Efficient Spherical Parametrization Using Progressive Optimization

Shenghua Wan[1], Tengfei Ye[2], Maoqing Li[2], Hongchao Zhang[3], and Xin Li[1,⋆]

[1] School of Electrical Engineering and Computer Science, Louisiana State University (LSU), Baton Rouge, LA 70803, USA
xinli@lsu.edu
[2] Department of Automation, Xiamen University, Xiamen 361005, China
[3] Department of Mathematics, LSU, Baton Rouge, LA 70803, USA

Abstract. Spherical mapping is a key enabling technology in modeling and processing genus-0 close surfaces. A closed genus-0 surface can be seamless parameterized onto a unit sphere. We develop an effective progressive optimization scheme to compute such a parametrization, minimizing a nonlinear energy balancing angle and area distortions. Among all existing state-of-the-art spherical mapping methods, the main advantage of our spherical mapping are two-folded: (1) the algorithm converges very efficiently, therefore it is suitable for handling huge geometric models, and (2) it generates bijective and lowly distorted mapping results.

Keywords: Spherical Parametrization, Hierarchical Optimization.

1 Introduction

Spherical parametrization seeks a bijective map $f : M \to S$ between a closed genus-0 surface M and a unit spherical domain S. For a very wide category of solid models without handles or voids, their boundary surfaces are closed and genus-0. The sphere is a natural parametric domain for these surfaces, on a sphere domain their seamless parametric representations can be constructed directly. A parameterization introducing small metric distortion is desirable. Isometry (preserving both angle and area) is ideal but usually not possible for a generally given M. We therefore seek for a map that minimizes either angle distortion, or area distortion, or a balancing between both of them.

Computing a map on a non-flattened domain, however, is often formulated as a non-linear optimization problem, and cannot be computed efficiently [6]. For example, a harmonic spherical map is conformal [2,7]. The resultant mapping is angle-preserving. However, its area distortion could be very large, especially in the long and thin protrusion regions (e.g. ears of the Stanford bunny). A map balancing angle and area distortions is therefore often desirable. Zayer et al. [11] proposed a Curvilinear Spherical Parametrization which better reduces area-distortion efficiently. Another state-of-the-art spherical mapping algorithm is

⋆ Corresponding author.

S.-M. Hu and R.R. Martin (Eds.): CVM 2012, LNCS 7633, pp. 170–177, 2012.

proposed by Praun and Hoppe [9]. They used the progressive mesh to iteratively optimize the $L2$ stretching energy [10] defined piecewise on the mesh of M. Such a coarse-to-fine solving scheme can effectively overcome the local minima issue existing in most spherical mapping formulations that aim to minimize angle and area distortion together. Inspired by this, we also adopt the progressive simplification and develop a hierarchical optimization scheme. But unlike [9], we utilize the distortion energy [1] which is shown converged to the continuous energy. Furthermore, we develop an effective hierarchical optimization scheme over the mesh (with different resolutions) from both local and global aspects, to improve the mapping efficiency and efficacy significantly.

We present a hierarchical optimization framework for the spherical parametrization problem. Compared with other state-of-the-art spherical mapping algorithms, our method generates a bijective and lowly-distorted mapping, and converges efficiently. Therefore, our algorithm can be applied on large geometric models with complex geometry (e.g. with long branches) robustly.

2 Hierarchical Spherical Parametrization

2.1 Mapping Distortion

The angle distortion per triangle can be measured [4] on the map of each triangle $f_T : T \to t$ by $E_D(T) = \cot\alpha|a|^2 + \cot\beta|b|^2 + \cot\gamma|c|^2$ where T and t are the triangle of mesh M and its image on the parametric sphere S respectively; α, β, γ are the angles in T and a, b, c are the corresponding opposite edge lengths in t. The area distortion can be measured by $E_A(T) = \frac{Area(t)}{Area(T)}$. The integrated (over the area of parameter triangle t) angle and area distortions of the entire spherical parametrization $f : M \to S$ are therefore $\bar{E}_D(M) = \sum_{i=1}^{N_F} E_D(T_i)Area(t_i)$ and $\bar{E}_A(M) = \sum_{i=1}^{N_F} E_A(T_i)Area(t_i)$, where N_F is the number of faces in M. Following the modification proposed by [1], we use formulations in Eqs (1) and (2) for new distortions, which provide upper bounds of the spherical integrals and avoid degeneracy during the optimization:

$$E_D(M) = \sum_{i=1}^{N_F} d_i^{-2} \cdot E_D(T_i)Area(t_i), \tag{1}$$

$$E_A(M) = \sum_{i=1}^{N_F} d_i^{-2} \cdot E_A(T_i)Area(t_i). \tag{2}$$

where d_i is the minimum distance from the origin to triangle t_i. And the objective function is their weighted sum:

$$E = \lambda E_D(M) + \mu E_A(M) \tag{3}$$

where λ and μ are parameters balancing angle and area distortions. Area distortion is a common issue for spherical mapping, leading to under-sampling, especially for the models with long and thin protrusions, which could cause undesirable artifacts in applications. We found that a relatively large weight on area distortion usually provides stable and desirable mappings; hence in our experiments, we set $\lambda = 0.1$ and $\mu = 1.0$ by default.

2.2 Algorithm Overview

The distortion energy introduced in Section 2.1 is nonlinear and nonconvex. Generally, directly optimizing the energy will get trapped in local minima inevitably. We therefore adopt the progressive mesh [3] to simplify the mesh into coarser resolutions and solve the optimization hierarchically while we gradually refine the mesh back to the original tessellation. The progressive scheme is similar to [9], but our optimization is developed differently and is more efficient and effective. Given a genus-0 mesh M with n vertices, we first progressively simplify it to a tetrahedron with 4 vertices, denoted as M^4. We then use M^k to denote the resolution of M with k vertices, and v_i^k to denote the vertex which will split during the refinement process. v_i^k is a vertex on M^k and it splits into v_i^{k+1} and v_{k+1}^{k+1} (suppose the newly inserted vertex is always given the id $k+1$) in the new mesh M^{k+1}. Based on above definitions, the algorithm pipeline is as follows:

1. Simplify M^n to a tetrahedron M^4 using progressive mesh;
2. Map M^4 onto a unit sphere domain S^4,get $f^4 : M^4 \to S^4$;
3. Following the vertex split order that refines M^4 back to M^n, optimize the spherical mapping $f^k : M^k \to S^k$ hierarchically.

2.3 Global Hierarchical Optimization

We progressively refine M^k from $k = 4, \ldots, n$, and during each vertex split $v_i^k \to \{v_i^{k+1}, v_{k+1}^{k+1}\}$, we find a locally optimal spherical position as the image for each of these two vertices v_i^k, v_{k+1}^{k+1} while fixing images of all their one-ring neighboring vertices. After every η (a constant integer) vertex splits, we optimize all these newly placed vertices as well as their neighboring vertices.

Ideally, after each split, we can perform a local optimization on images of v_i^k, v_{k+1}^{k+1}, and all their neighboring vertices until we get to a local optimum. However, this precise local optimization per every vertex split is relatively expensive and sometimes not necessary. Therefore, we only conduct the optimization after a set of vertices are inserted.

2.4 Local Optimization on a Vertex

After the split of a vertex v_i^k, we need to embed the images of the two new vertices v_i^{k+1} and v_{k+1}^{k+1} on the sphere. Here we solve a simple local optimization to determine valid (non-flipped) spherical locations for them. Later, after each η vertex splits, we will perform such local optimizations on new vertices and their neighboring vertices together. When the mapping of a vertex is updated and the objective energy change is bigger than a threshold, its one-ring vertices may need to be optimized again. We propagate this local refinement to larger regions using a priority queue.

We develop a local optimization to find the most suitable spherical embedding of each vertex through an efficient great-circle search. In this algorithm, we do not update a vertex's spherical embedding if the energy reduction is not significant. A line search mechanism is employed on the great circle of the spherical domain.

Note even if the initial position introduces flip-over, the energy minimization would guide the movement of vertex's spherical image to a valid position free of flip-over. This local optimization is efficient and will converge within finite steps (see Section 2.7 for detailed analysis).

2.5 Priority Queue

When optimizing spherical images of the vertices, we iteratively pick a vertex to do its local optimization. The order of picking vertices is important and it could greatly affect the result and computation efficiency. Intuitively, we shall optimize the vertex whose movement potentially reduces the distortion energy most significantly. Both the magnitude of the first order KKT [8] violation and the distance the vertex can move are critical for the energy reduction. For example, in a region whose spherical mapping shrinks severely, KKT violations of the objective functions on vertices could be big, but spherical embedding of these vertices could not move much (since all these spherical triangles are already very small) before flipover appear. Then moving such vertices may not have high priority. We therefore use the first order KKT violation magnitude multiplying the potential moving distance as the key for this priority queue.

Therefore, for the priority queue we adopt the following priority function τ defined on v_i's spherical image p_i:

$$\tau(v_i) = \rho(v_i) \cdot d \tag{4}$$

where d is the distance from p_i (v_i's image on sphere) to the boundary of its *spherical kernel* (see Section 2.6) along the negative gradient direction. And ρ is the magnitude of the first order KKT optimality violation:

$$\rho(v_i) = \|\nabla E(p_i)\| \sqrt{1 - (\frac{\nabla E(p_i)^T \cdot p_i}{\|\nabla E(p_i)\|})^2} \tag{5}$$

where $\nabla E(p_i)$ is the gradient of the objective function E of eq (3) at vertex v_i. Note that the feasibility condition $\|p_i\| = 1$ is always guaranteed by the construction. In our experiments, we simply use the average distance from p_i to its spherical one-ring to approximate d. $\tau(v_i)$ therefore estimates the aforementioned potential function reduction at vertex v_i, measured via the first order KKT optimality condition violation ρ at p_i multiplied by d.

2.6 Spherical Kernel and the Mapping Bijectivity

The spherical kernel can be defined on the spherical polygon formed by the one-ring neighboring vertices of a vertex v_i. It is defined and can be computed as the intersection of the open hemispheres defined by the spherical polygon edges. To avoid the flip-over on the spherical parametrization, we shall maintain a valid spherical embedding. This can be guaranteed if every vertex is inside its spherical kernel. We generalize the planar kernel computation algorithm [5] onto the spherical triangle mesh. The computation is efficient and takes $O(k)$, where k is the number of vertices on the spherical polygon.

The **bijectivity** of the spherical mapping can be shown. First, during local optimization, a non-flipped local region will not be converted into a flipped local region. Therefore, if we can guarantee the initial spherical embedding during the entire progressive refinement is valid, then our final parametrization is non-flipped. Through induction, we can show that a valid initial spherical embedding can always be constructed during vertex split. (1) After the progressive simplification, the mesh is simplified to a tetrahedron M^4 with 4 vertices, which can be embedded on the sphere. (2) Suppose the mesh M^k with k vertices has a valid spherical embedding, and the next refinement is to do the vertex split from v_i^k to $(v_i^{k+1}, v_{k+1}^{k+1})$, then the spherical kernel for v_i^k is not empty. Then it can be shown that non-empty spherical kernel regions for v_i^{k+1}, v_{k+1}^{k+1} can always be constructed [9]. Therefore, a valid spherical embedding for the refined mesh M^{k+1} exists and can be used as the initial spherical positions for the next insertion and refinement. The mapping bijectivity is therefore guaranteed.

2.7 Analyzing Convergence of the Optimization

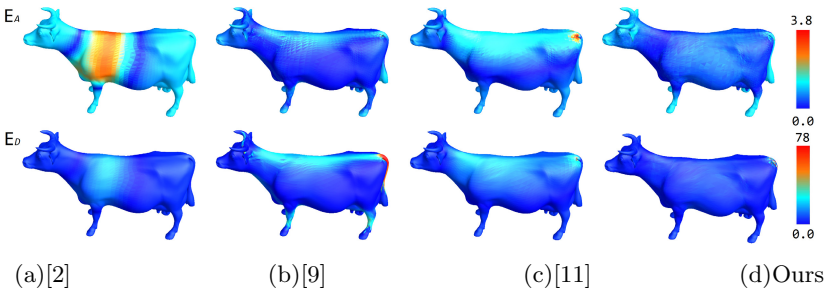
The first order KKT optimality condition of $\min E(p_i)$, subject to $\|p_i\| = 1$ can be written as

$$\nabla E(p_i) - \lambda p_i = 0, p_i^T p_i = 1. \tag{6}$$

where $\lambda \in R$ is Lagrange multiplier associated with the ball constraints.By considering $p^T p = 1$, which is guaranteed by the algorithm, we have $\lambda = \nabla E(p_i)^T p_i$. Then, the 2-norm residue of the left hand side of the first equation in Eq (6) can be written as

$$\rho(v_i) = \|\nabla E(p_i)\| \sqrt{1 - (\frac{\nabla E(p_i)^T \cdot p_i}{\|\nabla E(p_i)\|})^2} = 0 \tag{7}$$

which can be considered as the magnitude of the violation of KKT condition.



(a)[2] (b)[9] (c)[11] (d)Ours

Fig. 1. Comparison of Other Spherical Parametrization Algorithms and Our Method on the Cow model. (a) is from [2]; (b) is from (b)[9]; (c) is from [11] and (d) is from our method. E_A and E_D indicate area distortion and angle distortion. Warmer color, e.g red, indicates larger distortion; while cooler color, e.g. blue, indicates lower distortion. The rightmost column shows our results, which exhibits lower angle and area distortion. Please refer to this paper's online version for the color-encoding.

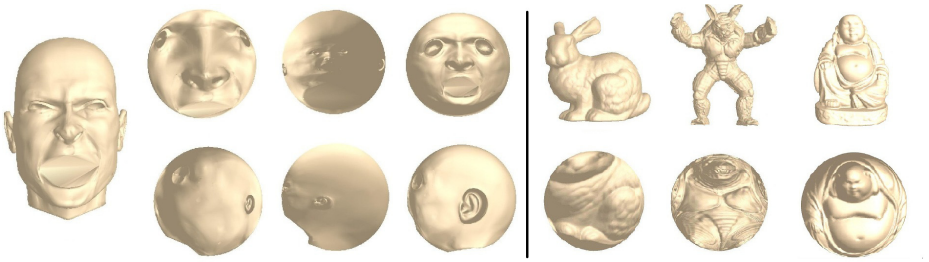
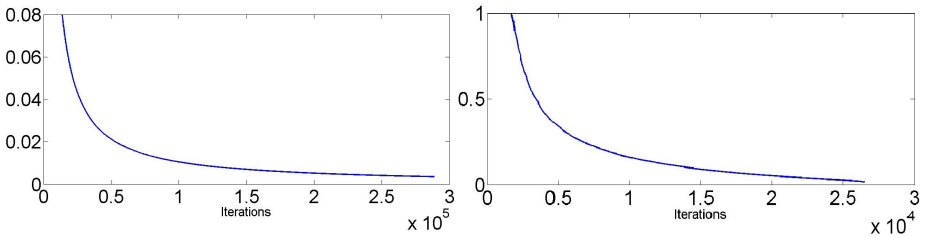


Fig. 2. (a)Comparison of Other Spherical Parametrization Algorithms and Our Method on the Head model. The leftmost column is the input model. Our approach preserves the facial features like eyes, nose, mouth and ears more naturally. (b) Some results of our approach.

When $\rho(v_i)$ is not small, p_i is not close to a local minimum. Then, because the angle between the asymptotic searching directions and the negative gradient direction of the energy function is an acute angle, the Armijo-type great-circle back tracking line search will be successful within a finite number of steps and a sufficient energy value reduction relative to the KKT violation $\rho(v_i)$ will be obtained. This would force the first order KKT violation $\rho(v_i)$ goes to zero, since the energy function value is always bounded above from zero.

Globally, the objective energy is also bounded below, actually nonnegative, and monotonically decreasing. Furthermore, the great-circle back tracking line search conditions will prevent the step size getting too small and the energy will be reduced sufficiently when p_i is far away from local minimum. Therefore, globally, the total energy will decrease relatively rapidly to a minimum value.

The graphs of the total distortion energy per vertex in the optimization are depicted in Figure 3. In this figure we observe the energy drops severely in the beginning and the slope of the graph asymptotically goes towards zero with increasing number of iterations. This indicates our approach finally converges.



(a) Bunny (b) Cow

Fig. 3. Energy (per vertex) with respect to iterations on Bunny and Cow. The y-axis shows the energy; the x-axis shows the iteration number.

3 Spherical Parametrization Results

To perform side-by-side comparisons, we have implemented the harmonic spherical mapping [2], curvilinear spherical parametrization [11], and we obtained mapping results from the progressive spherical parametrization of [9]. We also parameterize various input models using our algorithm under different weights. In experiments demonstrated in this section, we use $\lambda = 0.1$ and $\mu = 1.0$.

Figure 1 demonstrates the effectiveness of our approach on Cow(11K vertices) by color-encoding angle and area distortions of the spherical mappings computed by [2], [11], and [9]. We can see that results of our method in the rightmost column are in cooler color, and therefore it has lower angle and area distortions.

Figure 2-a demonstrates the results of our approach on the Head (13K vertices) model side by side compared with [2] and [11]. Our approach introduces smaller angle and area distortions, and hence better preserves the facial features like eyes, nose, mouth and ears on the sphere. Figure 2-b illustrates some more mapping results computed by our algorithm.

Table 1. Comparison of Statistics on Bunny

Bunny	#V=34K			
	[2]	[11]	[9]	Ours
#FO	2585	0	3	0
E_D	50.8	63.6	78.1	61.4
E_A	22.8	25.5	14.0	14.2
T(s)	2397	91	600	58

Cow	#V=11K			
	[2]	[11]	[9]	Ours
#FO	2536	2	0	0
E_D	51.2	73.2	117.3	69.9
E_A	32.9	23.8	14.4	15.5
T(s)	224	28	420	21

Gargoyle	#V=100K			
	[2]	[11]	[9]	Ours
#FO	6106	9	0	0
E_D	51.7	78.8	81.2	81.8
E_A	93.6	141.7	41.5	47.7
T(s)	24393	1151.4	1380	193

Table 2. Execution Time of Our Approach on Various Models

Models	Cow	Frog	Bunny	Horse	David	Venus	Gargoyle	Amardillo	Budda
#Vertices	11K	25K	34K	48K	50K	50K	100K	106K	400K
Time(s)	21	48	58	89	111	70	193	250	526

Numerically, the spherical mapping results of Bunny, Cow and Gargoyle, computed by [2], [9] and [11] are compared with our approach in Tables 1. The visualization of E_D and E_A for Cow is given in Figure 1, where cooler color indicates less distorted triangle map and warmer color indicates otherwise. Table 2 lists the running times of our method on 9 models (vertex sizes vary from 11K to 400K). Our experiments (the implementation is not optimized) are conducted on a desktop with AMD Athlon X2 2.9GHz CPU and 2GB RAM.

4 Conclusion

In this paper, we present an effective spherical mapping algorithm using hierarchical optimization scheme minimizing angle and area distortions. Compared with other state-of-the-art spherical mapping algorithms, our method generates a bijective and lowly-distorted mapping, and converges efficiently.

Our current spherical parametrization has not considered the preservation of any features or intrinsic structures of the given geometric model. For example, for local features such as semantic feature points, it is often desirable if the parametrization can flexibly control their distributions on the sphere; for global features such as symmetry, preserving such symmetric structure in its spherical image is also desirable for subsequent geometry processing tasks. These are currently not integrated in our mapping framework. We will explore the feature-preserving simplification (e.g. preserving both feature points and symmetry structure on the base domain) and consider the optimization satisfying such constraints.

References

1. Friedel, I., Schröder, P., Desbrun, M.: Unconstrained spherical parameterization. In: SIGGRAPH Sketches (2005)
2. Gu, X., Yau, S.T.: Global conformal surface parameterization. In: Proc. Symp. of Geometry Processing, pp. 127–137 (2003)
3. Hoppe, H.: Progressive meshes. In: SIGGRAPH, pp. 99–108 (1996)
4. Hormann, K., Greiner, G.: Mips: An efficient global parametrization method. In: Curve and Surface Design: Saint-Malo 1999, pp. 153–162 (2000)
5. Lee, D.T., Preparata, F.P.: An optimal algorithm for finding the kernel of a polygon. J. ACM 26(3), 415–421 (1979)
6. Li, X., Bao, Y., Guo, X., Jin, M., Gu, X., Qin, H.: Globally optimal surface mapping for surfaces with arbitrary topology. IEEE Trans. on Visualization and Computer Graphics 14(4), 805–819 (2008)
7. Li, X., He, Y., Gu, X., Qin, H.: Curves-on-surface: A general shape comparison framework. In: Proc. IEEE International Conf. on Shape Modeling and Applications, pp. 352–357 (2006)
8. Nocedal, J., Wright, S.J.: Numerical Optimization (2006)
9. Praun, E., Hoppe, H.: Spherical parametrization and remeshing. ACM Trans. Graph. 22, 340–349 (2003)
10. Sander, P.V., Snyder, J., Gortler, S.J., Hoppe, H.: Texture mapping progressive meshes. In: SIGGRAPH, pp. 409–416 (2001)
11. Zayer, R., Rossl, C., Seidel, H.P.: Curvilinear spherical parameterization. In: Proc. IEEE International Conf. on Shape Modeling and Applications (2006)