Texture Mapping via Texture-Adaptive Spherical Multi-Dimensional Scaling

Y. Keller[1] and R. Kimmel[2]

[1]Math department

Yale University, Connecticut, USA

Phone: +1-203-432-4345

Email: yosi.keller@yale.edu

[2]Computer Science Department

Technion–Israel Institute of Technology, Haifa 32000, Israel

Abstract

We present a technique for texture mapping arbitrary sphere-like surfaces with minimal distortions by spherical embedding. The embedding is computed using spherical multi-dimensional scaling (MDS). MDS is a family of methods that map a set of points into a finite dimensional domain by minimizing the difference in distances between every pair of points in the original and the new embedding domains. In this paper spherical embedding is derived using geodesic

*This work was supported by a grant from the Ministry of Science, Israel.

distances on triangulated domains, computed by the fast marching method. The MDS is formulated as a non-linear optimization problem and a fast multi-resolution solution is derived. Finally, we show that the embedding of complex objects which are not sphere-like, can be improved by defining a texture dependent scale factor. This scale is the maximal distance to be preserved by the embedding and can be estimated using spherical harmonics. Experimental results show the benefits of the proposed approach.

1 Introduction

Texture mapping is a fundamental technique in computer graphics where an image is mapped onto a given surface. This problem is closely related to the embedding of high dimensional data in a low dimensional space, such that a certain distortion measure is minimized. As texture maps are mostly defined on a plane or a sphere, the problem can be viewed as the embedding of curved surface onto a two dimensional flat or spherical space. Two coordinates are assigned to each of the original mesh vertices, a procedure also known as parametrization. The texture pixels are then mapped from the texture image (parametrization surface) to the faces of the triangle in 3D, and the embedding errors are perceived as visual artifacts. In addition, meshes mapped to a plane can be compressed using traditional wavelets based image compression schemes [1].

This topic was extensively studied in the computer graphics literature. Arad and Elber [2] preserve the local area of textures by finding, for a specific viewing direction, the four intersecting curves (in the parametric space) between a swept rectangle in the viewing direction and the surface. Then, they warp the square texture image to fit the four curves. This method is useful when the texture is mapped on a small region of the surface.

Kurzion et. al. [3] tried to preserve area by using a cube as a simple surrounding object. For each point they find two curvature values in specially selected directions, and then change the density of the surrounding image accordingly. This method is area preserving, however, it creates shear effects and is limited to C^2 continuous surfaces.

Environment mapping [4, 5] creates the effect of environment reflections on surfaces. It maps the original 2D texture to a sphere or a cube surrounding the surface. Then, the surface normal at each point is used to find the intersection of the reflected viewing vector with the surrounding simple object, and assigns the texture at that point to the corresponding surface point. These methods do not preserve the local area of the texture and introduce local deformations.

An extension to environmental mapping was given in [6] using a two step procedure. First, the texture is mapped onto a simple object. Then, it is mapped from that object to the given surface, using the surface normal's intersection with the simple object. This method also introduces visible deformations, however, it can reduces the distortions caused by the previous methods.

Bennis [7] piecewise flattens the surface and then maps the texture onto each flattened region. The flattening of a region evolves around an isoparametric curve selected manually. The growth is stopped when the accumulated distortion exceeds a given threshold. Discontinuities of the mapped texture are permitted in order to minimize distortions.

Floater [8, 9], presented a 'shape preserving' texture mapping. First, he limits the triangulated surface boundary vertices to specific coordinates of a convex polygon in R^2. Then he solves a set of linear equations that force all other interior vertices to be a convex combination of their neighbors. It is shown that there are no self intersections of triangles on the flat texture plane. Since the boundary texture coordinates are fixed, some deformations would occur, especially global ones.

Levy and Mallet [10] extended Floater's ideas by adding flexibility to the scheme. Instead of

solving a set of linear equations, a global criterion is minimized in the least square sense. This allows adding new measures in addition to Floater's linear equations. These measures preserve the perpendicularity and constant spacing of isoparametric curves traced on the surface.

In [11], Azariadis and Aspragathos proposed to minimize a functional that combines a dissimilarity measure for neighboring vertices and an area measure for the flattened triangles. They also restricted two curves in their mapping to have identical lengths as two selected curves on the surface. This constraint can be considered a boundary condition for their scheme.

Neyret and Cani [12] dealt with general surface topology by tiling together small textured patches with matching boundaries. Their method is limited to textures with relatively small details, as the tiles should be relatively small. A similar solution was introduced in [13], where features are detected in a small texture patch, and are repeatedly pasted onto any given surface until it is completely covered. These methods are impose some non-trivial difficulties for mapping an image onto a curved domain.

The connection between spherical parametrization problem and spectral graph theory was established in [14], which describes a generalization of the method of barycentric coordinates for planar parametrization. The problem of computing global conformal structures for general closed meshes was addressed in [15], where the proposed method approximates De Rham cohomology by simplicial cohomology, and computes a basis of holomorphic one-forms. It is generalized for surfaces with boundaries and there is no restriction of the geometric realization of the homology basis. Another approach to conformal mapping which preserves the angles of the mesh triangles was proposed by Haker et al. [16] to embed a closed surface onto a sphere.

Sheffer and de Sturler [17] concentrate on preserving the angles of the mesh while mapping it onto the 2D plane. The mapping is defined in terms of the angles only, and an optimal solution is

proven to exist. However, these methods still impose high distortion on highly curved surfaces and may even lead to self-intersections.

To cope with the distortion problem, the input mesh can be partitioned into several parts and each part will be mapped to a sphere. Thus, a set of embeddings is derived and a seam construction algorithm is needed to provide a continuous mapping within the set of embeddings. Such approaches were first applied to planar embeddings. In [18] and [19] the mesh is partitioned by computing a coarse base mesh, where each triangle in the base mesh defines a parametrization for a corresponding cluster of triangles in the input mesh. The embedding is then computed by harmonic maps. Similarly, in [20], the mesh was clustered according to the similarity in the directions of normals and of maximal curvatures. Each region is then embedded on a plane.

For spherical embeddings, Sheffer [21] introduced seams into the surface, computed by a minimal-spanning-tree algorithm. Since cutting the surface at the regions of high curvature reduces the Gaussian curvature, the seams improve the overall quality of the mapping. The self-intersections are detected in a post-process, and the parametrization needs to be recomputed to eliminate them.

The pioneering papers [22] and [23], are closely related to the algorithm presented here. First, geodesic distances between pairs of points on the surface are computed, using a computationally intensive scheme. Then, the MDS (Multi-Dimensional Scaling) is applied to flatten the surface using the geodesic distances.

Related approaches to planar embedding were more recently applied to volumetric and triangulated data in [24] and [25], respectively. They start by computing the geodesic distances on volumetric ([24]) or triangulated data ([25]), and then apply classical scaling to D, the matrix of distances between each pair of vertices on the mesh. The planar embedding is given by the first

two eigenvectors and eigenvalues of the double centered D.

Such an embedding is also applicable to unfolding of the curved and convoluted outer surface of the brain (known as the cortex or cortical surface) [26]. The 3D structure of the cortex (a mesh) is mapped onto a sphere or a plane, and the embedded representation can then be used to compare inter-patient neuroimaging data and visualize it using texture mapping.

In this paper, we present a spherical texture mapping scheme based on Multi-Dimensional Scaling (MDS) [27, 28, 29], which minimizes the difference in geodesic distances between corresponding vertices on the mesh and their embedding on the sphere. Formally, denote by $D_G(i,j)$ the geodesic distance between two vertices in the input mesh and $D_G^E(i,j)$ the geodesic distance between their corresponding dual points on a sphere.

The MDS aims to compute an embedding E that minimizes the embedding error

$$\varepsilon = \sum_{i,j}^{n} w_{ij}(D_G(i,j) - D_G^E(i,j))^2 \tag{1.1}$$

where $w_{ij} \geq 0$ and n is the number of vertices in the mesh.

The geodesic distances on the curved surface are efficiently computed using the fast marching method on triangulated domains [30]. Spherical mapping is more computationally demanding than planar mapping. Yet, for sphere-like surfaces such as faces and the brain cortex, it yields lower embedding errors.

The unweighted MDS $(w_{ij} = 1)$ preserves both the 'local' (small $D_G(i,j)$ values) as well as the 'global' (large $D_G(i,j)$ values) structure of the texture and does not require boundary conditions, while most of the previous schemes require them as they integrate local measures to preserve the global structure.

A preliminary version of the above scheme was used in [31] to texture map the Cortex. Next we extend these results by providing a faster scheme that better handles the embedding of meshes which are not sphere-like.

The embedding error of such meshes is large, yielding poor visual quality. We show that by adaptively setting the weights w_{ij}, the texture mapping quality is improved and that such a technique is related to *local regression* [32]. The weighted MDS allows us to specify a certain range of distances for which the texture structure should be preserved. This range can be determined for a given texture image using spherical harmonics [33, 34]. Finally, we present an efficient multi-resolution numerical scheme for the solution of Eq. (1.1) in both weighted and the unweighted cases and show its applicability.

We note that our scheme is conceptually related to the schwartz et al. dimensionality reduction algorithm [23] (that was recently popularized under the name of ISOMAP [35]). In our case the restriction of the distance on a sphere (rather than a plane) leads to better preservation of the original geodesic distances.

The outline of this paper is as follows: preliminaries are given in Sections 2 and 3 which provide a brief review of fast marching on triangulated domains and distance computations on spheres, respectively. Section 4 presents the proposed spherical MDS algorithm and the texture adaptive formulation is given in Section 5. These are experimentally verified in Section 6. Concluding remarks are given in Section 7.

2 Fast Marching Method on Triangulated Domains

The first step of the embedding procedure is to compute the geodesic distances between pairs of points on the surface. The fast marching method (FMM), introduced by Sethian [36] is a numerically consistent distance computation approach that is applicable to rectangular grids and was extended to triangulated domains by Kimmel and Sethian in [30].

The basic idea is to numerically solve a wave propagation problem given by an Eikonal equation , where at the source point the distance is known to be zero. The distance function is iteratively constructed by patching together small planes supported by neighboring grid points with gradient magnitude equal to one.

The distance function is constructed by starting from the sources point and propagating outwards. Applying the method to triangulated domains requires a careful analysis of the update of a single vertex in a triangle, when the distance function at the other vertices is given. The FMM on triangulated domains computes the geodesic distances between a single vertex and the rest of the n surface vertices in $O(n)$ operations. Repeating this computation for each vertex, we compute all the geodesic distances $D_G(i,j)$ $\{1 \leq i \leq n,\ j < i\}$ in $O(n^2)$ operations.

Thus, the essence of the FMM, is its low computational complexity as the distance from the source point gets larger. Note that, if the numerical grid given by triangles is pre-processed properly, that is, obtuse angles are subdivided by virtual edges [30], then the geodesic distance computation is accurate (first order) and the whole scheme is consistent.

3 Spherical geometry

Spherical MDS schemes map vertices onto a surface of a unit sphere. Points on a surface of the sphere are parameterized by a vector of spherical angles $\theta = \begin{bmatrix} \theta_1 & \theta_2 \end{bmatrix}^T$, where each point l is given by the coordinates $\left\{\theta_1^l, \theta_2^l\right\}$.

Let θ_1^l $\left(-\frac{\pi}{2} \leq \theta_1^l \leq \frac{\pi}{2}\right)$ and θ_2^l $\left(0 \leq \theta_1^l \leq 2\pi\right)$ be the spherical angles, such that

$$\begin{aligned} x_l &= \cos\theta_1^l \sin\theta_2^l \\ y_l &= \sin\theta_1^l \sin\theta_2^l \\ z_l &= \cos\theta_2^l. \end{aligned} \tag{3.1}$$

Then, the Euclidean distance d_{ij}^e between points on the sphere is given by

$$\left(d_{ij}^e\right)^2 = \left(x_i - x_j\right)^2 + \left(y_i - y_j\right)^2 + \left(z_i - z_j\right)^2. \tag{3.2}$$

Substituting Eq. (3.1) we have that

$$\left(d_{ij}^e\right)^2 = 2 - 2\sin\theta_2^i \sin\theta_2^j \cos\left(\theta_1^i - \theta_1^j\right) - 2\cos\theta_2^i \cos\theta_2^j \tag{3.3}$$

and by applying the Low of Cosines to the triangle depicted in Fig. 1 we get

$$\left(d_{ij}^e\right)^2 = 2R^2 - 2R\cos\varphi, \tag{3.4}$$

where φ is the planar angle and for the unit sphere $R = 1$.

$d^g_{ij}(i,j)$, the geodesic distance on the sphere is then given by

$$d^g_{ij}(i,j) = \varphi = \arccos \frac{2 - \left(d^e_{ij}\right)^2}{2}. \tag{3.5}$$

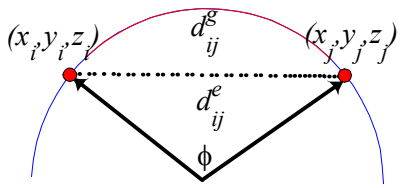


Figure 1: Spherical geometry. The geodesic distance on the surface of the sphere is given by the length of the arc corresponding to the angle φ.

4 Spherical MDS formulation

This Section applies the least squares MDS formulation to the spherical embedding problem. First we compute the geodesic distances between the vertices of the input mesh using the FMM described in Section 2. The spherical MDS is formulated in Section 4.1 and an iterative optimization scheme is presented. Further improvement is achieved by embedding the above procedure in a multi-resolution scheme given in Section 4.2.

4.1 The spherical MDS formulation

For the spherical MDS, $D^E_G(i,j)$, the distance between the embedded vertices is the geodesic distance on the sphere given by Eq. (3.5). The spherical MDS is derived by substituting $d^g_{ij}(\theta^i, \theta^j)$, into Eq. (1.1) and solving for the spherical embedding parameters θ

$$\theta = \arg\min_{\theta} \sum_{i,j} w_{ij} (d^g_{ij}\left(\theta^i, \theta^j\right) - D_G\left(i,j\right))^2. \tag{4.1}$$

The minimization is computed using steepest-decent [37]. Note that $D_G\left(i,j\right)$, the geodesic distance on the mesh, remains fixed. Let ε be the embedding error

$$\varepsilon = \sum_{i,j} w_{ij} (d^g_{ij}\left(\theta^i, \theta^j\right) - D_G\left(i,j\right))^2 = \sum_{i,j} w_{ij} \varepsilon^2_{ij}, \tag{4.2}$$

the steepest-decent iteration is given by

$$\theta^{k+1} = \theta^k - \lambda \frac{\partial \varepsilon}{\partial \theta}, \ k = 0.. \tag{4.3}$$

where $\frac{\partial \varepsilon}{\partial \theta} = \begin{bmatrix} \frac{\partial \varepsilon}{\partial \theta_1} & \frac{\partial \varepsilon}{\partial \theta_2} \end{bmatrix}^T$, λ is the step size discussed in Section 4.1.1 and θ^0 is given as input. ε^k denotes the embedding error after iteration k.

The partial derivatives are computed using the chain rule

$$\begin{aligned} \frac{\partial \varepsilon}{\partial \theta^l_m} &= \sum_{j=1}^{n} w_{ij} 2 (d^g_{lj}\left(\theta^i, \theta^j\right) - D_G\left(l,j\right)) \frac{\partial d^g_{ij}\left(\theta^i, \theta^j\right)}{\partial \theta_i} \\ &= \sum_{j=1}^{n} w_{ij} 2 \varepsilon_{il} \frac{\partial d^g_{ij}\left(\theta^i, \theta^j\right)}{\partial \theta^i_m}, m = 1,2, \ l = 1..n. \end{aligned} \tag{4.4}$$

By applying the chain rule again and substituting Eqs. (3.3) and (3.5) we get

$$\frac{\partial d^g_{ij}\left(\theta^i, \theta^j\right)}{\partial \theta^i_m} = \frac{\partial d^g_{ij}\left(\theta^i, \theta^j\right)}{\partial d^e_{il}} \frac{\partial d^e_{il}}{\partial \theta^i_m}$$

where

$$\frac{\partial d_{ij}^{g}\left(\theta^{i},\theta^{j}\right)}{\partial d_{il}^{e}}=\frac{\partial d_{ij}^{g}\left(\theta^{i},\theta^{j}\right)}{\partial\left(d_{il}^{e}\right)^{2}}2d_{il}^{e}=\frac{-2d_{il}^{e}}{\sqrt{1-\left(\frac{2-\left(d_{ij}^{e}\right)^{2}}{2}\right)^{2}}}$$

and

$$\begin{aligned}\frac{\partial d_{il}^{e}}{\partial\theta_{1}^{l}}&=-\frac{1}{d_{ij}^{e}}\sin\theta_{2}^{i}\sin\theta_{2}^{j}\sin\left(\theta_{1}^{i}-\theta_{1}^{l}\right)\\ \frac{\partial d_{il}^{e}}{\partial\theta_{2}^{l}}&=\frac{1}{d_{ij}^{e}}\left(-\cos\theta_{2}^{i}\sin\theta_{2}^{j}\cos\left(\theta_{1}^{i}-\theta_{1}^{l}\right)+\sin\theta_{2}^{i}\cos\theta_{2}^{l}\right)\end{aligned}\tag{4.5}$$

Equation (4.3) is reiterated until at most $N_{\max}$ iterations are performed or the decrease of the embedding error $|\varepsilon_k-\varepsilon_{k-1}|$ is less than a predetermined threshold.

4.1.1 Line search

An appropriate choice of the iterative step-size λ used in Eq. (4.3) is critical for the convergence properties of the steepest-decent scheme. Setting a low value for λ would result in slow convergence, while setting it too high, may cause the algorithm to diverge. A possible solution is to use a line search [37] to find the optimal value of λ that minimizes ε^k given $\frac{\partial\varepsilon}{\partial\theta}$.

Thus, at each iteration k,

1. Start by setting $\lambda=1$.
2. Compute the updated solution θ^k using Eq. (4.3) and the embedding error ε^k.
3. If $\varepsilon^k<\varepsilon^{k-1}$, set $\lambda=2\lambda$, $\varepsilon^{k-1}=\varepsilon^k$ and go to Step 2.
4. If $\varepsilon^k>\varepsilon^{k-1}$, set $\lambda=\frac{\lambda}{2}$, $\varepsilon^{k-1}=\varepsilon^k$.

5. If the current value of λ was already used then quit, else, go to Step 2.

Due to the symmetry of the geodesic distances $d_{ij}^g\left(\theta^i,\theta^j\right)=d_{ij}^g\left(\theta^j,\theta^i\right)$, the evaluation of Eq. (4.2) can be reduced to

$$\varepsilon=\sum_{i=1}^{n}\sum_{j=1}^{j<i}w_{ij}(d_{ij}^g\left(\theta^i,\theta^j\right)-D_G\left(i,j\right))^2,$$

and the computation of the embedding error is faster than the evaluation of the embedding error's derivative in Eq. (4.4).

4.2 Fast multi-resolution spherical MDS

Multi-resolution optimization techniques are widely used in computer vision [38] and are usually computationally more efficient than single resolution schemes, as they improve the accuracy and numerical conditioning. Denote $\{0\leq s\leq s_{\max}\}$ the scale of the mesh, where $s=0$ corresponds to the finest resolution and $s_{\max}$ is predefined. Then V_s, E_s and D_s are the mesh, its embedding and the distance matrix in a scale s, respectively.

Next, we define downscaling and upscaling procedures that relate $\{M_s,E_s,D_s\}$ to $\{M_{s+1},E_{s+1},D_{s+1}\}$ The downscaling is implemented using the mesh decimation scheme given in [39]. Thus, the embedding E_{s+1} is computed by copying the embedding of each vertex $V_i\in M_{s+1}$ from E_s, and D_{s+1} is a decimated replica of D_s where any row and column i such that $V_i\notin M_{s+1}$ are removed.

The upscaling is given by copying the embedding in E_{s+1} of the vertices $V_i\in M_{s+1}$, to the corresponding entries in E_s. The initial embedding of each vertex $V_i\notin M_{s+1}$ is approximated by using the embedding of the vertex $V_i\in M_{s+1}$ closest to it. The upscaled embedding is then used to initialize θ^0 in the iterative scheme given in Section 4.1.

The solver is randomly initialized in the coarsest scale. In general, this makes the scheme robust to local minima. As we did not encounter such cases in our tests, the solver can also be initialized by projecting the vertices on the unit sphere by computing their spherical coordinates and setting $r = 1$.

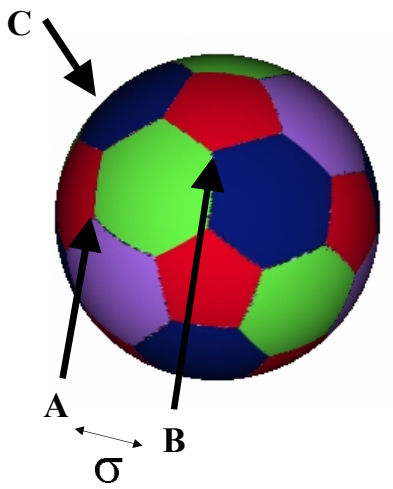
The computation is performed from coarse to fine. The result is refined in the finest resolution level, the global minimum is achieved. The downscaling and upscaling of E_s and D_s use only data structure operations and no numerical computations, making them fast and suitable for the embedding of large meshes.

5 Weighted MDS

For the particular application of texture mapping, the spherical embedding can be modified to improve the visual quality. The embedding error of meshes which are not sphere-like is large, yielding poor visual quality. Preserving both the local and global structure for such meshes is sometimes impossible. A possible solution is to partition the mesh and map each part separately. Such approaches were discussed in Section 1. The main problem there, is the seamless integration of the partial embeddings into a single continuous mapping. Hence, we are motivated to extend the current scheme to better handle complex meshes. It can also be used as a component of a partitioning based scheme.

For textures such as the soccer ball shown in Fig. 2, we denote σ, the intrinsic scale of the texture image. σ is the largest distance that should be preserved by the embedding, while larger distances can be distorted. For example, for the soccer ball, σ is the width of the pentagon.

This approach is related to local regression [32], where functions are estimated locally by



(a)

Figure 2: Deriving the intrinsic scale of a texture. In such a texture, it suffices to retain the distances between the points A and B which is the intrinsic scale σ. The distances AC and AB are visually less significant.

polynomials and the local scale is given by the scale parameter σ. Following the local regression formulations in [32] the weights were computed using the Nadaraya-Watson kernel

$$K_\sigma(x) = D\left(\frac{|x|}{\sigma}\right) \tag{5.1}$$

with

$$D(t) = \begin{cases} \frac{3}{4}(1-t^2) & \text{if } |t| \leq 1 \\ 0 & \text{otherwise.} \end{cases} \tag{5.2}$$

Equation (4.2) is reduced to

$$\varepsilon = \sum_{i,j} w_{ij}\varepsilon_{ij}^2 = \sum_{|D_G(i,j)|<\sigma} K_\sigma\left(D_G(i,j)\right)\varepsilon_{ij}^2 \tag{5.3}$$

and the iterative refinement term $\frac{\partial \varepsilon}{\partial \theta^l}$ for each vertex i becomes

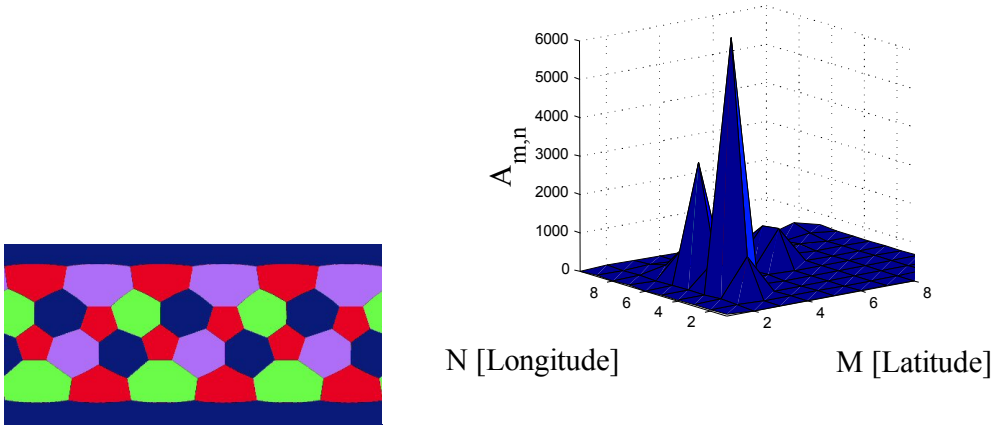
$$\frac{\partial \varepsilon}{\partial \theta^l} = \sum_{|D_G(i,j)|<\sigma} 2K_\sigma \left(D_G\left(i,j\right)\right) \frac{\partial d_{ij}^g \left(\theta^i, \theta^j\right)}{\partial \theta_m^l} \varepsilon_{ij}. \tag{5.4}$$

Note that by using the weighted MDS (WMDS), the scheme preserves only the distances $\left|D_G\left(\theta^i, \theta^j\right)\right| < \sigma$. Thus, there is no need to store all of the relative distances and compared to the unweighted MDS and [25] the matrix of distances becomes sparse. This makes the weighted MDS more suitable for embedding large meshes.

5.1 Intrinsic scale estimation

For a given spherical texture map the intrinsic scale can be estimated using spherical harmonics. This is a set of basis function defined on the surface of the sphere analogous to Fourier analysis. Spherical harmonics have been used in graphics to efficiently represent the bidirectional reflection distribution function (BRDF) of different materials [33, 34]. Thus by computing the spherical harmonics coefficients, one can estimate the dominant spherical frequencies corresponding to the intrinsic scale σ. Such an example is given in Fig. 3 where the spherical harmonics coefficients of Fig. 3a are given in Fig. 3b.

We note that the Weighted MDS can also be applied to planar least squares MDS, where the intrinsic scale σ can be estimated by the Fourier transform of a given texture map and detecting the dominant frequencies.



(a) (b)

Figure 3: Computing the intrinsic scale of a texture using spherical harmonics. The spherical texture map used in Fig. 2 is given in (a) in spherical coordinates. The spherical harmonics coefficients are shown in (b). The maximum is detected in $(m = 2, n = 4)$, corresponding to 2 cycles in the latitude (vertical axis of (a)) and 4 cycles in the longtitude (horizontal axis of (a)).

6 Experimental Results

The proposed technique was tested using several meshes. We start by comparing the proposed scheme to the MDS based planar embedding given in [25]. The 'face' mesh (6000 vertices) has a half-sphere-like topology and the 'David' mesh (10000 vertices) is used to compare the embeddings of a full-sphere-like object. We show the applicability of the scheme to medical visualization by annotating and texture mapping the 'cortex' mesh (6000 vertices) and a segment of it. Finally, we apply the WMDS to texture mapping the 'Stanford bunny' (6000 vertices) and show the improvement compared to the regular MDS.

Given θ^i (the spherical embedding of a vertex V_i), the color associated to V_i was determined by sampling the spherical texture image I at θ^i. Subpixel values of θ^i were handled by bilinear interpolation of I. The visualizations of the texture mapped meshes were produced by the VTK mesh viewer [40]. The multi-resolution MDS scheme was used in all of the simulations, with

three resolution scales. At each scale the computation continued until $\Delta\varepsilon$, the reduction of the embedding error, became less than 10^{-7}.

(a) Spherical MDS (b) Cartesian MDS

Figure 4: The Spherical and Planar MDS applied to the 'face' mesh (6000 vertices). Note the embedding error of the cartesian MDS around the region of the nose.

Name	Vertices#	Embedding type	Embedding error
Face	3000	spherical	$3.1 \cdot 10^{-5}$
Face	3000	cartesian	$3.4 \cdot 10^{-5}$
Face	6000	spherical	$1.15 \cdot 10^{-5}$
Face	6000	cartesian	$1.25 \cdot 10^{-5}$

Table 1: Embedding error comparison between cartesian and spherical embeddings.

Figure 4 and Table 1 show the visualizations artifacts and the embedding error of the 'face' mesh, respectively. This is a half-sphere-like object with low curvature, except for the nose area which is a high curvature structure. The embedding was computed using the regular (unweighted) MDS. Figures 4a and 4b, show similar visual quality, except for the nose area, where the planar MDS shows a larger distortion. Thus, both embedding have a similar average error.

A significant improvement is evident in the embedding of the full-sphere-like 'David' mesh given in Fig. 5. The planar embedding results in significant embedding error, while the spherical MDS gives reasonable results.

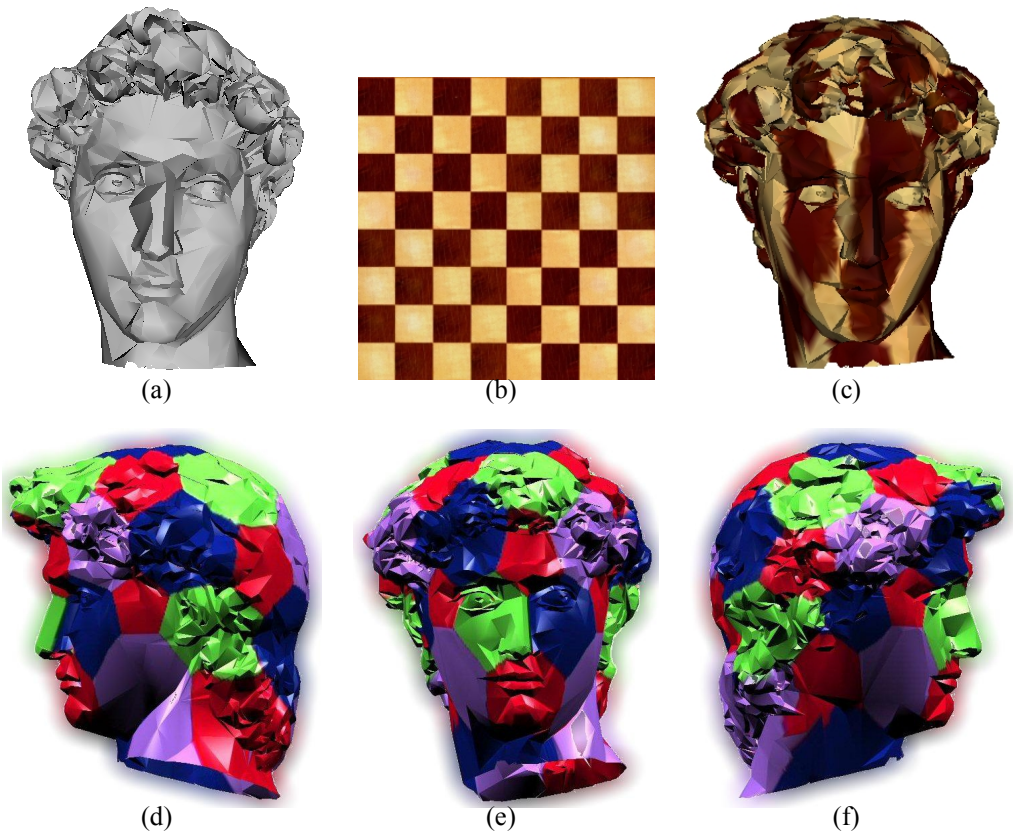


Figure 5: Texture mapping of a sphere-like mesh. (a) The head of the David mesh (1000 vertices) . (b) The planar chess texture used for the planar texture mapping in (c). (c) The results of a LSMDS based planar texture mapping. (d), (e) and (f) Texture mapping results using the spherical MDS.

Figures 6 and 7 apply the spherical MDS to the unfolding of the brain cortex for medical visualization. We were able to annotate the cortex in Fig. 6 and color it in Fig. 7. Note the lack of geometrical distortions in these figures.

Finally, we compare the results of the spherical and weighted spherical MDS using the 'Stanford bunny'. This object is far from a sphere and has points of negative and positive mean curvature. Texture mapping it using the spherical MDS, results in significant embedding errors, as depicted in Figs. 8a and 8b. There is no embedding that minimizes the difference in both the local and global distances. The results of the WMDS applied with $\sigma = 0.1$ are demonstrated in Figs. 8c and 8d, that depict the improved texture mapping. Fig. 8e depicts the texture mapping of the

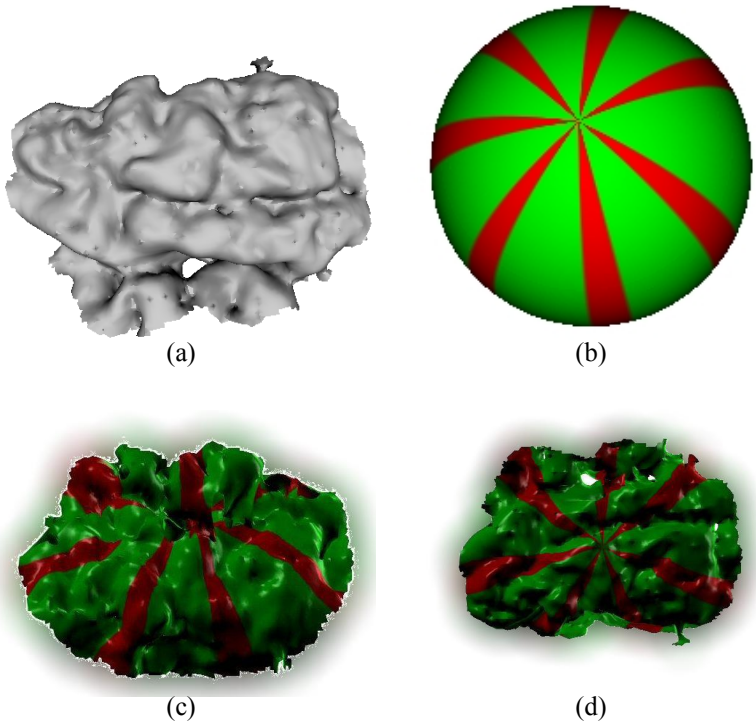


Figure 6: Annotation of parts of the Cortex.

earth's texture.

These results asses the validity of the mapping (bijectivity). For patches with high isoperimetric distortion (sphere-like shapes), classical geodesic-MDS often gives wrong results. This is not evident in our results. Note that the measure we optimize is a global one. The triangulation is a particular numerical representation of the geometry. While mapping a sphere-like surface onto a plane one could experience face flips. However, choosing the sphere as a target simplified map, such effects are reduced drastically. Using least squares scaling one could restrict the triangles to preserve their orientation on the sphere. However, we did not find such a restriction a necessity in our applications.

The timing results for the single and multiscale MDS are given table 2. The algorithms were implemented using non-optimized C++ and the multi-scale MDS implementation uses the same

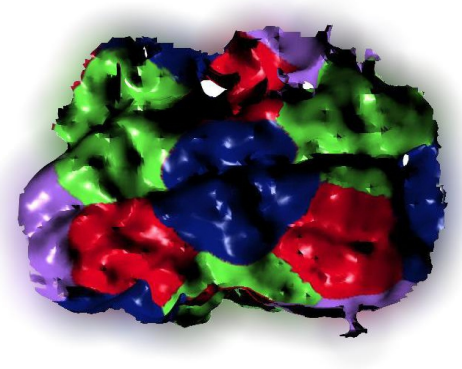


Figure 7: Coloring part of the cortex. Note the accuracy with which the pentagon patterns are preserved.

Mesh size [vertices#]	1000	2000	3000	6000	10000
Single scale	4.4s	11.4s	32.7s	166.8s	420.0s
Multiscale	2.0s	5.2s	25.2s	67.1s	217.5s

Table 2: Timing results for the single scale and multiscale MDS. The multi-scale scheme significantly reduces the computation time.

code as the single scale scheme for the computations within each resolution scale. The 'Stanford bunny' (6000 vertices) was used for the timing measurements on a 2.8GHz PC. Evidently, the multi-scale scheme improved the algorithm's timing by 50-100 percent.

7 Summary

An efficient and accurate method for embedding surfaces onto a sphere was presented. The method is based on the fast marching on triangulated domains algorithm followed by multi-dimensional scaling, and was shown to provide improved visual results compared to planar flattening. Furthermore, we presented a weighted MDS formulation which allows us to better handle objects with non-sphere-like geometry. Finally, we derived a fast multi-scale optimization scheme for the nu-

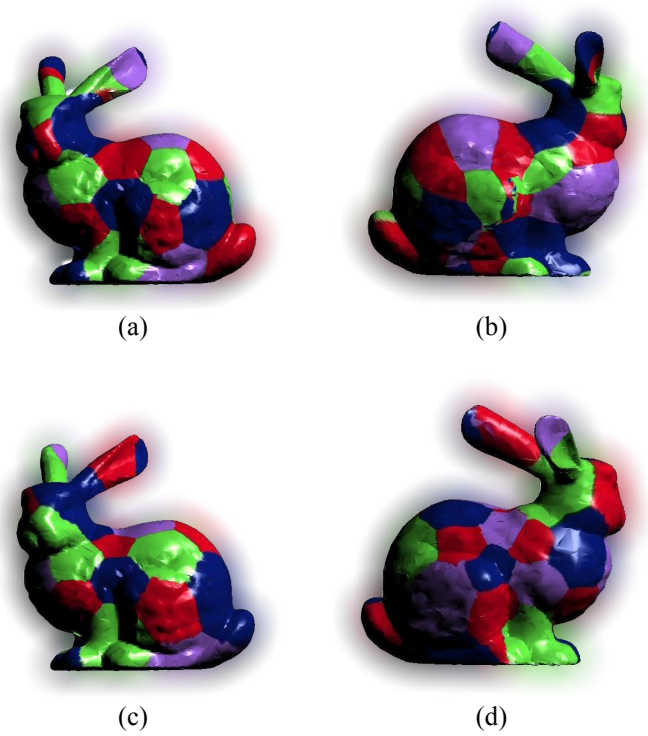


Figure 8: Local spherical MDS texture mapping results. (a) and (b) are the results of the regular spherical MDS while (c) and (d) are both sides of the Stanford bunny textured mapped using the local spherical MDS. Note the improved mapping in (b).

merical solution of the problem. In future work we will study the use of the proposed scheme in a partitioning based embedding scheme. The main challenge is to device a partitioning scheme that will allow a continuous mapping between the embedded partitions.

References

[1] Hoppe, H., Praun, E.: Shape compression using spherical geometry images, Springer-Verlag (2005) 27–46

[2] Arad, M.: Isometric texture mapping for free-form surfaces. Computer Graphics Forum **16** (1997) 247–256 ISSN 1067-7055.

[3] Kurzion, Y., Moller, T., Yagel, R.: Size preserving pattern mapping. In: Proceedings of the conference on Visualization '98, IEEE Computer Society Press (1998) 367–373

[4] Blinn, J.F., Newell, M.E.: Texture and reection in computer. Comm. of the ACM **19** (1976) 542–547

[5] Greene, N.: Environment mapping and other applications of world projections. IEEE Comput. Graph. Appl. **6** (1986) 21–29

[6] Bier, E.A., Sloan, K.S.: Two-part texture mapping. IEEE Computer Graphics and Applications (1986) 40–53

[7] Bennis, C., Vezien, J.M., Iglesias, G.: Piecewise surface flattening for non-distorted texture mapping. In: Proceedings of the 18th annual conference on Computer graphics and interactive techniques, ACM Press (1991) 237–246

[8] Floater, M.S.: Parametrization and smooth approximation of surface triangulations. Computer Aided Geometric Design **14** (1997) 231–250

[9] Floater, M.S.: Parametric tilings and scattered data approximation. International Journal of Shape Modeling **4** (1998) 165–182

[10] Levy, B., Mallet, J.L.: Non-distorted texture mapping for sheared triangulated meshes. In Proc. of Compuer Graphics, Annual Conference Series (1998) 343–352

[11] Azariadis, P.N., Aspragathos, N.A.: On using planar developments to perform texture mapping on arbitrarily curved surfaces. Computers and Graphics **24** (2000) 539–554

[12] Neyret, F., Cani, M.P.: Pattern-based texturing revisited. SIGGRAPH (1999) 235–242.

[13] Praun, E., Finkelstein, A., Hoppe, H.: Lapped textures. In: Proceedings of the 27th annual conference on Computer graphics and interactive techniques, ACM Press/Addison-Wesley Publishing Co. (2000) 465–470

[14] Gotsman, C., Gu, X., Sheffer, A.: Fundamentals of spherical parameterization for 3d meshes. ACM Transactions on Graphics (Proc. SIGGRAPH 2003) **22** (2003)

[15] Gu, X., Wang, Y., Chan, T.F., Thompson, P.M., Yau, S.T.: Genus zero surface conformal mapping and its application to brain surface mapping. IEEE Transaction on Medical Imaging **23** (2004)

[16] Haker, S., Angenent, S., Tannenbaum, A., Kikinis, R., Sapiro, G., Halle, M.: Conformal surface parameterization for texture mapping. IEEE Trans. on Visualization and Computer Graphics **6** (2000) 181–189

[17] Sheffer, A., de Sturler, E.: Parameterization of faceted surfaces for meshing using angle based flattening. Engineering with Computers **17** (2001) 326–337

[18] Eck, M., DeRose, T., Duchamp, T., Hoppe, H., Lounsbery, M., Stuetzle, W.: Multiresolution analysis of arbitrary meshes. Computer Graphics **29** (1995) 173–182

[19] Lee, A.W.F., Sweldens, W., Schröder, P., Cowsar, L., Dobkin, D.: MAPS: Multiresolution adaptive parameterization of surfaces. Computer Graphics **32** (1998) 95–104

[20] Maillot, J., Yahia, H., Verroust, A.: Interactive texture mapping. In: Proceedings of the 20th annual conference on Computer graphics and interactive techniques, ACM Press (1993) 27–34

[21] Sheffer, A.: Spanning tree seams for reducing parameterization distortion of triangulated surfaces. In: Proceedings of the Shape Modeling International 2002 (SMI'02), IEEE Computer Society (2002) 61

[22] Wolfson, E., Schwartz, E.L.: Computing minimal distances on polyhedral surfaces. 11 (1989) 1001–1005

[23] Schwartz, E.L., Shaw, A., Wolfson, E.: A numerical solution to the generalized mapmaker's problem: Flattening nonconvex polyhedral surfaces. 11 (1989) 1005–1008

[24] Grossman, R., Kiryati, N., Kimmel, R.: Computational surface flattening: A voxel-based approach. 24 (2002) 433–441

[25] Zigelman, G., Kimmel, R., Kiryati, N.: Texture mapping using surface flattening via MDS. IEEE Trans. on Visualization and Computer Graphics **8** (2002) 198–207

[26] Wandell, B.A., Chial, S., Backus, B.: Visualization and measurements of the cortical surface. Journal of Cognitive Neuroscience (2000)

[27] Cox, M., Cox, T.: Multidimensional Scaling. Chapman and Hall (1994)

[28] Borg, I., Groenen, P.: Modern Multidimensional Scaling - Theory and Applications. Springer (1997)

[29] Kruskal, J.B., Wish, M.: Multidimensional Scaling. Sage (1978)

[30] Kimmel, R., Sethian, J.: Computing geodesic paths on manifolds. Proc. of National Academy of Science **95** (1998) 8431–8435

[31] Elad, A., Kimmel, R.: Spherical flattening of the cortex surface. In Malladi, R., ed.: Geometric Methods in Biomedical Image Processing, Springer (2002) 77–90

[32] Hastie, T., Ribshirani, R., Friedman, J.H.: The elements of statistical learning: data mining, inference and prediction. Springer (2002)

[33] Cabral, B., Max, N., Springmeyer, R.: Bidirectional reflection functions from surface bump maps. In: Proceedings of the 14th annual conference on Computer graphics and interactive techniques, ACM Press (1987) 273–281

[34] Ramamoorthi, R., Hanrahan, P.: A signal-processing framework for inverse rendering. In: Proceedings of the 28th annual conference on Computer graphics and interactive techniques, ACM Press (2001) 117–128

[35] Tenenbaum, J.B., de Silva, V., Langford, J.C.: A global geometric framework for nonlinear dimensionality reduction. Science **290** (2000) 2319–2323

[36] Sethian, J.: A review of the theory, algorithms, and applications of level set method for propagating surfaces. Acta Numerica, Cambridge University Press (1996)

[37] Gill, P.: Practical Optimization. Academic Press (1982)

[38] Mann, S., Picard, R.: Virtual bellows: constructing high quality stills from video, Austin, TX (1994) 363–367

[39] Melax, S.: A simple, fast and effective polygon reduction algorithm. Game Developer Journal (1998)

[40] Schroeder, W., Martin, K., Lorensen, B.: The Visualization Toolkit: An Object-Oriented Approach to 3D Graphics. Prentice Hall (1997)