimensional boundary to be predefined, does not place any restrictions on the shape of the boundary or the surface curvature, it guarantees a continuous mapping, and the numerical solution is guaranteed to converge; see Sheffer and de Sturler [2000b, 2000a]. Another quasi-conformal mapping method was recently suggested by Hurdal [Hurdal et al. 1998]. It has many of the properties of the ABF, in terms of continuity and boundary definition. The drawback of both methods is that they minimize angular distortion, disregarding the linear distortion that may occur.

Contribution

In this paper we introduce a method for computing a surface parameterization that tries to minimize both angular distortion and linear distortion. This method consists of two steps. The first step computes a quasi-conformal mapping, using the ABF procedure, and the second step adapts this mapping to minimize linear distortion.

This adaptation is implemented as follows. We superimpose a uniform Cartesian grid on the ABF-generated flat triangulation. We use the associated parameterization to compute the lengths of the uniform grid edges mapped to the three-dimensional surface. Then we iteratively relax vertices in the Cartesian grid based on their associated edge lengths on the three-dimensional surface. The relaxation reduces the difference in (3D) length of the edges attached to the relaxed vertex. Note that all this is carried out in the plane.

The combined mapping has the following advantages over many existing mapping methods: (1) The method provably provides a continuous parameterization for any surface that can be mapped to the plane. (2) The method minimizes the distortion of both shape and length caused by the parameterization. (3) The solution process computes the optimal planar domain boundary in terms of minimal distortion, and does not place any restrictions on the boundary shape, for example convexity. (4) The algorithm is efficient and takes a matter of seconds to generate the mapping.

Finally, we would like to point out that the overlay grid algorithm for minimizing linear distortion can be used in conjunction with *any* other algorithm that computes a (preliminary) surface parameterization. It is especially suited as a post-processing step to results of other quasi-conformal methods such as harmonic mapping [Eck et al. 1995] and others [Hurdal et al. 1998; Hormann and Greiner 2000]. In this paper we provide several examples of texture mapping using the developed algorithm and compare it with two commonly used methods. We also include an example of using it as a post-processing step for harmonic mapping (Figure 6(f)).

2. ALGORITHM OVERVIEW

In order to preserve the surface metric structures in parameterization, both angle and length distortion have to be minimized. For a general surface, there is no mapping to the plane which fully preserves lengths or areas [Ahlfors and Sario 1960]. Neither is there a mapping to the plane for a faceted surface with nonzero curvature that preserves angles. Hence, we suggest a method that strives to minimize both types of distortion. The first step of the combined algorithm we suggest minimizes the angular distortion and the second minimizes the linear distortion.

Based on Riemann's theorem [Ahlfors and Sario 1960] for a smooth surface, there exists a mapping to the plane which is conformal, that is, it preserves the surface angles. Such a mapping is also guaranteed to locally preserve the shape and the metric structures. In general, however, a conformal mapping does not exist for a faceted (non smooth) surface. The first part of our algorithm computes a mapping from the three-dimensional surface to the plane that approximates a conformal mapping (Section 3). Since the preservation of lengths is of major importance to texture mapping, a second mapping that minimizes length distortion has to be applied. Note that the two metrics do not usually have common minima, hence reducing the length distortion may increase the angular one. However, from our experience, using the technique described below, the improvement in length preservation does not lead to significant deterioration in angle preservation. This is due to the fact that angle preservation preserves length ratios locally.

After providing a flat parameterization using the ABF algorithm we compute a mapping from a rectangular region of the plane, containing the flat triangulation to itself. The mapping is designed to mimic the distortion of the distances of the ABF parameterization. The mapping computation is based on applying a mesh smoothing procedure to a Cartesian grid. The smoothing uses a sizing function which is based on the ratios between the lengths of the edges in the three-dimensional surface and their counterparts in the flat surface. It is described in detail in Section 4. The mapping is provably one-to-one, hence an inverse mapping exists. The inverse mapping is defined by mapping the smoothed Cartesian grid to the unsmoothed (regular) one. By applying the inverse mapping to the result of the ABF parameterization we generate a combined, continuous mapping that has small metric distortion.

The next section briefly reviews the ABF parameterization method. The second mapping stage is described in detail in Section 4. We use the following notations throughout the paper.

The index $i,\ i = 1 \dots P$ always indicates faces, the index $j,\ j = 1, 2, 3$ indicates angles inside a face, and the index $k,\ k = 1 \dots M$ indicates nodes. We use $\alpha_i^j, i = 1 \dots P, j = 1, 2, 3$ to denote the flat mesh angles. The vector of all angles is denoted by α. We use $\beta_i^j, i = 1 \dots P, j = 1, 2, 3$ to denote the corresponding original mesh angles.

3. ANGLE BASED FLATTENING

This section briefly outlines the angle based flattening algorithm that provides the first component of the texture mapping function. For a more complete description we refer the reader to Sheffer and de Sturler [2000a, 2000b].

As mentioned above, the ABF algorithm finds the planar parameterization by solving a constrained minimization problem defined in terms of the angles in the flat mesh. The minimized functional aims at generating a quasi-conformal mesh and is based on minimizing the sum of relative square distances between the angles in the original and flat meshes, that is the vectors β and α, respectively. The constraints restrict the vector α to define a valid planar mesh. We say the flat mesh is valid if its associated graph is equivalent to that of the three-dimensional surface mesh, if it has no interior inconsistencies (see below), and if it has no intersecting boundary edges. We say the resulting flat mesh has an interior inconsistency if (a) the order of the nodes in any face is reversed with respect to the three-dimensional surface mesh, which means the face has changed orientation (folded-over) with respect to the mesh, or (b) the angles in any face do not sum to π, or (c) two neighboring faces disagree on the placement of their shared edge. The latter inconsistency can arise in two different ways. First, (c1) the angles in each face may require different directions for the shared edge; second, (c2) the lengths of their non-shared edges (together with the given angles) for each face may require different lengths for the shared edge. We note that either problem can occur for only one edge attached to an interior node. For more details we refer again to Sheffer and de Sturler [2000a]. To guarantee the interior validity we add four sets of constraints to our optimization problem and to avoid boundary intersections we use a postprocessing step (rarely needed), which is discussed at the end of this section.

We convert the constrained minimization problem into an unconstrained problem using a Lagrange multiplier formulation. We use Newton's method to find the minimum. In almost all cases Newton's method converges in less then five iterations. The "robust Newton methods" (e.g., trust region methods) are guaranteed to converge for minimization problems, such as the one we have. The sparse linear systems of equations that arise in Newton's method are solved using a sparse direct solver from the SuperLU package [Demmel et al. 1999a, 1999b].

Finally, after the algorithm computes the angles of the flat mesh, we compute the locations of the flat mesh nodes, after 'arbitrarily' choosing the placement of a first edge. The node placement together with a local bijection per face (between corresponding faces in the surface and flat mesh) defined by equal barycentric coordinates yields the parameterization function T^1 from the three-dimensional surface to the flat domain. Figure 1 shows two examples of applying the flattening procedure to three-dimensional surfaces. The generated textures are shown in Figure 4.

The constraints above guarantee the continuity of the mapping T^1 (no foldovers). However, they do not prevent the flat surface from having self-intersections at the boundary. Such intersections can occur since the boundary is found as part of the solution and not defined beforehand. Boundary intersections make the mapping from the surface in $\mathbf{R}^3$ to $\mathbf{R}^2$ a many to one mapping instead of one-to-one.

For applications where the mapping has to be one to one, such as for mapping of a larger image or pattern or for paint applications, our method removes intersections in a post-processing step [Sheffer and de Sturler 2000a]. Alternatively the surface can be subdivided into two or more patches, separating the overlaping parts [Levy 2002].

For texture mapping with a small regular pattern only the $\mathbf{R}^3$ to $\mathbf{R}^2$ mapping is used, providing the texture value for each point on the three-dimensional surface. The mapping in case of boundary overlaps remains continuous and smooth and therefore it preserves the texture patterns (Figure 2). Hence, for this type of application the intersections can be ignored.

4. LENGTH PRESERVING MAPPING IN $\mathbf{R}^2$

After the flat triangulation has been generated, the mapping T^1 from the three-dimensional surface to $\mathbf{R}^2$ can be computed. For every mesh node its image under T^1 is given by the coordinates of the corresponding node in the flat mesh. Each triangle in the three-dimensional surface is mapped to the corresponding planar triangle using barycentric coordinates. This means that for every point p on

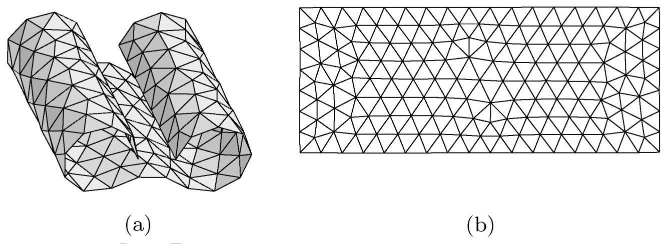
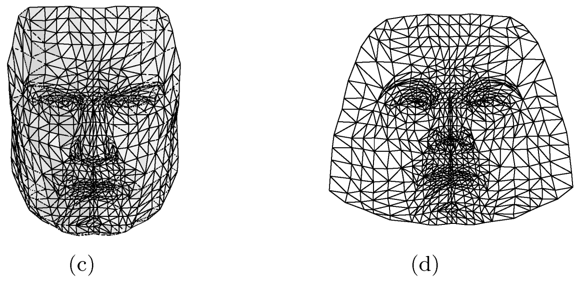
Fig. 1. Flat triangulation examples. (a)-(b) Folded plane (nearly developable surface). (a) Original model. (b) Flat triangulation. (c)-(d) Human face. (c) Original model. (d) Flat triangulation.

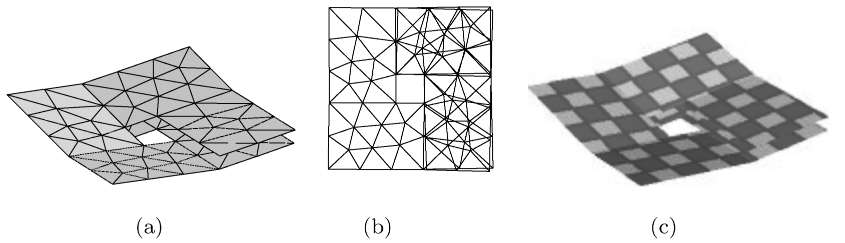
Fig. 2. Texture mapping for a spiral surface with non unique two-dimensional mapping. (a) Original surface (110 elements). (b) Flat parameterization containing boundary intersections. (c) Final texture.

the interior of a triangle its image under T^1 is the point with the same barycentric coordinates in the planar triangle which p had in the corresponding surface triangle. This mapping minimizes the angular distortion, but it may not preserve lengths or areas accurately. To improve length (distance) preservation we compute an additional mapping T^2 from $\mathbf{R}^2$ to $\mathbf{R}^2$, as explained below.

4.1 Sizing Function

To compute the mapping T^2, we first compute a *sizing function* representing the (approximately isotropic) distortion of length, that is, linear distortion, at any point on the flat mesh. The sizing function S is computed as follows.

—For each edge $e = (n_k, n_l)$, compute the distortion ratio as $S_{k,l} = \frac{\|e_{2D}\|}{\|e_{3D}\|}$, the ratio of the edge length in the flat mesh to the edge length in the original surface.

—For each node $n_k,\ k = 1 \ldots M$, compute the distortion ratio at the node, $\tilde{S}_k$, as the average of the distortions computed over the edges containing n_k.

—For each point p we locate the mesh triangle $f_i = (n_a, n_b, n_c)$ to which it belongs. We compute the linear distortion $S(p)$ using the barycentric coordinates of p in f_i. For $p = n_a u + n_b v + n_c w$, where $u + v + w = 1$ we set $S(p) = \tilde{S}_a u + \tilde{S}_b v + \tilde{S}_c w$.

The function S approximates the linear distortion of T^1 for each point (target) on the planar domain. Since T^1 is quasi-conformal, edge length ratio locally remains almost unchanged. Hence, S has very small local variation (derivative).

4.2 Mapping

A texture mapping function T defines a mapping from the three-dimensional surface to a rectangular (u, v) domain in $\mathbf{R}^2$. If T^1 is used as this function, then the linear distortion of the mapping will be (approximately) given by S. To reduce the distortion, an additional mapping T^2 from $\mathbf{R}^2$ to $\mathbf{R}^2$ needs to be introduced with linear distortion (approximately) given by S^{-1}. The combined mapping will then have the desired property of being nearly free from linear distortion.

To compute such T^2 we consider the inverse problem of finding $T^{2^{-1}}$ and inverting it. To achieve this, the following procedure has been developed. Let F be the flat triangulation.

1. Compute the bounding box of F.
2. Generate a uniform Cartesian grid G_1 for a rectangular region containing the bounding box. The grid region has to be sufficiently large to allow the necessary freedom for the interior nodes during the smoothing procedure in stage 4. The choice of region size and grid density are discussed in Section 4.4. The Cartesian grid is constructed by using a triangular mesh, where each Cartesian square is represented by two right-angle triangles sharing the hypotenuse. The triangular representation avoids the problems of mapping between possibly non-convex quadrilaterals.
3. Generate a copy G_2 of the grid G_1, and embed F in G_2.
4. Adapt G_2 by grid smoothing using the sizing function S defined on F as explained below (Section 4.3). (The mapping from G_1 to G_2 gives us the desired $T^{2^{-1}}$, i.e. a mapping with linear distortion S.)
5. Define $T^2 : \mathbf{R}^2 \to \mathbf{R}^2$ by mapping each point on G_2 to the point on the corresponding triangle in G_1 with the same barycentric coordinates.

The steps of the procedure are shown in Figure 3. The flat triangulation, Figure 3-(b), is embedded in the smoothed grid, Figure 3-(e). We define the texture mapping function T as:

$$T \equiv T^2 \circ T^1 : \mathbf{R}^3 \to \mathbf{R}^2. \tag{1}$$

Due to the use of the additional sizing-based mapping T^2, the final texture mapping function preserves lengths fairly accurately.

4.3 Smoothing

Mesh smoothing is a procedure that changes mesh specifications (size/element shape) by changing the locations of the mesh nodes, without changing the mesh connectivity [Owen 1998; Mallet 1989]. In our application we use the smoothing procedure to modify the mesh sizing. We start with an initial uniform grid and modify it to conform to the sizing function defined above. There are several methods [Canann et al. 1998] for smoothing two-dimensional meshes. We use Laplacian smoothing [Field 1988] for its simplicity.

The procedure to modify G_2 to conform to the sizing function S is as follows. *Note that the smoothing algorithm uses only the Cartesian grid edges and ignores the mesh "diagonals"* .

1. For each edge $e = (n_a, n_b) \in G_2$ compute the desired edge length (up to a global scaling factor):
$$l(e) = \frac{S(n_a) + S(n_b)}{2} \tag{2}$$
(outside of F, S is set to 1).

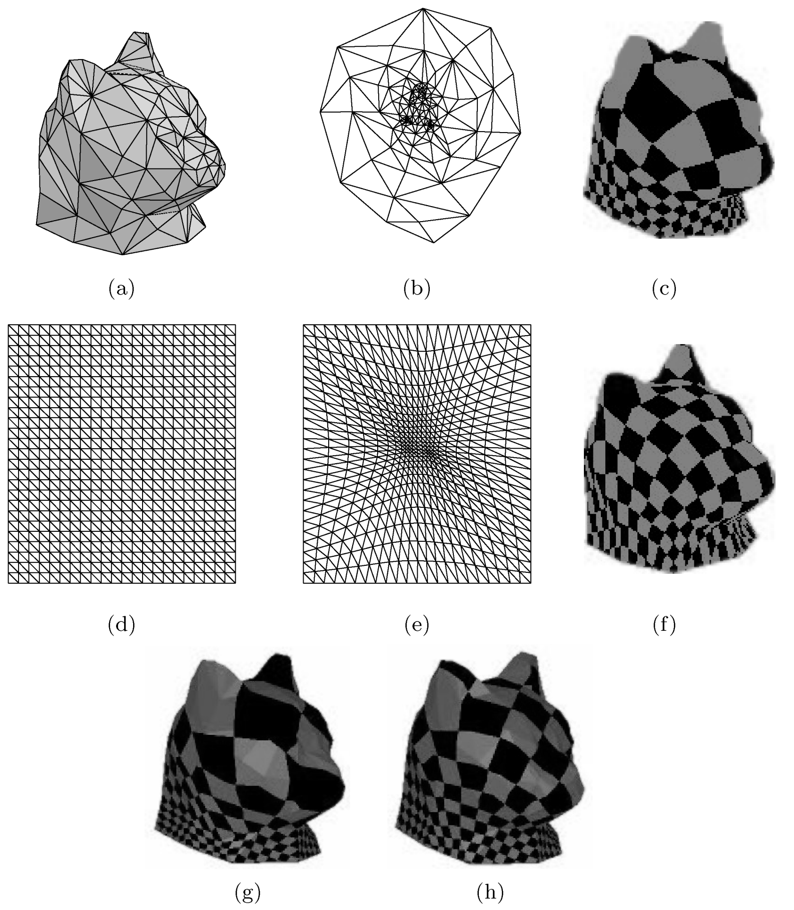

Fig. 3. Texture mapping for a cat head model. (a) The head (256 elements). (b) Flat parameterization. (c) Texture using only T^1 (without sizing function). (d) Uniform grid G_1 of the bounding box. (e) Smoothed grid G_2. (f) Final texture using T. (g), (h) Texture generated with convex combinations and harmonic mapping, respectively. Note that harmonic mapping is very close to ABF only mapping (c).

2. Apply Laplacian smoothing to adjust the edge lengths to the sizing function:
   2.1 For each interior node $N \in G_2$ in turn,

$$\text{compute } \bar{N} = \frac{\sum_{e=(N,N')} \frac{1}{l(e)} N'}{\sum_{e=(N,N')} \frac{1}{l(e)}},$$

$$\text{compute } d = \max(d, \| \bar{N} - N \|),$$

$$\text{and set } N = \bar{N}.$$

   2.2 Repeat 2.1 while $d > \epsilon$.
3. Recompute the desired lengths and repeat the smoothing (goto 1.), until the lengths $l(e)$ or the node locations no longer change.

As discussed in the next section, the procedure converges to a mesh which conforms to the sizing function. For the texture mapping we use the mapping function from the smoothed grid G_2 to the original uniform, Cartesian grid G_1.

4.4 Algorithm Accuracy and Complexity

The accuracy of the smoothing procedure, namely how closely it satisfies the sizing function, depends on two factors. The first is the size of the grid boundary. Boundary nodes remain static throughout the procedure. Therefore, the mesh near the boundary might not have enough degrees of freedom to satisfy the sizing. To generate the desired inverse mapping, only the part of the grid within the ABF mapping domain D is of interest. Hence, to satisfy the sizing within the domain D we extend the grid boundary beyond the bounding box of D. By having several mesh layers outside, we obtain sufficient degrees of freedom for the mesh on the interior. Clearly, increasing the boundary size means increasing the number of elements and nodes in the grid. This increases the computation cost, both in terms of memory and processing time. We found that by increasing the bounding box by a factor of 1.2 or 1.3, we were able to obtain a sufficiently accurate mapping at a reasonable cost.

Another, more important, factor linking accuracy and complexity is the level of grid refinement. When computing the weights for smoothing, the method samples the sizing at the end nodes of each edge and uses the average as edge weight. To compute the weight accurately, taking into account the sizing along the edge, we should optimally use the integral over the sizing function S along the edge. However, this is not practical; hence the use of the average to approximate the integral by the trapezoid rule. Since all the edges in the regular grid are of equal size, we can take the edge length in it to be any constant (here one). This is true throughout the algorithm iterations, as the length of the projected grid edges on the surface is expected to be constant. The accuracy of the sampling, and hence the smoothing, can be achieved either by sampling the size along the edge (not just at end points), or using shorter grid edges. We choose the second option, linking accuracy to grid refinement level. This choice is also motivated by the desire to capture the sizing function across the region, which might not happen when sampling only along (possibly long) edges.

Since S is piecewise linear, sampling it, that is having grid points at shorter distances than the shortest edge in the flat mesh, will not increase sampling accuracy. As a consequence, the lower bound for the grid refinement is the shortest edge length in the flattened mesh. However, this leads to a very fine mesh and makes the smoothing process very expensive. For the dome example (Figure 5) the process reached the bound of 200 iterations after 11 *minutes* and gave 0.077 and 0.186 as angular and linear distortions. The sizing we used was based on the median edge length in the flat mesh. For the same model it gave just slightly worse linear and angular distortion and took 1 second to converge (Table I). The motivation in using the median rather than the average is the large deviation in edge lengths in the flat meshes, as a result of regions which were "stretched out" versus ones which were "squeezed," for example Figure 3. Therefore, the median tends to be significantly smaller than the average. The coarser grid has a drastically smaller computational cost but can potentially "miss" very small details in the planar mesh, generated in areas where the three-dimensional mesh has very high curvature.

Clearly, using a uniform grid for the grid (mesh) smoothing step creates a problem. In order to capture small details we need a fine grid that leads to wasted computational effort in regions were such high resolution is not needed. This is all the more so because the convergence of point smoothing deteriorates with finer grids. We can resolve both problems by combining two techniques. The first is to replace the uniform grid by an adaptively refined grid, for example, based on quad trees. This allows us to locally adapt the grid resolution as needed. The second is to do multilevel smoothing rather than just point smoothing. Multilevel algorithms generally lead to significant convergence improvements [Brandt 1977; Trottenberg et al. 2001]; in many cases the total amount of work is proportional to the number of nodes in the (finest) grid. Clearly these two approaches will go together very well. We plan to discuss this new multilevel adaptive grid smoothing strategy in a follow-up paper. We will not discuss it further here.

Finally, the grid smoothing algorithm aimed at reducing linear distortion will generally increase the angular distortion. Although in general this seems to create few problems, it would be good to be able to make an explicit trade-off between the two types of distortion. Since grid smoothing makes small updates to each point, bounded by the quadrilateral defined by its neighbors, we can currently do this by simply stopping the smoothing when angular distortion reaches some tolerance. However, we have not experimented with this yet as the deterioration of angular distortion seems to be negligible. An alternative approach would be to apply some weighting between the original flat mesh and the smoothed mesh after each sweep of smoothing (one update for each point).

5. ALGORITHM VALIDITY

To show that the presented algorithm is valid, we need to prove that both parts of the algorithm provide the valid parameterization functions T^1 and T^2. Namely, both algorithms find a one-to-one mapping between the source and target for valid inputs. In this context, a valid input is a mesh surface with genus zero and a single outside boundary loop. A closed surface or surface of higher genus cannot by definition be parameterized in the plane. The ABF algorithm, as well as most other parameterization methods described in Section 1., cannot handle inputs with multiple loops, even if they can be mapped to the plane. The Cartesian grid mapping defining T^2 can handle such surfaces with no special treatment.

In Sheffer and de Sturler [2000a] we prove the correctness of the ABF procedure, which provides the first step of the texture mapping. It shows that given valid input, the algorithm is guaranteed to converge to a valid flat triangulation that minimizes angular distortion. Hence all we need to show is that the Cartesian grid mapping provides a one-to-one mapping and the solution method (smoothing procedure) is guaranteed to converge.

The weighted Laplacian smoothing procedure is applied to a Cartesian grid of a rectangular domain. The fact that such smoothing over a convex domain is guaranteed to converge to a valid (no-foldovers) mesh, has long been taken for granted by the finite-element community [Field 1988]. Recent results by Floater [2001] reinforce this claim. Given a triangular mesh of a convex domain, Floater proves the following. A system of linear equations that represents each mesh node as a convex combination of its neighbors has as its solution a mesh with no foldovers. A convex combination in this context is a weighted sum with nonnegative weights. He also proves that iterative solution of such systems (smoothing) converges to the actual solution. This proof provides the necessary guarantee for the Cartesian grid mapping method to be valid; namely we know the smoothing will converge and the mapping from the smoothed grid to the original will be one-to-one.

In conclusion, both steps of the algorithm are guaranteed to converge to valid solutions, and combining the two mappings provides a mapping which is valid.

The new mapping appears to minimize both angular and area/length distortions. As mentioned above, classical results in differential geometry [Ahlfors and Sario 1960] assert that it is not possible to map a non-developable surface to a plane without some distortion of the distances. It is also impossible to map a faceted surface with non-zero local curvature to a plane without some angular distortion. Hence, at best, a compromise can be made between length preservation and conformity of the map, minimizing the two types of distortion. This is what the presented method tries to achieve.

6. EXAMPLES

Throughout the paper the texture mapping procedure is demonstrated on several examples of varying complexity. The properties of our examples are summarized in Table I. The number of iterations in the

Table I. Flattening Examples Data. (Time measurements are on a 800MHz/384MB RAM laptop.) * Signifies that the Harmonic Mapping Generates Foldovers in the Mesh for this Model

					Distortion							
	#	Runtime	# T^1	# T^2	ABF		Final		Conv. Comb.		Harmonic	
Model	faces	(sec.)	iter.	iter.	ang.	len.	ang.	len.	ang.	len.	ang.	len.
Folded Plane	280	1	2	1	6e-7	5e-5	7e-7	5e-5	0.21	0.11	0.19	0.11
Face	1394	22	5	43	0.003	0.019	0.006	0.01	0.28	0.18	0.011	0.04
Cat Head	257	3	4	48	0.029	0.7	0.079	0.24	0.135	1.28	0.039	0.74
3 Balls	1032	9	4	28	0.012	0.1	0.028	0.055	0.28	0.21	0.17*	0.17*
Dome	244	1	4	12	0.025	0.23	0.093	0.19	1.05	1.5	0.027	0.23
Mech. form	864	8	4	59	0.0018	0.039	0.0098	0.017	0.25	0.105	0.093	0.066
Full Cat	671	59	5	127	0.019	0.94	0.138	0.23	0.184	0.92	0.04	0.99
Cut Cat	671	2	4	14	0.0029	0.031	0.0071	0.016	0.62	0.17	0.12	0.3

second stage (the computation of T^2) indicates the outer iterations, involving the computation of the edge lengths (steps (1)–(3)). We measure both angular and length distortion of the flat surface with regard to the original mesh. We use

$$\frac{1}{3P}\sum_{i,j}\frac{\left(\alpha_i^j-\beta_i^j\right)^2}{{\beta_i^j}^2}$$

to measure angular distortion. This value is similar to the minimized functional in the ABF procedure. To measure distance distortion, we first compute the average edge length in two- and three-dimensional meshes, a_{2D} and a_{3D}, and the ratio between them $a=\frac{a_{2D}}{a_{3D}}$. For a non-distorted parameterization, the ratios of two-dimensional to three-dimensional lengths for each edge should be equal to a. Hence we use the following as distortion measure

$$\frac{1}{|\{e\}|}\sum_{e}\frac{\left(\frac{\|e_{2D}\|}{\|e_{3D}\|}-a\right)^2}{a^2}.$$

In both cases the division by the number of angles or edges, respectively is done to normalize the error, making it independent of mesh size.

The table compares the distortion with and without smoothing, and also compares the results to two popular parameterization methods: convex combinations [Floater 1997] and harmonic mapping [Eck et al. 1995]. Both methods require an *a priori* fixed planar boundary. In both cases a circle boundary was used, being the most general. For all the models the combined method gives lower angular and linear distortion than convex combinations. For models where the optimal boundary given by the combined method is far from a circle, the method also significantly outperforms harmonic mapping in both parameters. For models where the boundary is close to a circle (Figures 3, 5, and 8) the harmonic mapping results are only slightly worse than ABF results. This is due to the fact that harmonic mapping does in fact minimize angular distortion, but with a fixed boundary constraint. After the smoothing stage, the angular distortion of the combined mapping can therefore become higher than that achieved by harmonic mapping. However, the linear distortion is significantly lower and the texture looks (empirically) much better (Figures 3, 5, 8).

Figure 4 shows the texture mapping for the models in Figure 1. Figure 4(a) shows the texture generated for a nearly developable surface. It demonstrates the advantage of nonfixed boundary schemes,

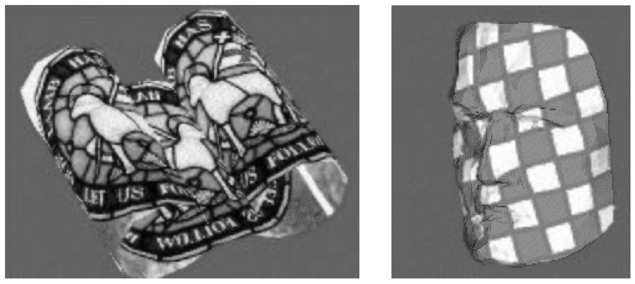

Fig. 4. Textures generated for the models in Figure 1.

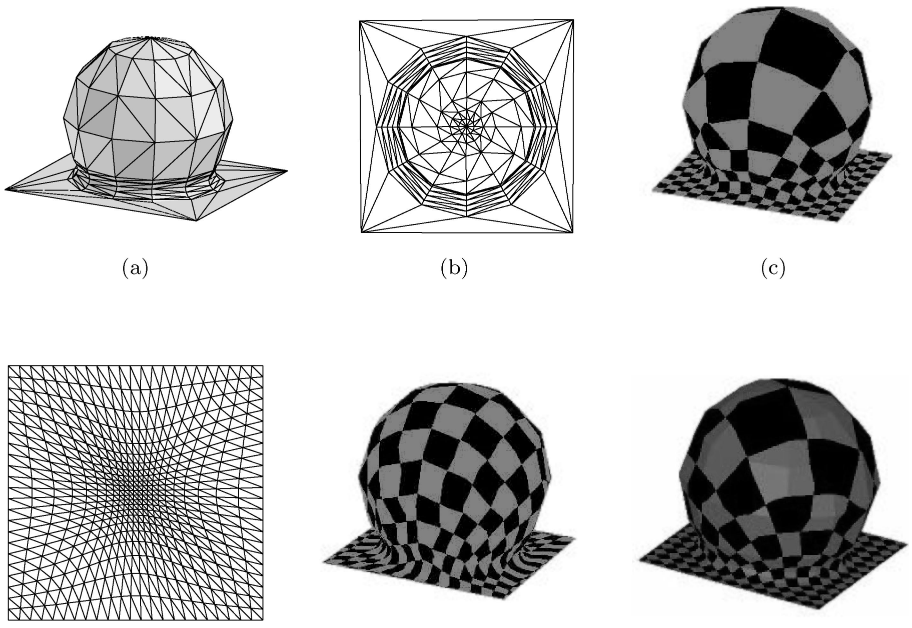

Fig. 5. Texture mapping for a dome shape. (a) Original surface (256 elements). (b) Flat parameterization (providing T^1). (c) Texture using only T^1 (without sizing function). (d) Smoothed grid G_2. (e) Final texture using T. (f) Texture using harmonic mapping.

which indeed find a zero distortion parameterization for developable surfaces. Figure 4(b) shows the texture for the face model in 1(c). The checkered pattern is used in this and other examples to emphasize the preservation of both angles and lengths in the mapping.

In Figure 3 we show a model of a cat's head (cut around the neck). The model has a large overall curvature, hence the distortion is relatively high. The figure shows the difference between using the result of the first step directly for the texture mapping and using the combined mapping. Adding the smoothing step dramatically reduces the distance and area distortion without affecting the angular deformation. An empiric comparison with the results of convex combinations and harmonic mapping shows that the texture looks significantly better after the combined mapping. Figure 5 shows another example emphasizing the need for the smoothing step. Notice that the results of harmonic mapping

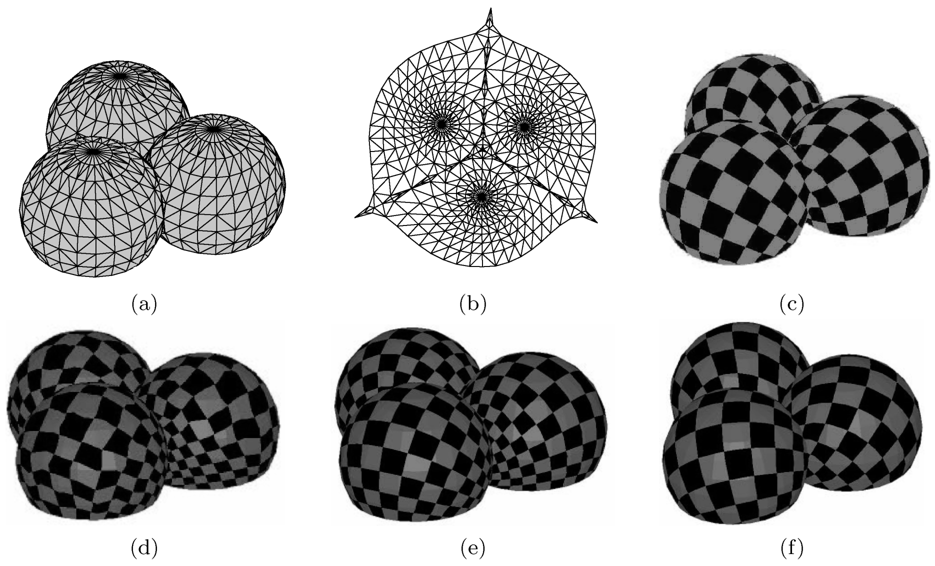

Fig. 6. Texture mapping for three balls (non-smooth surface). (a) Original surface (1032 elements). (b) Flat parameterization (providing T^1). (c) Final texture using T. (d) (e) Texture map using convex combinations and harmonic mapping. The harmonic mapping generates a discontinuity (folds over) at the meeting point of the three spheres. (f) Texture after applying overlay grid smoothing to harmonic mapping results.

are almost identical to ABF, since in this case the optimal outer boundary is identical to the fixed one (four equidistant points on a circle).

Figure 6 displays a surface build from three spheres positioned at 120° around a joint axis. The spheres are cut at about a quarter of the way up from the base, to create a surface which can be parameterized. The surface mesh is highly nonsmooth with high local curvature changes and multiple sliver triangles, but despite this, the parameterization converges in a small number of iterations and gives good results. Note that despite the increase in model size (compared to the cat and dome models), the number of iterations required to obtain a solution during the ABF stage does not increase. The results are compared to convex combinations and harmonic mapping. In the first case, the mapping does not preserve angles or distances that well. In the case of harmonic mapping, the mapping actually folds over at the meeting point of the three spheres (not seen on the picture) generating a minor texture discontinuity. This explains the high angular distortion 0.17 (Table I) Additionally, lengths are not preserved as well when comparing the top of the balls with the creases and bottom areas. After applying the overlay grid smoothing to harmonic map results (Figure 6(f)), the linear distortion is significantly reduced (from 0.17 to 0.096); the angular distortion is increased (to 0.19), but not significantly. The second step does not address the mapping discontinuity generated by the harmonic mapping, which therefore remains.

Figure 7 shows a mechanical form model, courtesy of Kay Hormann [Floater and Hormann 2001]. This model demonstrates the method's ability to handle surfaces with high variation in curvature. It includes nearly flat regions as well as highly curved ones.

Finally, Figure 8 shows a model of an entire cat (without the planar base). This example demonstrates the robustness of our method and its ability to handle highly curved surfaces. For any flattening algorithm the distortion increases with the increase in the surface-area-to-perimeter ratio as well as with the increase in local and global curvature. As a result the texture distortion for a model as complicated

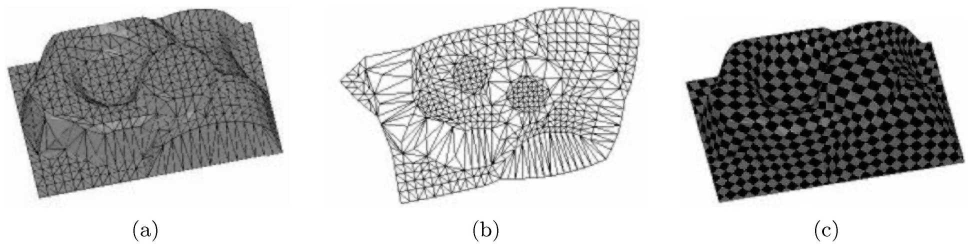

Fig. 7. Texture mapping for a mechanical form. (a) Original surface (864 elements). (b) Flat parameterization (after applying T^2 to T^1 results). (c) Final texture using T.

as the entire cat tends to be high (Figure 8(c)). As shown by the comparison to other methods, the combined mapping still provides a better result both visually and in terms of length distortion. At the same time the smoothing introduces high angular distortion (Figure 8(b)). This is due to the fact that even though conformal (quasi-conformal) mapping preserves length ratios locally, it can accumulate high distortion over large distances on the mesh. As a result the smoothing introduces relatively high angular deformation when reducing the length distortion. Due to the high initial linear distortion, the algorithm takes significantly more iterations to converge. However, it still remains within the one minute timeframe. The main alternative for reducing the distortion is the introduction of seams in the model as shown in Figure 8(f). This reduces the texture distortion but introduces texture discontinuities along the seam. The seams were added using an algorithm described in Sheffer [2002].

7. SUMMARY

We have proposed a new method for texture mapping based on the combination of a recently proposed algorithm for flattening three-dimensional surfaces and an algorithm for mesh smoothing that takes into account a given sizing function. We have outlined the main properties and underlying theory of our algorithm. Our method can be applied to very general problems and has few restrictions. Specifically, it does not require the boundary to be predefined or convex.

We used the ABF parameterization as a first stage of the mapping. The ABF is a quasi-conformal mapping based on minimizing the (relative) distortion of the mesh angles in each face subject to the necessary and sufficient conditions of a valid two-dimensional mesh.

The main contribution of this work is the introduction of the second mapping stage, aimed at reducing the length distortion of the parameterization. We suggested the use of a sizing function as a tool to measure the linear distortion of the mapping at any point in the planar domain. The sizing function is then used to set weights in a Laplacian smoothing procedure applied to a Cartesian grid covering the planar domain. The mapping obtained by barycentric projection of points on the smoothed grid to the regular grid has inverse linear distortion to the original ABF parameterization. Hence, combining the two provides a mapping with low linear distortion.

The use of an overlay grid rather than smoothing the existing planar mesh provides several major advantages. First, the boundary of the existing mesh need not be fixed in the smoothing process. Second, in cases of non-convex mesh boundary a smoothing procedure can lead to foldover in the mesh, that is generate mapping discontinuity. This problem is avoided when a rectangular grid is used instead. A rectangular grid also has the advantages of initially well shaped elements, avoiding numerical instabilities which can arise when smoothing the planar mesh or any mesh incorporating it, for example Delaunay mesh of its vertices. And finally, a rectangular grid has the advantage of simplicity during both construction and manipulation.

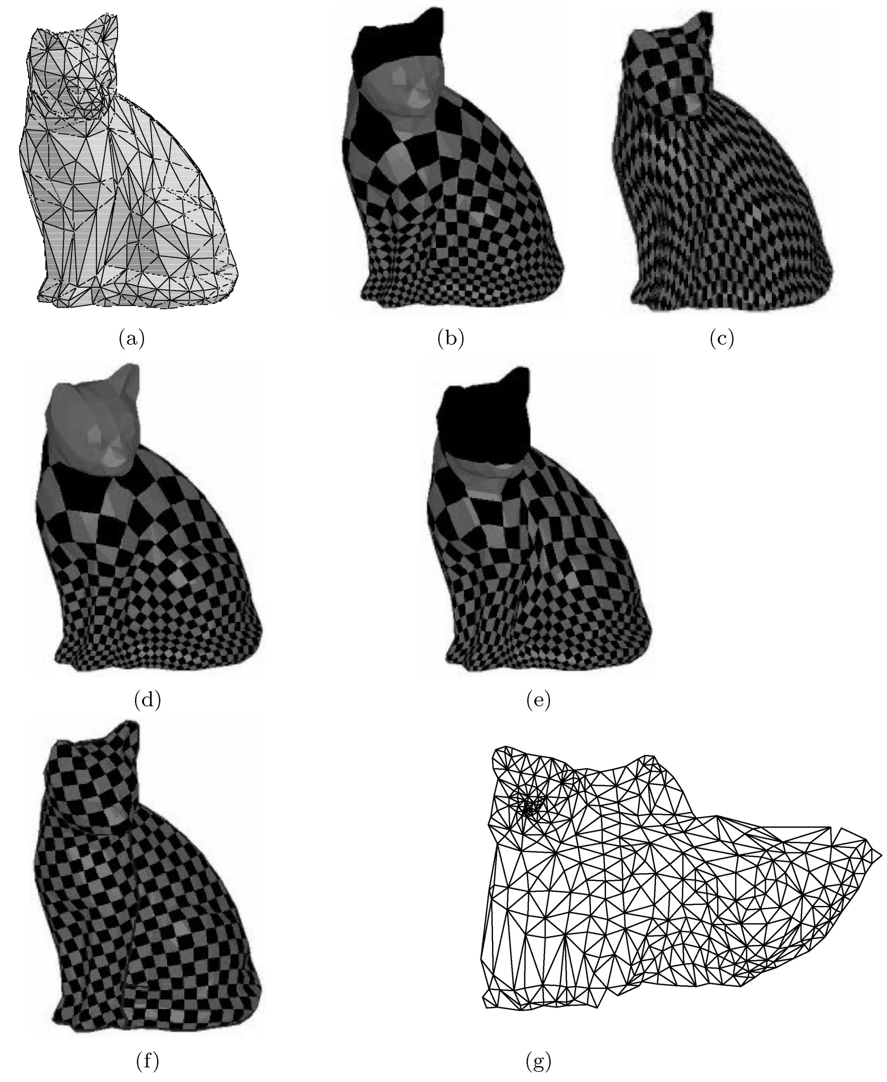

Fig. 8. Texture mapping for a full cat model (without the planar base). (a) The model (671 elements). (b) Texture using ABF. (c) Texture using the combined mapping. (d),(e) Texture using convex combinations and harmonic map respectively. (f) Texture generated after cutting seams (highlighted in blue). (g) Flat parameterization using seams.

The grid mapping is guaranteed to be one-to-one. Since quasi-conformal mapping preserves edge ratios locally, the local change in length distortion is small. As a result, in most cases, the smoothing procedure does not significantly alter the angles. In our examples we demonstrate that the proposed method generates good texture maps for complex surfaces in several seconds.

An important problem in future research is the trade-off between the quality of sizing approximation provided by the smoothing and the size of the Cartesian grid (Section 4.4). Increasing the size enables better capture of the sizing but increases both time and memory costs. This issue grows in importance when handling large models with hundreds of thousands of elements. We plan to investigate a multigrid approach for generating and smoothing the Cartesian grid.

Another interesting question that should be investigated is the trade-off between angular and linear distortion. Currently the smoothing procedure concentrates on reducing linear distortion. The user has no control on how much it increases the angular distortion in the process. Providing such control would be helpful for the user, especially for more complicated models where the trade-off issue becomes significant.

8. ACKNOWLEDGMENTS

The authors would like to thank S. Har-Peled for his contribution to the development of the surface cutting procedure and John Hart for useful discussions and pointers to the literature.

REFERENCES

AHLFORS, L. V. AND SARIO, L. 1960. *Riemann Surfaces*. Princeton University Press, Princeton, New Jersey.

AZARIADIS, P. N. AND ASPRAGATHOS, N. A. 2001. Geodesic curvature preservation in surface flattening through constrained global optimization. *Computer-Aided Design (CAD) 33*, 8, 581–591.

BENNIS, C., VÉZIEN, J. M., AND IGLÉSIAS, G. 1991. Piecewise surfaces flattening for non-distorted texture mappinng. *Proceedings SIGGRAPH 91 25*, 237–246.

BRANDT, A. 1977. Multi-level adaptive solutions to boundary-value problems. *Math. Comput. 31*, 333–390.

CANANN, S. A., TRISTANO, J. R., AND STATEN, M. L. 1998. An approach to combined Laplacian and optimization-based smoothing for triangular, quadrilateral, and quad-dominant meshes. *7th International Meshing Roundtable*, 479–494.

DEMMEL, J. W., EISENSTAT, S. C., GILBERT, J. R., LI, X. S., AND LIU, J. W. H. 1999a. A supernodal approach to sparse partial pivoting. *SIAM J. Matrix Anal. Appl. 20*, 3, 720–755.

DEMMEL, J. W., GILBERT, J. R., AND LI, X. S. 1999b. SuperLU user's guide. available from http://www.nersc.gov/xiaoye/SuperLU/index.html.

ECK, M., DEROSE, T., DUCHAMP, T., HOPPE, H., LOUNSBERY, M., AND STUETZLE, W. 1995. Multiresolution analysis of arbitrary meshes. *Computer Graphics (Annual Conference Series, 1995. SIGGRAPH '95)*, 173–182.

EELLS, J. AND LEMAIRE, L. 1988. Another report on harmonic maps. *Bull. London Math. Soc. 20*, 385–524.

EELLS, J. AND SAMPSON, J. H. 1964. Harmonic mapping of Riemanian manifolds. *Am. J. Math. 86*, 109–160.

FIELD, D. A. 1988. Laplacian smoothing and Delaunay triangulations. *Communications in Applied Numerical Methods 4*, 709–712.

FLOATER, M. S. 1997. Parametrization and smooth approximation of surface triangulations. *Computer Aided Geometric Design 14*, 231–250.

FLOATER, M. S. 2001. Convex combination maps. *Algorithms for Approximation IV*.

FLOATER, M. S. AND HORMANN, K. 2001. Parameterization of triangulations and unorganized points. *Principles of Multiresolution in Geometric Modelling*, 127–154.

FOLEY, J., VAN DAM, A., FEINER, S., AND HUGHES, J. 1992. *Computer Graphics*. Addison-Wesley Publishing, Massachusetts.

HORMANN, K. AND GREINER, G. 2000. Mips: an efficient global parameterization method. In *Curve and Surface Design: St. Malo 1999*. Vanderbilt University Press, 153–162.

HURDAL, M. K., BOWERS, P. L., STEPHENSON, K., SUMNERS, D. W. L., REHM, K., SCHAPER, K., AND ROTTENBERG, D. A. 1998. Quasi-conformally flat mapping the human cerebellum. In *Medical Image Computing and Computer-Assisted Intervention - MICCAI'99 (Lecture Notes in Computer Science)*. Springer, Berlin, 279–286.

LEVY, B. 2001. Constrained texture mapping for polygonal meshes. *Proc. SIGGRAPH 2001*, 417–424.

LEVY, B. 2002. Least squares conformal maps for automatic texture atlas generation. *Proceedings SIGGRAPH 2002*.

LEVY, B. AND MALLET, J. 1998. Non-distorted texture mapping for sheared triangulated meshes. *Proceedings SIGGRAPH 1998*, 343–352.

MAILLOT, J., YAHIA, H., AND VERROUST, A. 1993. Interactive texture mapping. *Proceedings SIGGRAPH 1993*, 27–34.

MALLET, J. L. 1989. Discrete smooth interpolation in geometric modeling. *ACM-Trans. Graph. 8*, 2, 121–144.

MARCUM, D. L. AND GAITER, J. A. 1999. Unstructured surface grid generation using global mapping and physical space approximation. *8th International Meshing Roundtable*, 397–406.

MCCARTNEY, J., HINDS, B. K., AND SEOW, B. L. 1999. The flattening of triangulated surfaces incorporating darts and gussets. *Computer-Aided Design (CAD) 31*, 249–260.

OWEN, S. J. 1998. A survey of unstructured mesh generation technology. *7th International Meshing Roundtable*, 239–267.

PRAUN, E., FINKELSTEIN, A., AND HOPPE, H. 2000. Lapped textures. *Proceedings SIGGRAPH 2000*, 465–470.

SAMEK, M., SLEAN, C., AND WEGHORST, H. 1986. Texture mapping and distortions in digital graphics. *The Visual Computer 2*, 5, 313–320.

SANDER, P. V., SNYDER, J., GORTLER, S. J., AND HOPPE, H. 2001. Texture mapping progressive meshes. *Proceedings SIGGRAPH 2001*, 409–416.

SHEFFER, A. 2002. Spanning tree seams for reducing parameterization distortion of triangulated surfaces. *Shape Modelling International'02*.

SHEFFER, A. AND DE STURLER, E. 2000a. Parameterization of faceted surfaces for meshing using angle based flattening. *Engineering with Computers 17*, 3, 326–337.

SHEFFER, A. AND DE STURLER, E. 2000b. Surface parameterization for meshing by triangulation flattening. *Proceedings of the 9th International Meshing Roundtable*, 161–172.

SORKINE, O., COHEN-OR, D., GOLDENTHAL, R., AND LISCHINSKI, D. 2002. Bounded-distortion piecewise mesh parameterization. Proceedings of IEEE Visualization'02(October).

TROTTENBERG, U., OOSTERLEE, C., AND SCHÜLER, A. 2001. *Multigrid*, 1 ed. Academic Press, London.

TUTTE, W. T. 1960. Convex representation of graphs. *Proceedings of the London Mathematical Society 10*.

TUTTE, W. T. 1963. How to draw a graph. *Proceedings of the London Mathematical Society 13*.

ZIGELMAN, G., KIMMEL, R., AND KIRYATI, N. 2001. Texture mapping using surface flattening via multi-dimensional scaling. *IEEE Trans. Vis. Comput. Graph.*

Received October 2001; revised May 2002; accepted May 2002