Parallel Computation of Spherical Parameterizations for Mesh Analysis

Theodoros Athanasiadis[1], Ioannis Fudos[1,*]

Abstract

Mesh parameterization is central to a broad spectrum of applications. In this paper, we present a novel approach to spherical mesh parameterization based on an iterative quadratic solver that is efficiently parallelizable on modern massively parallel architectures. We present an extensive analysis of performance results on both GPU and multicore architectures. We introduce a number of heuristics that exploit various system characteristics of the underlying architectures to speed up the parallel realization of our algorithms. Furthermore, we demonstrate the applicability of our approach to real-time feature detection, mesh decomposition and similarity-based 3D object retrieval. Finally, we offer visual results and a demonstration video.

Categories and Subject Descriptors (according to ACM CCS): Computer Graphics [I.3.1]: Hardware Architecture—Parallel processing Computer Graphics [I.3.5]: Computational Geometry and Object Modeling—Curve, surface, solid, and object representations Computer Graphics [I.3.m]: Miscellaneous—Mesh parameterization

1. Introduction

Fast mesh parameterization is central to many applications such as remeshing, filtering, texture mapping, compression, mesh completion and morphing. The surface that the mesh is parameterized on is typically referred to as the parameter domain. The purpose of mesh parameterization is to obtain a piecewise linear map, associating each triangle of the original mesh with a surface patch of the domain. An important goal of parameterization is to obtain *bijective (invertible)* maps, where each point on the domain corresponds to exactly one point of the mesh. The bijectivity of the map guarantees that there is no triangle flipping or overlapping.

Since the geometric shape of the domain surface patches will typically be different than the shape of the original triangles, angle and area distortion is introduced. The distortion of the parameterization is an important factor, therefore applications typically try to minimize the distortion for the whole mesh. Maps that minimize the angular distortion are called *conformal*, maps that minimize area distortion are called *authalic*, and maps that minimize distance distortion are called *isometric*.

In this work, we deal with the problem of computing a bijective mapping between a closed genus-0 mesh and a spherical domain, such that distortion is globally minimized. It is important to note that on such a domain, an arbitrary mesh mapping can be *authalic*, or *conformal* but not *isometric*, as it would have to be both authalic and conformal and in general this is not feasible.

Although mainly due to recent fundamental theoretical work [13, 3, 4, 38, 43, 21, 39, 9], there is a good understanding of the mathematical aspects underlying spherical mesh parameterizations, the problem of computing such parameterizations efficiently remains open.

The existing spherical mesh parameterization methods can roughly be classified into two categories : (i) methods that attempt to extend planar methods and (ii) methods that use some kind of non-linear optimization. Typical methods of the former category generalize planar parameterization methods of barycentric coordinates [45] to the spherical domain [3, 25, 23]. For example in the work of Haker et al. [23] the non-linear spherical problem is transfered to the disk and then the stereographic projection is used to obtain the spherical mapping. Other methods in this category proceed by splitting the mesh in two half-meshes and mapping each half individually onto a hemisphere [25].

Amongst the latter category, several methods use non-linear optimization [13, 38, 21] and usually have a higher computational cost. An indicative example of this category is the work by Praun and Hoppe [38] that combines a hierarchical method with the optimization of a stretch metric to obtain geometric images of closed meshes. An-

[*]Dept of Computer Science, University of Ioannina, GR45110 Ioannina, Greece

Email addresses: thathana@cs.uoi.gr (Theodoros Athanasiadis), fudos@cs.uoi.gr (Ioannis Fudos)

[1]University of Ioannina, Greece

other method following a hierarchical approach to obtain an approximate solution was introduced by Birkholz [11]. Firstly, the original mesh is simplified by using an edge-collapse technique until a tetrahedron is obtained. Afterwards, the simplification process is reversed by reinserting the vertices.

Another method which does not fall directly in either category, is presented by Gu and Yau [22] where the parameterization is based on the properties of the complex conformal gradient field. This method has also the advantage of being able to handle higher genus meshes.

The extension of planar methods to the spherical domain is attractive, since planar parameterizations require the solution of a simple linear system. However, they are usually required to introduce some cuts in the mesh. Such cuts induce distortion that may be undesirable for most applications.

Therefore, the generalization of planar barycentric mapping to the spherical domain is important. The weights assigned to the vertices offer some degree of control over the final parameterization and it is guaranteed that the final parameterization will be fold free provided that the weights are positive. Nevertheless, earlier fast methods by Alexa [3] that attempt to converge to valid barycentric parameterizations by employing simple projected *Gauss-Seidel* techniques are bound to fail in certain cases [39]. More recently, approaches that combine various techniques from the above-mentioned parameterization methods have appeared (e.g. the work by Saba et al. [39]). Still, they need several minutes to calculate parameterizations for a typical, by today's standards, model. In addition, prior methods do not consider the issues arising from a parallel implementation.

This paper makes the following technical contributions:

- Introduces a novel iterative quadratic solver for spherical mesh parameterization.
- Presents an efficient parallel implementation along with a number of heuristics that speed up significantly the parallel realization on modern hardware.
- Demonstrates the usefulness of the parallel mesh parameterization algorithm in several applications that exploit mesh morphology analysis.

The rest of the paper is organized as follows. Section 2 offers some background material on planar parameterizations and spherical parameterizations that use reduction to the planar case. Section 3 presents our iterative quadratic solver for spherical mesh parameterization. Section 4 describes our parallel implementation along with an experimental study and several heuristics that speedup the parallel realization on modern architectures. Section 5 presents applications of our technique on automated feature selection, mesh decomposition and similarity-based object retrieval. Section 6 offers conclusions.

2. Preliminaries

2.1. Planar Parameterizations

A *planar triangulation* is a simple triangulated plane graph the edges of which are represented by straight lines. The triangulation is called *valid* when the only intersections between its edges are at the common endpoints. It is known by Fary [16] that every planar graph G has a valid straight line representation. Therefore, for any planar graph there exist a set of points p such that the induced triangulated graph $T(G, p)$ is valid. A way to construct such a graph is described by Tutte [45]. The boundary vertices of G are mapped to a convex polygon with the same number of vertices and in the same order. Then, the interior vertices are placed such that each vertex is the centroid of its neighboring vertices. This was extended by Floater [17] who has proven that the vertices can be any convex combination of its neighboring vertices.

$$\begin{aligned} v_i &= \sum_{j \in N_i} w_{ij} v_j \\ \sum_{j \in N_i} w_{ij} &= 1 \\ w_{ij} &> 0 \end{aligned} \tag{1}$$

Consequently, for finding a one-to-one bijective mapping for a mesh with an open boundary to a convex parametric domain (unit disk, unit square), a sufficient condition is to find a set of positive weights that satisfy (1) and solve the corresponding linear system for those weights. The resulting system has always a unique solution provided that the boundary vertices are fixed. A straightforward choice is to choose equal weights so that each vertex represents the centroid of its neighbors. This is also referred to as barycentric mapping.

Though through this process a valid triangulation can always be created with no folded triangles, it is generally desirable that the resulting mapping minimizes a distortion metric that is related with some distinct shape characteristics of the original mesh.

Two possible sets of weights are described by Desbrum et al. [14] that produce *authalic* and *conformal* parameterizations. Nevertheless, these weights might be negative when the polygon is not convex [18]. Thus, although the final parameterization is *conformal* or *authalic* it may

contain folded triangles. In addition, since the weights are not positive, the corresponding linear system may be singular. To overcome this limitation for conformal weights we can clamp the angles between 0^o and 90^o degrees and therefore have strictly positive weights. Another possible set of weights are the *mean value* coordinates introduced by Floater [18]. These weights are not symmetric ($w_{ij} \neq w_{ji}$).

2.2. Spherical Parameterization by Reduction to the Planar Case

The methods for planar parameterizations [48] can be directly extended to a spherical domain by reducing the spherical parameterization problem to the planar case. A first approach to reducing the problem is to select two vertices as the poles (north and south) of the parameterization. Subsequently, a geodesic path must be established between the poles over the mesh surface. The path connecting the two poles defines the boundaries of the parameterization and thus the spherical surface can be converted to a unit disk. If equal weights are chosen and the poles are selected based on the largest distance along the z direction in object space, the resulting system is the linear system proposed by Brechbuhler et al. [13]. This approach yields a valid spherical parameterization for every mesh. Nevertheless, the choices for the poles and the path directly affect the quality of the parameterization.

Moreover, it turns out that selecting a good path is a difficult problem on its own and usually there is severe distortion in the final parameterization. The underlying issue is that the mapping of a set of boundary vertices to a fixed convex polygon is far from trivial. This is due to the fact that in most cases the boundary vertices do not form a convex polygon. Therefore, the obtained parameterizations exhibit high deformation. To tackle these difficulties Lévy et al. [31] construct parameterizations with free boundaries. Nevertheless, the seams introduced by the cuts in the mesh may be undesirable for certain applications.

A second approach to reduce the problem is to cut out a triangle from the mesh, leaving an open boundary, and to make the mesh homeomorphic to the unit disk. This approach, also referred to in the literature as stereo mapping, usually results in heavily distorted parameterizations since using the corresponding unit triangle as a boundary tends to cluster the remaining vertices in the center of the triangle.

3. Spherical Parameterizations

The main drawback of extending the planar methodologies to the spherical domain is the unnecessary distortion introduced in the parameterization. Therefore, it is advantageous to directly parameterize the meshes on the spherical domain to allow seamless continuous parameterizations of genus-0 meshes. Unfortunately, generalizing the barycentric coordinates and the planar parameterization theory to a spherical domain is not straightforward. Since the domain is non-planar, expressing a vertex on the sphere as a convex combination of its neighbors is in general not feasible. This would imply for example that if the neighbors of a vertex are co-planar, then the vertex should also lie on the same plane. Nevertheless, the mathematical aspects of the parameterization on the spherical domain have received increasing attention in the last few years. One important observation according to Gotsman et al. [21] is the following:

Theorem 1. *If each vertex position is expressed as some convex combination of the positions of its neighbors projected on the sphere (2), then the formed spherical triangulation is valid.*

$$\begin{aligned} v_i &= \frac{\sum_{j\in N_i} \lambda_{ij} v_j}{\| \sum_{j\in N_i} \lambda_{ij} v_j \|} \\ \sum_{j\in N_i} \lambda_{ij} &= 1 \\ \lambda_{ij} &= \lambda_{ji} \\ \lambda_{ij} &> 0 \end{aligned} \tag{2}$$

The spherical triangulation may be controlled by choosing a proper set of symmetric weights, similarly to the planar case. The system of equations (2) can also be expressed as a set of non-linear equations for the nodes $i = 1, ..., n$ of a mesh, seeking solution for the positions of the vertices (x_i, y_i, z_i) and the n auxiliary variables a_i,

$$\begin{aligned} a_i x_i - \sum_{j\in N_i} \lambda_{ij} x_j &= 0 \\ a_i y_i - \sum_{j\in N_i} \lambda_{ij} y_j &= 0 \\ a_i z_i - \sum_{j\in N_i} \lambda_{ij} z_j &= 0 \\ x_i^2 + y_i^2 + z_i^2 &= 1 \end{aligned} \tag{3}$$

The physical interpretation of the equations (3), assuming that the weights λ_{ij} correspond to spring constants, is the minimization of the sum of the squared weighted lengths (spring energy) subject to the vertices being on the sphere. Therefore, the energy that is minimized is:

$$E(v_1, v_2, ..., v_n) = \frac{1}{2} \sum_{(i,j)\in E} \lambda_{ij} \|v_i - v_j\|^2 \tag{4}$$

Generally, a solution of this system is not unique. Without restricting some degrees of freedom, there may be infinite solutions due to the possible rotations over the sphere. More importantly, there are degenerate solutions that satisfy (3). The most obvious one is observed when $a_i = 0$ where all the vertices of the parameterization collapse to one point on the sphere. Another possible degenerate solution can occur when the mesh contains a Hamiltonian cycle and the vertices are mapped to the equator of the sphere. Other degenerate solutions have been presented (for example by Gotsman et al. [21]).

Moreover, even a robust and stable non-linear solver may calculate degenerate solutions for the system of equations (3). A key observation that sheds light on this situation, is that as the solver iterations proceed, some triangles start growing and eventually pass through the equator of the sphere. The fundamental problem is that the spherical energy minimum occurs at a collapsed configuration, since the area of a planar triangle is always smaller than the area of the corresponding spherical triangle. Such cases are problematic, because there is an estimation error introduced in the calculation of the distortion metric over the surface. This error increases disproportionately with the size of the triangles. Therefore, the non-linear optimizer may minimize the corresponding distortion metric (energy function) over the sphere surface by increasing the size of the triangles with the largest error.

One way to avoid these degenerate solutions is to constrain three or more vertices, thus constraining the solver. However, in practice there are two problems: (i) the extra constraints introduce additional distortion in the parameterization, making the determination of a proper set of constrained vertices difficult, (ii) in addition, without paying special attention to the set of the constrained vertices, the non-linear problem may become infeasible.

3.1. An energy decreasing algorithm

Summarizing the above observations, if we try directly to solve (3), the following problems occur,

- *Non convexity*. The constraints $x_i^2 + y_i^2 + z_i^2 = 1$ are not convex. Therefore classical convex minimization cannot be used directly.
- *Non uniqueness*. The energy does not have a unique minimum and degenerate solutions always exist.
- *High computation cost*. Due to the above reasons, the usual approach of non-linear optimization has a high computation cost.

An approach to tackle these difficulties was proposed by Friedel et al. [19]. Here a penalty term d_{min}^{-2}, where d_{min} is the minimum distance of each triangle from the sphere center, was added in the corresponding planar quadratic energy and therefore there is no need to constrain any vertices or reproject the solution to the sphere. The motivation of this approach is to provide an upper bound of the spherical energy by scaling the corresponding planar energies of the triangles. Therefore, the corresponding problem (3) is transformed to an unconstrained one, that can be solved with standard methods.

Another possible approach to overcome the high computation cost is to use iterative procedures that attempt to converge to a valid parameterization by applying local improvement (relaxation) [3]. The idea is to reduce the spring energy of the points with Laplacian smoothing ignoring the sphere constraint and renormalise the solution to obtain valid spherical points. However, these algorithms are heuristic based and there is no guarantee that they will terminate. In practice, the iterative process can collapse and may require a restart. Furthermore, the termination criteria for such an algorithm are difficult to define. For example, a similar method is used by Saba et al. [39] to calculate an initial guess for the solution. However, the residual reduction criterion proposed to terminate the method is usually too conservative and the initial guess is far away from the solution. Thus, there is the need to complement it with a non-linear optimization step.

Moreover, techniques that rely upon non-linear software need to devise new parallel solutions and strategies to conform to new parallel architectures. For this reason, the effectiveness of all the techniques relying upon non-linear optimization is limited on inherently parallel architectures like modern GPUs.

To efficiently employ iterative procedures, a central issue is the renormalization step. The iterative procedure to solve the problem with an energy decreasing step and the renormalization of the solution can be described through the following steps:

1. Let vertices $v_1^0, ..., v_n^0$ be an initial guess for the solution
2. For $j = 0 \ldots$ until convergence
   (a) Find $v_1^{j+1}, ..., v_n^{j+1}$ such that $E(v_1^{j+1}, ..., v_n^{j+1}) \leq E(v_1^j, ..., v_n^j)$
   where v_i^{j+1} may not belong to the sphere
   (b) Set $v_i^{j+1} = \frac{v_i^{j+1}}{\|v_i^{j+1}\|}$ for $i = 1, ..., n$

The question that arises is whether the energy is still decreasing after the renormalization step and whether the algorithm converges to a solution of (3). The problem

is the unknown behavior of the energy after the normalization. In other words, the gain obtained by the energy decreasing step can be lost during renormalization.

It is therefore evident that the extension of iterative schemes in the spherical domain is not straightforward. A useful observation for the energy is the following,

Proposition 1. *If $v_i \in \mathbb{R}^3$ and $\|v_i\| \geq 1$ for the nodes $i = 1, ..., n$ of the mesh, then $\frac{v_i}{\|v_i\|}$ is on the sphere surface and moreover for the energy (4) with barycentric or conformal weights :*

$$E(\frac{v_1}{\|v_1\|}, ..., \frac{v_n}{\|v_n\|}) \leq E(v_1, .., v_n) \tag{5}$$

Proof. First we observe that $\forall(i, j)$:

$$\begin{aligned} \|\frac{v_i}{\|v_i\|} - \frac{v_j}{\|v_j\|}\|^2 \leq \|v_i - v_j\|^2 \\ \|v_i\| \geq 1 \\ \|v_j\| \geq 1 \end{aligned} \tag{6}$$

Therefore, by the definition of the energy (4), and because the barycentric and the (clamped) conformal weights [14] are positive, moving each vertex to the sphere cannot increase any term of the summation.
The above observation motivates the following iterative procedure,

1. Let vertices $v_1^0, ..., v_n^0$ be an initial guess for the solution
2. For $j = 0 \ldots$ until convergence
   (a) Find $v_1^{j+1}, ..., v_n^{j+1}$ such that $E(v_1^{j+1}, ..., v_n^{j+1}) \leq E(v_1^j, ..., v_n^j)$
   subject to $v_i^{j+1} \cdot v_i^j = 1$
   (b) Set $v_i^{j+1} = \frac{v_i^{j+1}}{\|v_i^{j+1}\|}$ for $i = 1, ..., n$

At each iteration, we seek a solution that minimizes the energy function subject to the constraint that the new vertices should be coplanar with the vertices in the previous iteration. Thus, the nonlinear constraints are converted to linear ones. Furthermore, the energy is decreasing after the renormalization step since $\|v_i^{j+1}\| \geq 1$. For the cases of barycentric and conformal quadratic energy functions, the energy minimizing problem in step 2(a) is a saddle point problem.

Saddle point problem solution

We first present an iterative process for solving the generic saddle point problem and then we reduce our problem to this process. Consider the block 2x2 linear system of the form,

$$\begin{pmatrix} A & B \\ B^T & 0 \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} r \\ q \end{pmatrix} \tag{7}$$
$$A \in \mathbb{R}^{n \times n}, B \in \mathbb{R}^{n \times m}, n \geq m$$

It is known that the solution of the linear system is equivalent to minimizing a function f subject to a set of m linear constraints [10],

$$\begin{aligned} \min_u f(u) = \frac{1}{2} u^T A u - u^T r \\ \text{s.t. } B^T u = q \end{aligned} \tag{8}$$

When A is a symmetric positive semidefinite matrix, this equality-constrained quadratic problem describes a (generalized) saddle point problem. In this case the variable v represents the vector of Lagrange multipliers. Any solution (u_*, v_*) of (7) is a saddle point for the Lagrangian

$$\mathcal{L}(u, v) = \frac{1}{2} u^T A u - r^T u + (Bx - q)^T v \tag{9}$$

where a saddle point $(u_*, v_*) \in \mathbb{R}^{n+m}$ satisfies

$$\mathcal{L}(x_*, y) \leq \mathcal{L}(u_*, v_*) \leq \mathcal{L}(u, v_*), u \in \mathbb{R}^n \text{ and } v \in \mathbb{R}^m \tag{10}$$

Under the following conditions there is a solution to the system (7) and it is unique,

Theorem 2. *Let,*

1. *A be a real symmetric positive semi-definite $n \times n$ matrix*
2. *B be a real $n \times m$ matrix with full column rank*
3. *A and B^T have no nontrivial null vectors in common*

Then (7) has a unique solution for u,v.

Proof. See Theorem 2 by Benzi et al. [10].

Given a non zero vector u^0 and assuming a splitting of the matrix $A = M - N$, an iterative scheme to solve (7) is the following,

$$\begin{pmatrix} M & B \\ B^T & 0 \end{pmatrix} \begin{pmatrix} u^{k+1} \\ v^{k+1} \end{pmatrix} = \begin{pmatrix} N & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} u^k \\ v^k \end{pmatrix} + \begin{pmatrix} r \\ q \end{pmatrix} \tag{11}$$

Therefore to solve (7) a procedure is,

1. Solve $M\tilde{u}^{k+1} = Nu^k + r$
2. Solve $(B^T M^{-1} B)v^{k+1} = B^T \tilde{u}^{k+1} - q$
3. Solve $M(u^{k+1} - \tilde{u}^{k+1}) = -Bv_{k+1}$

If $A = D - L - L^T$, where D is a nonsingular diagonal matrix and L is a strictly lower triangular matrix, then the iterative scheme is convergent for the following choices of M and N [15],

$$M = \frac{1}{\omega}D, N = \frac{1-\omega}{\omega}D + L + L^T \tag{12}$$

with $\omega > 0$ so small that $\frac{2}{\omega}D - A$ is a positive definite matrix.

$$M = \frac{1}{\omega}D - L, N = \frac{1-\omega}{\omega}D + L^T \tag{13}$$

with $0 < \omega < 2$.

(12) and (13) are the usual *Jacobi* and *Gauss-Seidel* iterations with over relaxation parameter ω.

To map our problem to (11), we first establish some necessary notation. Without loss of generality, we assume that some vertices are fixed on the sphere and those are numbered last. Therefore, let n and b represent the number of free and fixed vertices respectively. Next, we define the weighted $m \times m$ block matrix Λ for the vertices of the mesh,

$$\begin{pmatrix} \sum_j^m w_{1j}I_3 & -w_{12}I_3 & & \cdots & -w_{1m}I_3 \\ -w_{21}I_3 & \sum_{j=1}^m w_{2j}I_3 & & & -w_{2m}I_3 \\ \vdots & \vdots & \ddots & & \vdots \\ & & & \sum_{j=1}^m w_{m-1j}I_3 & -w_{m-1m}I_3 \\ -w_{m1}I_3 & -w_{m2}I_3 & \cdots & -w_{mm-1}I_3 & \sum_{j=1}^m w_{mj}I_3 \end{pmatrix}$$

where $m = n + b$, w_{ij} is the weight (conformal or barycentric) of the edge connecting i and j vertices, and I_3 is the 3×3 identity matrix. Moreover, $w_{ij} = 0$ iff $(i, j) \notin E$. In addition, E,V denote the edges and the vertices of the mesh, and (x_i, y_i, z_i) are the coordinates of the *ith* vertex of the mesh. We further denote by $\tilde{\Lambda} = [A_U, A_C]$ the matrix that is derived from Λ by deleting its last $3b$ rows. A_U contains all of the weights corresponding to the free vertices and is an $3n \times 3n$ matrix, A_C contains all of the weights corresponding to the fixed vertices and is an $3n \times 3b$ matrix.

We use the following matrices to minimize the quadratic energies for barycentric, and conformal mesh mappings,

$$A := A_U, A \in \mathbb{R}^{3n \times 3n} \tag{14}$$

$$B := \begin{pmatrix} x_1 & & \\ y_1 & & 0 \\ z_1 & & \\ & \ddots & \\ & \ddots & \\ & \ddots & \\ & & x_n \\ 0 & & y_n \\ & & z_n \end{pmatrix}, B \in \mathbb{R}^{3n \times n} \tag{15}$$

Furthermore, we have the vectors,

$$u^k = [x_1^k, y_1^k, z_1^k, ..., x_n^k, y_n^k, z_n^k]^T, u^k \in \mathbb{R}^{3n} \tag{16}$$

$$r = -A_C[x_{n+1}, y_{n+1}, z_{n+1}, ..., x_{n+b}, y_{n+b}, z_{n+b}]^T, r \in \mathbb{R}^{3n} \tag{17}$$

$$q = [1, ..., 1]^T, q \in \mathbb{R}^n \tag{18}$$

using the splitting of A, $M = \frac{1}{\omega}D$ and $N = \frac{1-\omega}{\omega}D + L + L^T$ (Jacobi iteration) we observe that $B^T M^{-1} B = \omega D$ (since the vertices are on the sphere). Furthermore, assuming that we use the normalized weights at each vertex, the iterative procedure becomes,

Algorithm 1 Iterative Saddle point solution

1: **for** k=0 until convergence **do**
2: **for** i=1 until n **do**
3: $[\tilde{x}_i, \tilde{y}_i, \tilde{z}_i] = (1 - \omega)[x_i^k, y_i^k, z_i^k] + \omega(\sum_{j=1, j\neq i}^n w_{ij}[x_j^k, y_j^k, z_j^k] + \sum_{j=n+1}^{n+b} w_{ij}[x_j, y_j, z_j])$
4: $\lambda_i = [x_i, y_i, z_i]^T[\tilde{x}_i, \tilde{y}_i, \tilde{z}_i] - 1$
5: $[x_i^{k+1}, y_i^{k+1}, z_i^{k+1}] = [\tilde{x}_i, \tilde{y}_i, \tilde{z}_i] - \lambda_i[x_i, y_i, z_i]$
6: **end for**
7: **end for**

Proposition 2. *The iterative algorithm 1 converges to the unique solution of the saddle point problem (7) for symmetric weights.*

Proof. We first prove the conditions for Theorem 2. Because the mesh is connected and a positive weight is associated with each edge, A is irreducible. In addition, $|a_{ii}| \geq \sum_{j=1, j\neq i}^{3n} |a_{ij}|$ for each row. Equality is true if the corresponding vertex is connected only with free nodes. Consequently, if the weights are symmetric and at least one node is fixed, the matrix A is a hermitian irreducibly diagonally dominant matrix, and therefore positive-semidefinite with $ker(A) = 0$. In addition, it is obvious that the matrix B has full column rank since at

least one of the components of each vertex will be non-zero in the spherical domain. Therefore, the conditions of Theorem 2 are satisfied. Furthermore, $\frac{2}{\omega}D - A$ is positive definite for $0 < \omega < 1$ since it is a strictly diagonally dominant and irreducible matrix with positive diagonal elements. Thus, the iterative procedure converges to the unique solution.

4. Parallel Parameterization

We have developed a parallel implementation of the algorithm presented in Section 3. The software is available at `http://www.cs.uoi.gr/~fudos/smi2011.html`. The inherent parallelism of the specific method enables us to map the problem to the hardware as efficiently as possible. As an API for our implementation, we have used OpenCL 1.1 to achieve almost direct portability of our core source to both GPU and CPU based architectures.

To maximize the performance of our parallel implementation we have considered a number of key factors. A characteristic that affects parallel algorithm effectiveness on all architectures is the number of sequential steps of the algorithm. To maximize the parallel execution we have employed the Jacobi iteration.

We present heuristics that optimize the parallel performance of the proposed algorithm. We have investigated the employment of three optimization principles and we have evaluated their effect on the performance for both GPU and multicore architectures:

- Optimize memory usage to maximize instruction throughput.
- Test for convergence only every n iterations have been carried out, to reduce the data synchronization overhead.
- Increase the processing unit cache hit ratio.

One important consideration in modern GPUs are the effective memory usage to achieve the maximum memory bandwidth and the optimization of the instruction usage to achieve the maximum instruction throughput [36]. To optimize the memory usage we have further minimized the data transfer between the host and the GPU device by reducing the number of residual tests needed for the convergence test for the solution of the saddle point problem. Since the convergence of the Jacobi is guaranteed, it is not necessary to test the convergence of the linear system at every iteration. More specifically, we have opted to perform convergence tests only every a certain number of iterations. This is beneficial to modern GPUs since it is better to increase the OpenCL kernel invocations on the GPU and reduce the synchronization between the host and the device.

We have conducted experiments with equal and conformal weights. Since conformal weights can be negative in certain cases, all input angles can be clamped between 5^o and 85^o degrees as suggested by [19]. For all the examples, the algorithm termination δ for the residual reduction was set to 10^{-7}.

To reduce the data synchronization overhead, the residual of the saddle point problem was tested for convergence every 1000 iterations. The sparse residual check policy has a large impact on GPUs and especially on GPUs with slower buses (PCIe 1.0).This has a small positive effect on multicores as well. Table 1 summarizes the results of the parameterization on different commonly used models [2, 12] using an NVIDIA GTX 480. We have also obtained performance results by carrying out our algorithm on two multi core processors, an Intel Core Duo E6600 and an Intel Core i7-870. Figure 1 compares the performance of these architectures. Table 3 illustrates the difference in running times on the GPU and on the CPU, while Table 4 presents a direct comparison with the running times of the publicly available parameterization software by Saba et al. [39]. For the results of Table 4 we have applied our approach to the models accompanying the software, using though a much smaller residual target than the one used by Saba et al. [39].

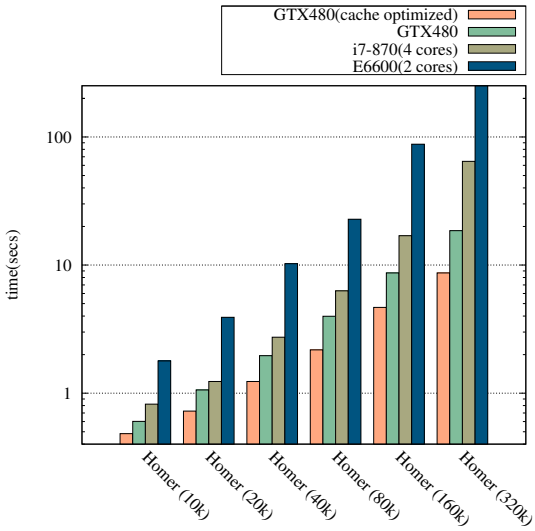


Figure 1: Performance difference between GPU and CPU OpenCL implementation in logarithmic scale. Results with vertex cache optimization are also included.

Finally, many cache misses can be avoided by combining the method with preprocessing techniques that further improve the cache locality of the mesh indices. Figure 2

Table 1: Numerical results for finding a spherical parameterization on the GPU with different models. In this context, the number of iterations is the number of saddle point problems solved.

model	map	# vertices	# faces	# iterations	L_2 res ($\times 10^{-8}$)	time (secs)
Suzanne	Barycentric	7573	15142	4	5	0.575
Suzanne	Conformal	7573	15142	3	5	0.589
Gargoyle	Barycentric	24990	49976	4	2	1.706
Gargoyle	Conformal	24990	49976	3	6	2.326
Igea	Barycentric	25586	51168	3	4	0.908
Igea	Conformal	25586	51168	2	3	0.936
Lion Vase	Barycentric	38952	77900	3	3	1.567
Lion Vase	Conformal	38952	77900	3	3	2.053
Homer	Barycentric	78850	157696	3	1	4.923
Homer	Conformal	78850	157696	3	4	10.920
Buste	Barycentric	183580	367156	3	1	13.759
Buste	Conformal	183580	367156	2	1	22.667

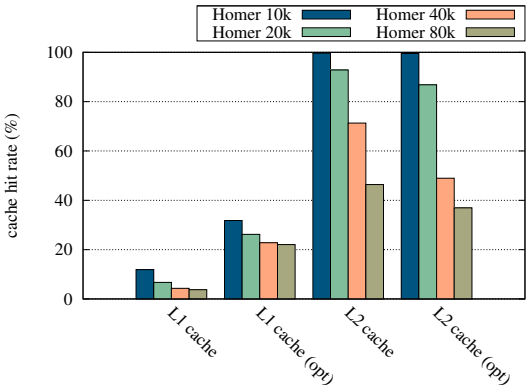


Figure 2: Cache hit rate statistics on the GPU. Results with a vertex locality optimization step from Sander et al. [40] are also shown for comparison.

presents experimental results regarding the cache hit ratio for various models. The cache efficiency is decreased as the size of the model increases. As a fast preprocessing step, we have used the vertex locality optimization proposed by Sander et al. [40]. The optimized results are also included in Table 2 for comparison. We observe that with the optimized meshes the computation time is reduced up to 50% on the GPU. On multicore architectures this has no effect since the CPU cache is usually large enough to fit the entire mesh index.

5. Applications

5.1. Mesh Segmentation

Different measures have been used to detect structural features on meshes such as curvature based computations [49], average error from fitting with surface patches, planes, cylinders or spheres [30, 6], dihedral angles of adjacent triangles [50], electrical charge density distribution [47], region flatness or smoothness [24], geodesic distances [27] and convexity [37]. Such measures have been used in conjunction with region growing [41], watershed functions [28], reeb graphs [5], skeletons [32], clustering [27] and hierarchical clustering [20] and segmentation [6], boundary extraction [29], Morse theory [46] or probabilistic fields [37]. For an extended survey of mesh decomposition techniques the reader is referred to Agathos et al. [1] and Attene et al. [7].

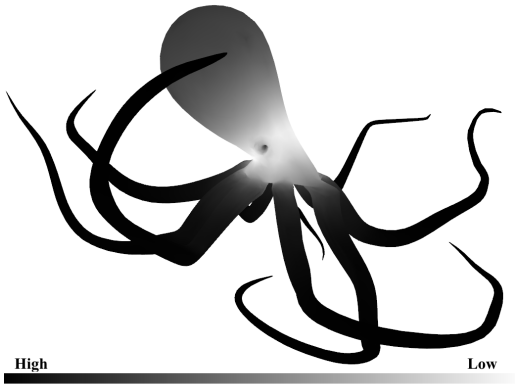


Figure 3: Visualization of the area stretch factor.

Lien and Amato [33] and Lien et al. [34] present a shape decomposition and skeletonization method for polyhedrons that is based on approximate convex decom-

Table 2: Numerical results for barycentric mapping on the GPU with different levels of detail

model	# vertices	# faces	# iterations	L_2 res ($\times 10^{-8}$)	time (secs)	opt time (secs)
Homer (Lod1)	5002	10000	4	10	0.577	0.483
Homer (Lod2)	10002	20000	4	2	1.059	0.725
Homer (Lod3)	20002	40000	4	2	1.961	1.224
Homer (Lod4)	40002	80000	3	2	3.986	2.178
Homer (Original)	78850	157696	3	1	4.923	3.636

Table 3: Comparison of running times (in secs) between GPU and CPU with different core configurations.

model	map	# vertices	# faces	# iterations	GTX 480	i7-870 (4)	i7-870 (2)	i7-870 (1)
Gargoyle	Barycentric	10002	20000	4	0.946	1.186	1.950	3.135
Gargoyle	Conformal	10002	20000	4	0.949	1.045	1.685	2.714
Torso	Barycentric	11362	22720	4	0.718	1.107	1.731	2.808
Torso	Conformal	11362	22720	3	0.870	1.123	1.747	2.745
Skull	Barycentric	20002	40000	3	0.649	1.076	1.719	2.904
Skull	Conformal	20002	40000	2	0.643	0.920	1.373	2.230
Bunny	Barycentric	67038	134074	3	1.217	3.616	6.635	12.038
Bunny	Conformal	67038	134074	2	2.158	3.778	7.737	14.118

position. This principle was used by Stamati and Fudos [44] to decompose point clouds into components that represent features by using the concavity intensity to detect saddle points and the discrete curvature to detect edges. A more general principle that can handle even complex (non star-shaped) objects is to derive a minimal length 3D path (curve segment) to connect the point to the convex hull without crossing the mesh. Since the convex hull has a straightforward spherical parameterization, finding a path that connects v to the convex hull corresponds to blowing the interior of the object until it expands to the convex hull. This is a CPU-intensive process that can be simulated with a spring system [34, 35]. Our efficient parallel spherical mesh parameterization can derive a measure that is somehow related to the minimum path by computing the deformation that has been applied to the adjacent triangles of a vertex. This yields a more robust measure that can be computed exactly very efficiently. As compared to very sophisticated techniques such as the one presented by Katz et al. [26] that uses mesh coarsening, MDS transforms and refinement, our work yields results of comparable quality much faster. Post processing can always be used for application specific mesh-segment refinement.

Our approach is based on the key idea that the spherical embedding represents a pose invariant representation of the mesh for quasi articulated objects. Any spherical embedding is expected to create some dense concentrations of faces on the sphere due to the prominent extremities of the mesh. The extruding parts of the meshes, for example the limbs, are expected to be mapped to relatively small regions on the sphere. Therefore, the *area stretching factor* (ratio of area in the surface and in the mapping) in those parts is expected to be much higher than in the rest of the mesh. Moreover, in the case of the conformal map the angles are generally preserved in the mapping. Therefore, the area distortion is affected more by the geometry of the model and less by the distortion introduced in the mapped angles. Consequently, our thesis is that the spherical embedding of a mesh contains a substantial amount of information about its geometric shape. To illustrate, consider Figure 3 where the area stretch factor of the conformal parameterization for the octopus model is depicted. Figure 4 visualizes the distortion of the parameterization in four typical models with limbs.

Definition 1. The *Area stretch factor* of a vertex v_0 of a mesh denoted by $A(v_0)$ is the average of the area stretch deformation of its adjacent faces.

Furthermore, we have carried out a number of experiments with a region growing approach that takes advantage of the above observation. The method starts from an initial vertex (the seed) and expands while a threshold in the variation of the area stretch factor is satisfied. Figures 5 and 6 show the segmentation of four typical models and a comparison with the mesh segmentation method presented by Katz and Tal [27]. Katz and Tal [27] quoted running times of a few minutes for segmenting moderately sized meshes by relying on mesh simplification to

Table 4: Comparison of running times of our method vs the one by Saba et al. [39] on the same CPU (E6600).

model	map	# vertices	# faces	[39] method (secs)	our method (secs)
Gargoyle	Barycentric	10002	20000	23.58	3.422
Gargoyle	Conformal	10002	20000	62.86	2.734
Torso	Barycentric	11362	22720	26.50	2.704
Torso	Conformal	11362	22720	84.49	2.719
Skull	Barycentric	20002	40000	67.73	2.828
Skull	Conformal	20002	40000	87.26	2.469

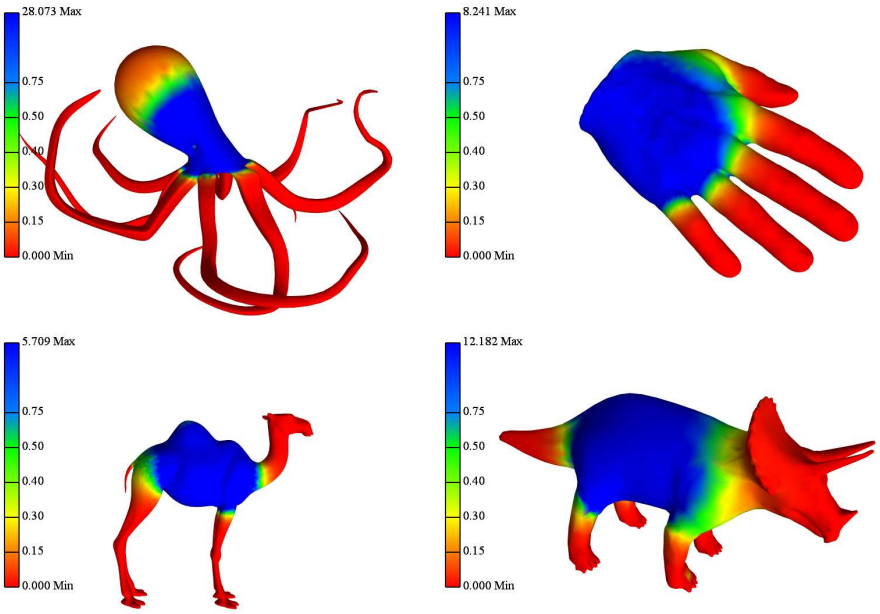


Figure 4: Visualization of the ratio between the mapped area and the original surface. Blue and red colors correspond to high and low distorted areas respectively.

reduce the computation cost. To compare the efficiency of our method to theirs, it is important to note that our approach does not require any pre-process or simplification of the meshes. In all the experiments performed, the running time for our segmentation method is dominated by the parameterization step. For meshes up to 100K triangles, the overall time was less than 5 seconds.

Definition 2. A vertex v_0 is called a *seed candidate* if and only if $A(v_0)$ exhibits a local minimum or maximum at v_0.

We first derive the candidate seeds for our model. Subsequently, the seeds are sorted in descending order according to the area stretch factor and our region growing approach is instantiated from these seeds. When a candidate seed is included in a new region it is removed from the set. This results in a number of regions that represent object extremities.

5.2. Texture mapping

With our method angle-preserving parameterizations can be efficiently obtained and are often suitable for texture mapping. Figure 10 shows the differences between the parameterizations using equal and conformal weights. Furthermore, in Figures 9, 11, 12 and 13 we have applied a checker texture to the meshes, by using the spherical coordinates of the parameterization as *uv* coordinates, to visualize the deformation differences between the two types of parameterization. By acquiring this correspondence one may apply deformations affecting the area of texture features of the original texture, prior to mapping it to our original mesh.

5.3. Shape Search

One important goal of shape searching algorithms is to represent the mesh vertices with a pose invariant representation [8, 42]. Initial tests showed that the spherical parameterization can be used to find similar poses of meshes. The key idea is to compare the signatures derived from the conformal mappings of the meshes. To derive the signature of a mesh we use the histogram of the area stretch factor. Since a conformal mapping is independent of the resolution of the mesh and preserves the consistency of the orientation, we can further make the signature invariant to the tessellation of the mesh. This can be achieved with uniform or random sampling of the meshes. Figures 7, 8 show the histograms obtained from various poses of the same meshes. A thorough comparison of the proposed shape search measure to other approaches, in terms of hits and misses, remains as future work.

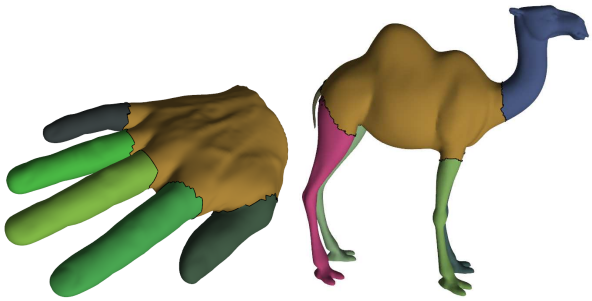
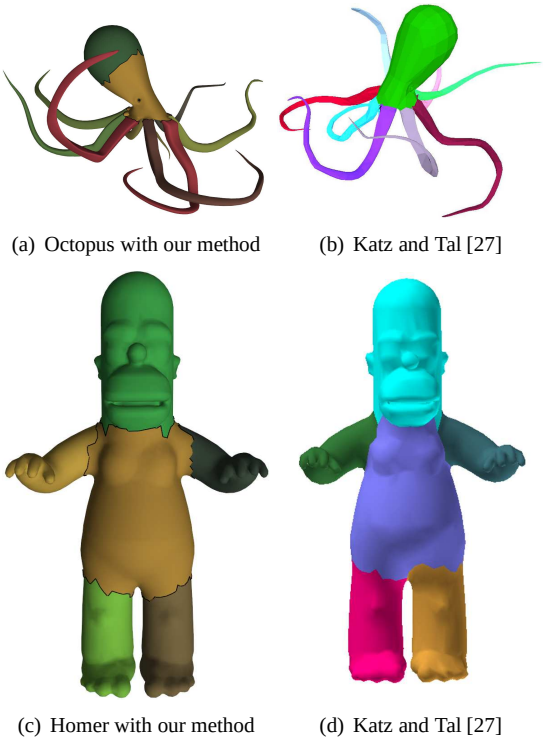


Figure 5: Automatic mesh segmentation.



(a) Octopus with our method (b) Katz and Tal [27]

(c) Homer with our method (d) Katz and Tal [27]

Figure 6: Comparison of mesh segmentation results.

6. Conclusions

We have presented a simple and efficient parallel numerical scheme to approximate a spherical parameterization of a genus-0 mesh. We have successfully used our scheme to parameterize meshes of up to 400K triangles in less than 25 secs.

We have carried out a large number of experiments to validate that our iterative method converges to the actual bijective mapping. Using a number of standard graphical models, we have confirmed that in each case the L_2 residual is decreased below a small tolerance value (10^{-7}).

A possible extension of our work would be a theoretical result of the convergence behavior. This could be reached from the fact that the algorithm is energy-decreasing so that the iterative solution follows a path close to the solution of the non-linear equations (3).

Finally, exploring other implementation options might result in improved computational efficiency. In particular, a more sophisticated iterative method for solving the saddle point problem can be used to reduce the number of iterations required for convergence. Nevertheless, our profiling tests show that our implementation is dominated by the memory access latency and the data synchronization delay between the host and the GPU. Therefore, whether such an improvement would be beneficial remains to be determined.

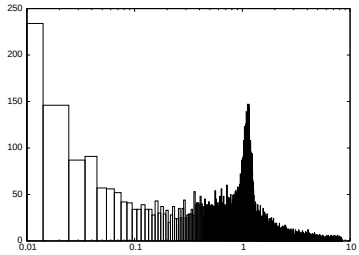
Acknowledgement

The work in this paper was partially supported by a Heraclitus II grant through the operational programme ”Education and Lifelong Learning” which is co-financed by Greece and the European Union through the European Social Fund.

References

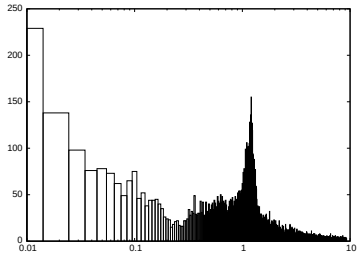
[1] Agathos, E., Pratikakis, I., Perantonis, S., Sapidis, N., Azariadis, P., 2007. 3d mesh segmentation methodologies for cad applications. Computer-Aided Design and Applications 4, 827–842.

(a) Pose 1



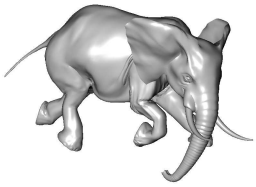
(b) Histogram

(c) Pose 2

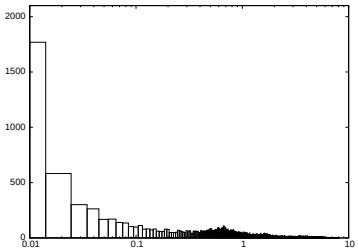


(d) Histogram

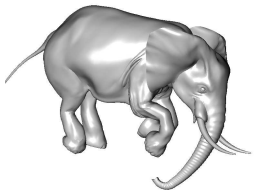
Figure 7: Hand mesh.



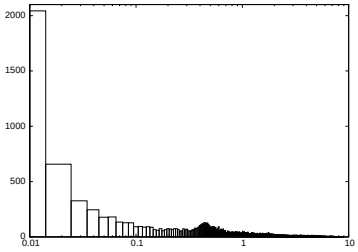
(a) Pose 1



(b) Histogram

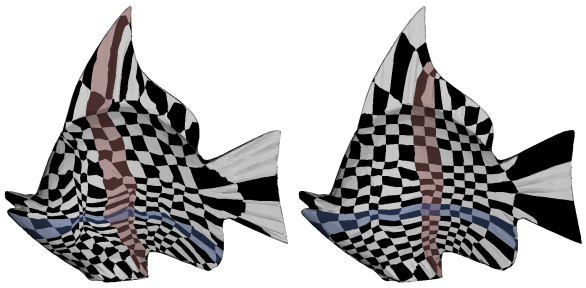


(c) Pose 2



(d) Histogram

Figure 8: Elephant mesh.



(a) Barycentric (b) Conformal

Figure 9: Comparison of texture mapping results for the Fish model [2].

[2] Aim@shape, AIM@SHAPE Project. AIM@SHAPE Shape Repository v4.0, Department of Genova, Institute for Applied Mathematics and Information Technologies, CNR, http://shapes.aimatshape.net.

[3] Alexa, M., 2000. Merging polyhedral shapes with scattered features. The Visual Computer 16 (1), 26–37.

[4] Alexa, M., 2002. Recent advances in mesh morphing. Computer Graphics Forum 21 (2), 173–197.

[5] Antini, G., Berretti, S., Bimbo, A. D., Pala, P., July 2005. 3d mesh partitioning for retrieval by parts applications. In: Multimedia and Expo, 2005. ICME 2005. IEEE International Conference on. pp. 1210 –1213.

[6] Attene, M., Falcidieno, B., Spagnuolo, M., 2006. M.: Hierarchical mesh segmentation based on fitting primitives. The Visual Computer 22, 181–193.

[7] Attene, M., Katz, S., Mortara, M., Patane, G., Spagnuolo, M., Tal, A., June 2006. Mesh segmentation - a comparative study. In: Shape Modeling and Applications, 2006. SMI 2006. IEEE International Conference. pp. 7 –7.

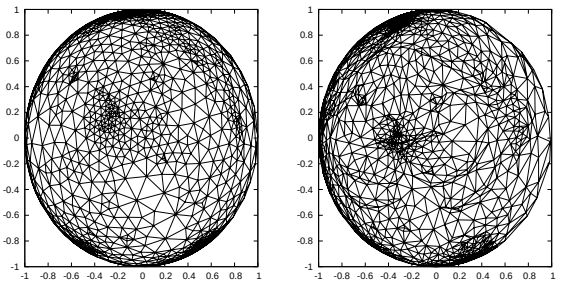
[8] Ben-chen, M., Gotsman, C., 2008. Characterizing shape using conformal factors. In: Eurographics Workshop on 3D Object Retrieval.

[9] Ben-chen, M., Gotsman, C., Bunin, G., 2008. Conformal flattening by curvature prescription and metric scaling. Computer Graphics Forum 27.

[10] Benzi, M., Golub, G., Liesen, J., 2005. Numerical solution of saddle point problems. Acta Numerica, 1–137.

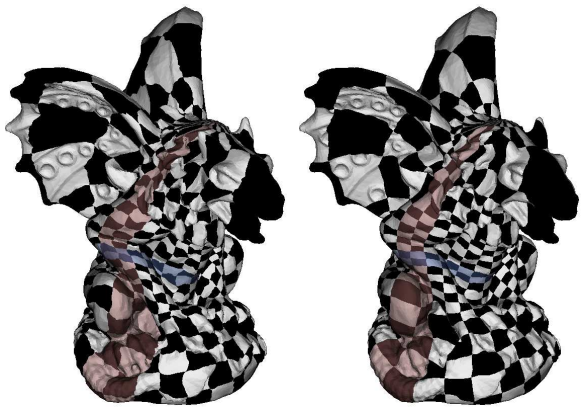
[11] Birkholz, H., 2004. Shape-preserving parametrization of genus 0 surfaces.

[12] Blender, Blender Foundation. Blender Suite, Open Source Suite, http://www.blender.org.



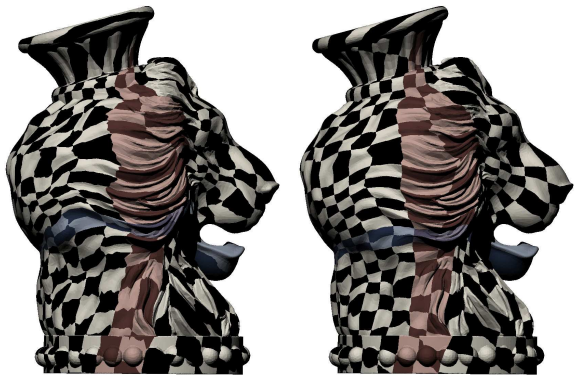
(a) Barycentric (b) Conformal

Figure 10: Comparison of sphere mapping results for the Fish model.



(a) Barycentric (b) Conformal

Figure 11: Comparison of texture mapping results for the gargoyle mesh [2].



(a) Barycentric (b) Conformal

Figure 12: Comparison of texture mapping results for the Lion Vase model [2].

[13] Brechbuhler, C., Gierig, G., Kubler, O., August 1995. Parametrization of closed surfaces for 3d shape description. Computer Vision and Image Understanding 61 (2), 154–170.

[14] Desbrum, M., Meyer, M., Alliez, P., 2002. Intristic parameterization of surface meshes. In: Eurographics Proceedings. pp. 209–218.

[15] Dyn, N., W. E. Freguson, J., July 1983. The numerical solution of equality-constrained quadratic programming problems. Mathematics of Computation 41 (163), 165–170.

[16] Fary, I., 1948. On straight line representation of planar graphs. Acta Univ. Szeged Sect. Sci. Math. 11, 229–233.

[17] Floater, M. S., 1997. Parametrization and smooth approximation of surface triangulations. Computer Aided Geometric Design 14, 231–250.

[18] Floater, M. S., 2003. Mean value coordinates. In: CAGD. pp. 19–27.

[19] Friedel, I., Schröder, P., Desbrun, M., 2005. Unconstrained spherical parameterization. In: SIGGRAPH '05: ACM SIGGRAPH 2005 Sketches. ACM, New York, NY, USA, p. 134.

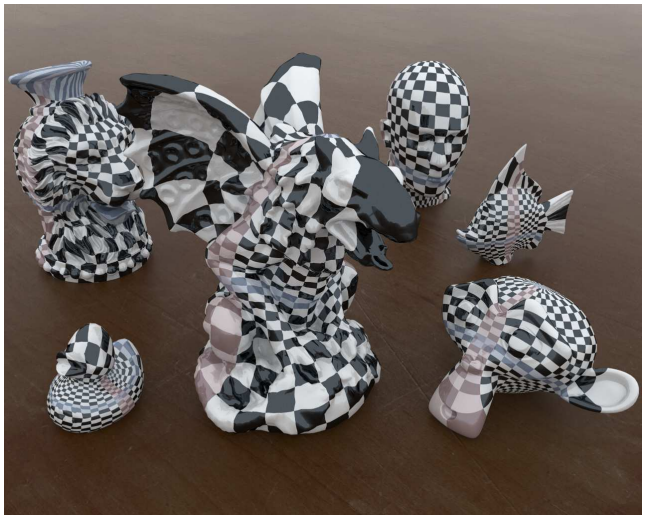


Figure 13: Texture mapping results for different models from [2],[12].

[20] Garland, M., Willmott, A., Heckbert, P., 2001. Hierarchical face clustering on polygonal surfaces. In: Proceedings of the 2001 symposium on Interactive 3D graphics. I3D '01. ACM, New York, NY, USA, pp. 49–58.

[21] Gotsman, C., Gu, X., Sheffer, A., July 2003. Fundamentals of spherical parameterization for 3d meshes. In: ACM Transactions on Graphics 22. pp. 358–363.

[22] Gu, X., Yau, S. T., 2003. Global conformal surface parameterization. In: Proceedings of the 2003 Eurographics/ACM SIGGRAPH symposium on Geometry processing. SGP '03. Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, pp. 127–137.

[23] Haker, S., Angenent, S., Tannenbaum, A., Kikinis, R., Sapiro, G., Halle, M., April 2000. Conformal surface parameterization for texture mapping. IEEE Transactions on Visualization and Computer Graphics 6, 181–189.

[24] Inoue, K., Itoh, T., Yamada, A., Furuhata, T., Shimada, K., 2001. Face clustering of a large-scale cad model for surface mesh generation. Computer-Aided Design 33 (3), 251 – 261.

[25] Isenburg, M., Gumhold, S., Gotsman, C., 2001. Connectivity shapes. In: In IEEE Visualization 2001 Conference Proceedings (2001. pp. 135–142.

[26] Katz, S., Leifman, G., Tal, A., 2005. Mesh segmentation using feature point and core extraction. The Visual Computer 21 (8-10), 649–658.

[27] Katz, S., Tal, A., July 2003. Hierarchical mesh decomposition using fuzzy clustering and cuts. ACM Trans. Graph. 22, 954–961.

[28] Koschan, A. F., 2003. Perception-based 3d triangle mesh segmentation using fast marching watersheds. In: in Proceedings of the International Conference on Computer Vision and Pattern Recognition, II. pp. 27–32.

[29] Lee, Y., Lee, S., Shamir, A., Cohen-Or, D., Seidel, H., 2004. Intelligent mesh scissoring using 3d snakes. In: Proceedings of the Computer Graphics and Applications, 12th Pacific Conference. PG '04. IEEE Computer Society, Washington, DC, USA, pp. 279–287.

[30] Leonardis, A., Jaklič, A., Solina, F., November 1997. Superquadrics for segmenting and modeling range data. IEEE Trans. Pattern Anal. Mach. Intell. 19, 1289–1295.

[31] Lévy, B., Petitjean, S., Ray, N., Maillot, J., July 2002. Least squares conformal maps for automatic texture atlas generation. ACM Trans. Graph. 21, 362–371.

[32] Li, X., Woon, T., Tan, T., Huang, Z., 2001. Decomposing polygon meshes for interactive applications. In: Proceedings of the 2001 symposium on Interactive 3D graphics. I3D '01. ACM, New York, NY, USA, pp. 35–42.

[33] Lien, J., Amato, N. M., 2005. Approximate convex decomposition of polyhedra. Tech. rep., In Proc. of ACM Symposium on Solid and Physical Modeling.

[34] Lien, J. M., Keyser, J., Amato, N. M., 2006. Simultaneous shape decomposition and skeletonization. In: Proceedings of the 2006 ACM symposium on Solid and physical modeling. SPM '06. ACM, New York, NY, USA, pp. 219–228.

[35] Mortara, M., Patane, G., Spagnuolo, M., Falcidieno, B., Rossignac, J., October 2003. Blowing bubbles for multi-scale analysis and decomposition of triangle meshes. Algorithmica 38, 227–248.

[36] NVIDIA, July 2009. Opencl best practices guide.

[37] Pichler, A., amd M. Vincze, R. F., 2004. Decomposition of range images using markov random fields. ICIP, 1205–1208.

[38] Praun, E., Hoppe, H., 2003. Spherical parameterization and remeshing. In: SIGGRAPH '03: ACM SIGGRAPH 2003 Papers. ACM, New York, NY, USA, pp. 340–349.

[39] Saba, S., Yavneh, I., Gotsman, C., Sheffer, A., 2005. Practical spherical embedding of manifold triangle meshes. In: Proceedings of the International Conference on Shape Modeling and Applications 2005. pp. 258–267.

[40] Sander, P. V., Nehab, D., Barczak, J., 2007. Fast triangle reordering for vertex locality and reduced overdraw. In: ACM SIGGRAPH 2007 papers. SIGGRAPH '07. ACM, New York, NY, USA.

[41] Sapidis, N., Besl, P., April 1995. Direct construction of polynomial surfaces from dense range images through region growing. ACM Trans. Graph. 14, 171–200.

[42] Sellamani, S., Muthuganapathy, R., Kalyanaraman, Y., Murugappan, S., Goyal, M., Ramani, K., Hoffman, C. M., 2010. Pcs: Prominent cross-sections for mesh models. Computer Aided Design and Applications 7, 601–620.

[43] Sheffer, A., Gotsman, C., Dyn, N., 2004. Robust spherical parameterization of triangular meshes. Computing 72 (1-2), 185–193.

[44] Stamati, V., Fudos, I., Beijing, China 2007. A feature-based approach to re-engineering objects of freeform design by exploiting point cloud morphology. In: Proc. SPM 2007. ACM.

[45] Tutte, W. T., 1963. How to draw a graph. Proc. London Math. Soc 13, 743–768.

[46] Várady, T., Facello, M., Terék, Z., May 2007. Automatic extraction of surface structures in digital shape reconstruction. Comput. Aided Des. 39, 379–388.

[47] Wu, K., Levine, M., November 1997. 3d part segmentation using simulated electrical charge distributions. Pattern Analysis and Machine Intelligence, IEEE Transactions on 19 (11), 1223 –1235.

[48] Yoshizawa, S., Belyaev, A., peter Seidel, H., 2005. A moving mesh approach to stretch-minimizing mesh parameterization. International Journal of Shape Modeling 11, 25–42.

[49] Zhang, Y., Paik, J., Koschan, A., Abidi, M. A., 2002. A simple and efficient algorithm for part decomposition of 3d triangulated models based on curvature analysis. In: in Proceedings of the International Conference on Image Processing, III. pp. 273–276.

[50] Zuckerberger, E., Tal, A., Shlafman, S., 2002. Polyhedral surface decomposition with applications. Computers & Graphics 26 (5), 733 – 743.

Appendix A. Demonstration Video (for online publication only)

Figure A.14: Demonstration video illustating how segmentation is performed using the area stretch factor derived by our spherical parameterization method.