Parameterizing Subdivision Surfaces

Lei He[*] Texas A&M University
Scott Schaefer[†] Texas A&M University
Kai Hormann[‡] University of Lugano

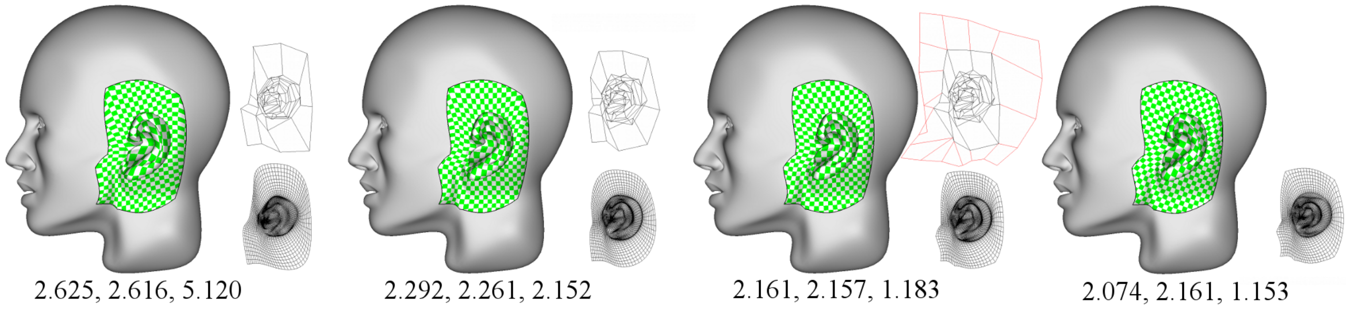

Figure 1: *(a) As-rigid-as-possible (ARAP) polygon parameterization [Liu et al. 2008] applied to the control mesh of a subdivision surface, (b) our subdivision parameterization method, (c) our method using extended charts and (d) ARAP polygon parameterization applied to the control mesh subdivided 3 times. The control mesh of the chart and subdivided chart are shown next to each figure except for the last method, which operates on the subdivided surface directly. The angle, area and stretch distortion for each method are shown below.*

Abstract

We present a method for parameterizing subdivision surfaces in an as-rigid-as-possible fashion. While much work has concentrated on parameterizing polygon meshes, little if any work has focused on subdivision surfaces despite their popularity. We show that polygon parameterization methods produce suboptimal results when applied to subdivision surfaces and describe how these methods may be modified to operate on subdivision surfaces. We also describe a method for creating extended charts to further reduce the distortion of the parameterization. Finally we demonstrate how to take advantage of the multi-resolution structure of subdivision surfaces to accelerate convergence of our optimization.

CR Categories: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Boundary representations; Curve, surface, solid, and object representations; Geometric algorithms, languages, and systems

Keywords: subdivision, parameterization

1 Introduction

Subdivision surfaces [Catmull and Clark 1978] have become a standard for representing highly detailed, smooth shapes for non-real-time applications such as movies. However, with the advent of hardware tessellation in DirectX 11 [Drone et al. 2008], smooth surface representations or approximations thereof [Loop and Schaefer 2008] are now within the grasp of real-time applications such as computer games. Therefore, subdivision surfaces are poised on the threshold of wide adoption over traditional representations such as polygon models.

One of the advantages of subdivision surfaces is that they can model smooth surfaces of arbitrary topology, yet can be controlled by manipulating a polygonal surface called a *control mesh*. This control mesh defines the subdivision surface through recursive refinement using linear combinations of vertices. Hence, the subdivision surface P^∞ is defined as the limit of the process

$$P^{k+1} = S_k P^k$$

where P^0 is an $n \times 3$ vector of control points from the control mesh and S_k is a matrix whose entries depend solely on the local topology of the surface. If the rules encoded by the matrix S_k are chosen correctly, then the limit surface P^∞ is guaranteed to be smooth regardless of its connectivity/topology.

While the geometry of the subdivision surface is important, the shape alone is not sufficient to create digital characters or other realistic objects. Typically we annotate these shapes with additional information such as color, normals or even displacements through the use of a texture map. To create a texture map, the user breaks the surface into a set of charts (regions of the surface that are disconnected in texture space) that can be flattened into the plane. This process is typically referred to as parameterization of the surface. Once flattened, the user may view the chart as a 2D image and paint on the surface as if painting an image.

Given locations of the vertices of each chart in the texture, we can map the data stored in the texture to the surface using standard texture mapping operations built into the graphics pipeline. However, DeRose et al. [1998] showed that using standard linear or bilinear coordinates for the vertices of the charts produces only continuous mappings of the data from the texture to the surface and lacks smoothness, which results in visual artifacts in the final surface. To fix this lack of smoothness, DeRose et al. [1998] take advantage of the parametric nature of subdivision surfaces and subdivide the texture coordinates. Therefore, subdivision not only produces a 3D position for each vertex, but subdivided texture coordinates as well.

[*]e-mail: leih@cs.tamu.edu
[†]e-mail:schaefer@cs.tamu.edu
[‡]e-mail:kai.hormann@usi.ch

Given that the control mesh of a subdivision surface resembles a polygon mesh, it is tempting to apply polygon parameterization methods directly to the control mesh of the subdivision surface. However, this approach ignores the underlying structure of the subdivision surface and leads to more distortion in the parameterization than is necessary.

Contributions

In this paper, we provide a method for parameterizing subdivision surfaces that directly accounts for the underlying parametric representation of the shape. In particular,

- we show how to parameterize a subdivision surface through iterative optimization of a non-linear functional that attempts to make the mapping as rigid as possible;
- we demonstrate that the choice of subdivision rules applied to the charts can affect the distortion of the parameterization and show how to eliminate that distortion along chart boundaries by creating extended charts;
- we also show how to take advantage of the natural, multi-resolution structure of subdivision surfaces to build a multi-resolution optimization that converges quickly to the parameterization with minimum distortion.

2 Related Work

The overall goal of surface parameterization is to find a mapping between a given 3D model and a suitable 2D parameter domain that minimizes the inevitable metric distortion. Recent research has been almost entirely focused on the parameterization of triangle meshes and the last decade has seen an abundance of methods for computing 'optimal' piecewise linear mappings, with various definitions of optimality; see [Floater and Hormann 2005; Sheffer et al. 2006; Hormann et al. 2007] for a comprehensive overview.

Most approaches try to preserve angles, either directly [Sheffer et al. 2005] or by minimizing some discrete measure of conformality [Hormann and Greiner 2000; Lévy et al. 2002; Desbrun et al. 2002], and the results can be further improved by an adequate handling of cone singularities [Kharevich et al. 2006; Kälberer et al. 2007; Ben-Chen et al. 2008; Springborn et al. 2008]. But reducing angle deformation usually comes at the cost of high area distortion, which is undesired in many applications.

Therefore, other methods minimize the stretch of the mapping [Sander et al. 2001; Sorkine et al. 2002] or balance between angle and area distortion [Degener et al. 2003; Tarini et al. 2004; Dominitz and Tannenbaum 2010; Pietroni et al. 2010]. Alas, the respective optimization problems are non-linear and tend to be computationally expensive. Recently, Liu et al. [2008] presented a clever way of efficiently minimizing a non-linear energy that is similar to the Green-Lagrange deformation tensor [Maillot et al. 1993] and measures how far the parameterization is from being isometric. Hence, the resulting as-rigid-as-possible (ARAP) mappings tend to balance the deformation of angles and areas very well.

Almost all these methods are based on discretizing some notion of distortion that is valid for smooth surfaces, but very few works actually deal with computing parameterizations of the latter, despite this problem being well understood in theory [Kreyszig 1991]. While closed-form solutions are known only for simple shapes like the sphere [Mercator 1569; Lambert 1772], a general surface is usually approximated up to a desired accuracy by a triangle mesh to then utilize one of the methods above. For very large meshes the computation can be sped up by exploiting a mesh hierarchy [Hormann et al. 1999; Ray and Lévy 2003; Aksoylu et al. 2005].

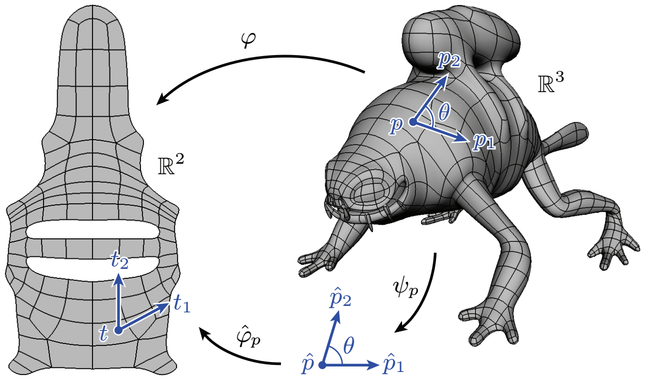

Figure 2: *The parameterization φ that locally maps the surface tangents (p_1, p_2) at p to the texture tangents (t_1, t_2) at t is split into an isometric mapping ψ_p and an affine mapping $\hat{\varphi}_p \colon \mathbb{R}^2 \to \mathbb{R}^2$.*

We are aware of only one related work for reparameterizing single NURBS patches [Woo 1998]. However, this method does not find texture coordinates for the control points of the NURBS patch, but instead creates a simple, non-linear modification of the parametric coordinates during evaluation. Unfortunately this approach is simply not practical for more complex parameterization energies.

Piponi and Borshukov [2000] provide a method for texturing subdivision surfaces seamlessly. However, their technique still treats the control mesh as a polygonal surface and parameterizes the shape using a spring system. To create a seamless texture map on the surface, the authors blend the texture in a small region around the chart boundary.

In this paper, we do not propose yet another parameterization method for triangle meshes, but rather show how the existing methods can be extended for parameterizing subdivision surfaces and how to exploit the particular hierarchical structure of these surfaces.

3 Parameterization

Let us assume that the surface is already split into charts and each chart is a subdivision surface P^∞ with control points P^0. Our goal now is to find a set of parameter points T^0, one for each control point, such that the mapping φ between P^∞ and the limit parameter domain T^∞ has low distortion. We use the energy functional proposed by Liu et al. [2008] for ARAP mappings as the method penalizes both stretching and angle distortion, but our method would also work with any other deformation energy that can be expressed as a function of the 2D parameter points.

For any surface point $p \in P^\infty$ let $p_1, p_2 \in \mathbb{R}^3$ be the surface tangent vectors and $t = \varphi(p)$ be the corresponding parameter point with tangents $t_1, t_2 \in \mathbb{R}^2$ in T^∞. We then map the local frame at p isometrically into the plane to create and consider the affine function $\hat{\varphi}_p$ that maps the vectors

$$\hat{p}_1 = \|p_1\| \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \qquad \hat{p}_2 = \|p_2\| \begin{pmatrix} \cos\theta \\ \sin\theta \end{pmatrix},$$

where θ is the angle between p_1 and p_2, to the vectors t_1, t_2; see Figure 2. We then wish to solve

$$\min_{T^0} \int_{P^\infty} \|\nabla\hat{\varphi}_p - R_p\|_F^2 \, dp, \tag{1}$$

where $R_p \in \mathbb{R}^{2\times 2}$ is the rotation that approximates

$$\nabla\hat{\varphi}_p = (t_1, t_2) \cdot (\hat{p}_1, \hat{p}_2)^{-1}$$

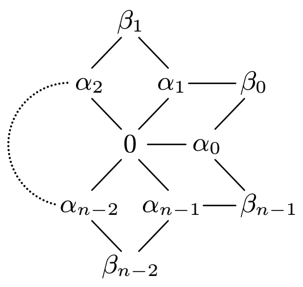

Figure 3: *Tangent mask for a Catmull-Clark surface using the weights in Equation* (2)*.*

best in the Frobenius norm $\|\cdot\|_F$ and a simple, closed-form solution of R_p is given in the Appendix. As shown by Liu et al. [2008], minimizing Equation (1) is equivalent to minimizing

$$\min_{T^0} \int_{P^\infty} (\sigma_1 - 1)^2 + (\sigma_2 - 1)^2 \, dp,$$

where σ_1, σ_2 are the singular values of $\nabla\varphi_p$ at any surface point $p \in P^\infty$, which explains why the resulting parameterizations tend to be close to isometric (isometric mappings satisfy $\sigma_1 = \sigma_2 = 1$).

3.1 Subdivision Surfaces

While many subdivision schemes exist, we concentrate on Catmull-Clark subdivision [1978] due to its widespread adoption. However, all of our results easily extend to other linear subdivision schemes.

For subdivision surfaces, tangents of the limit surface that correspond to vertices of the control mesh are easy to find and can be written as a weighted combination of vertices of the surface. Figure 3 shows the tangent mask for a valence n vertex of a Catmull-Clark surface, which depends solely on the local topology of the surface. The tangent $p^k_{j,1}$ for the j^{th} vertex $p^k_j \in P^k$ at subdivision level k can be found by applying the weights in Figure 3 to the 3D positions of the vertices and summing, where

$$\begin{aligned} \alpha_i &= \left(\frac{1}{n} + \frac{\cos\left(\frac{\pi}{n}\right)}{n\sqrt{4+\cos^2\left(\frac{\pi}{n}\right)}} \right) \cos\left(\frac{2\pi i}{n}\right), \\ \beta_i &= \left(\frac{1}{n\sqrt{4+\cos^2\left(\frac{\pi}{n}\right)}} \right) \cos\left(\frac{2\pi i + \pi}{n}\right). \end{aligned} \qquad (2)$$

Likewise, $p^k_{j,2}$ is given by simply rotating the vertices that the weights are applied to once around the central vertex. Similarly, $t^k_{j,1}$ and $t^k_{j,2}$ are given by applying the tangent mask to the texture coordinates $t^k_j \in T^k$ of the vertices at level k.

Hence, we discretize Equation (1) by summing over the vertices at level k to obtain

$$\min_{T^0} \sum_{p^k_j \in P^k} \left\| (t^k_{j,1}, t^k_{j,2})(\hat{p}^k_{j,1}, \hat{p}^k_{j,2})^{-1} - R^k_j \right\|^2_F \, \|p^k_{j,1} \times p^k_{j,2}\| \quad (3)$$

with one optimal 2D rotation R^k_j per vertex, and minimize this error with the global/local algorithm described in Liu et al. [2008]. We start with an initial guess for the parameterization by triangulating the control mesh and applying a polygon parameterization method such as [Lévy et al. 2002] or [Sheffer et al. 2005] to the polygons. Next, we fix the vertices of the parameterized surface and determine R^k_j for each vertex of the surface at subdivision level k as described in the Appendix. Given the optimal rotations R^k_j, we then solve for

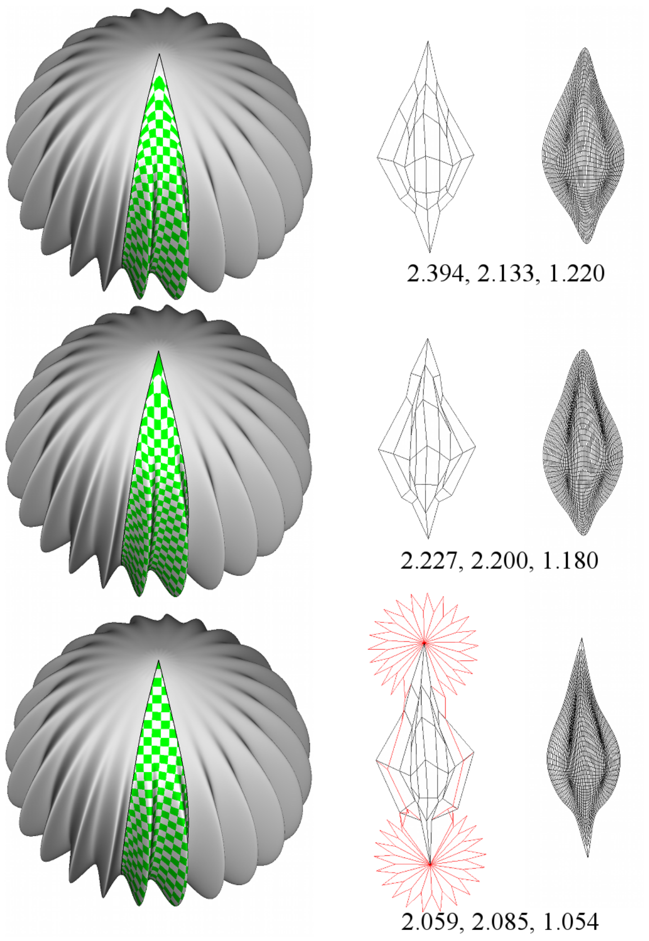

Figure 4: *Parameterization near a valence 24 vertex. Top to bottom: ARAP applied to the control mesh, our subdivision parameterization method and our method using extended charts. The chart control mesh and subdivided chart are shown to the right with the distortion metrics (angle, area, stretch) below. The difference between the surface and texture subdivision rules causes large amounts of distortion near the extraordinary vertex.*

the positions of the parameterized vertices in T^0 by noticing that $\hat{p}^k_{j,1}$, $\hat{p}^k_{j,2}$, $\|p^k_{j,1} \times p^k_{j,2}\|$ are fixed and $t^k_{j,1}$, $t^k_{j,2}$ are linear combinations of the vertices of T^0. Therefore, Equation (3) becomes a simple least squares problem, which can be efficiently solved using a Cholesky decomposition; see [Liu et al. 2008] for details. We then iterate this process starting with the estimation of the rotations R^k_j until convergence.

While the subdivision level k used in Equation (3) is unspecified, we would obviously like to find the limit as k tends toward infinity. Notice that as k increases, the size of the linear system that we solve remains constant as its size is equal to the number of variables in T^0. However, the number of rigid transformations R^k_j that we must estimate increases exponentially with k. Hence, we can take advantage of the multi-resolution structure of the subdivision surface by first computing a solution with $k = 0$. Once the solution has converged, we then subdivide the control mesh as well as the vertex positions of the parameterized mesh. We repeat this process by increasing the subdivision level until the solution converges. In practice, we find that only a few levels of subdivision are necessary to find good solutions and that the optimization converges quickly.

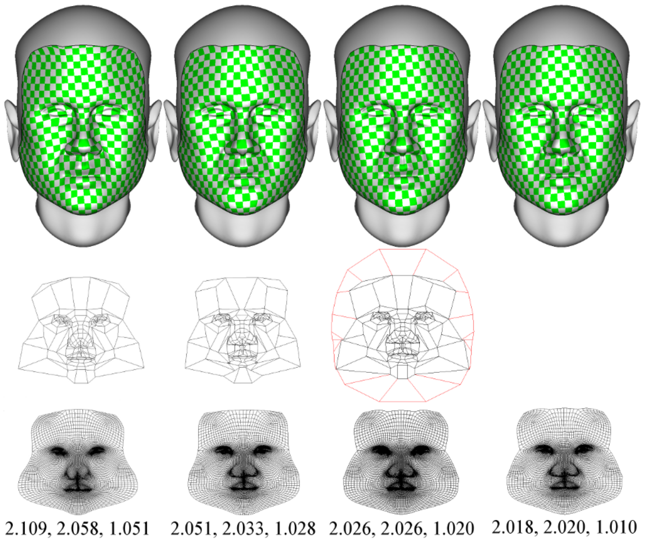

Figure 5: *An example parameterization of a face. From left to right: ARAP applied to the control mesh, our method using boundary rules, our method with extended charts and ARAP applied to the control mesh subdivided three times. Even though the chart control mesh may fold back on itself with our method, the subdivided chart does not. The red edges show the vertices of the extended chart.*

3.2 Chart Boundaries

When parameterizing a polygon mesh, each polygon is assigned to a single chart and its vertices have texture coordinates in the parameter space. This implies that a single vertex may have multiple texture coordinates if its adjacent polygons belong to different charts. The same process applies to subdivision surfaces with polygons in the control mesh assigned to charts. However, each chart in texture space has a boundary at its edge that may not geometrically correspond to a boundary on the surface. For example, Figure 4 shows a surface without boundary even though the parameterized chart contains a boundary. Given that the subdivision rules depend on the topology of the shape, we cannot apply the same subdivision rules to the texture coordinates as to the surface coordinates along the chart boundaries because the topology is different. The natural solution is to apply boundary subdivision rules [Biermann et al. 2000] to the chart while interior subdivision rules are applied to the surface. This difference between subdivision rules does not affect the optimization in Equation (3) except that the weights used to compute $t^k_{j,1}$, $t^k_{j,2}$ change near the chart boundary.

Unfortunately, this seemingly natural decision produces unnecessary distortion along the boundary. Away from corners (valence 1 vertices), the boundary subdivision rules are designed to produce smooth curves. However, the 3D image of the chart boundaries on the surface will not be smooth at vertices of the control mesh unless they pass through the opposite edge of an even valence vertex. Figure 4 (top, middle) shows an example of the effect of using boundary rules in this case. Unfortunately the chart vertices can never be placed in a position to avoid this distortion.

Our solution is to extend each chart to create additional degrees of freedom. For each chart, we find the set of vertices face-adjacent to the chart boundary and give these vertices texture coordinates that correspond to that adjacent chart. The implication of this process is that vertices may have texture coordinates for charts whom their adjacent polygons do not belong to.

The benefit of this chart extension is twofold. First, each chart is given additional degrees of freedom to help minimize the distortion, which results in better parameterizations. Since these vertices affect the geometry of the subdivision surface, so too should they affect the parameterization. Second, the same subdivision rules used for the surface can be used on the charts, which means the same weights are used to construct $t^k_{j,1}$, $t^k_{j,2}$ as $p^k_{j,1}$, $p^k_{j,2}$. Hence, the texture can match the shape of the boundary curve and angles formed by edges on the surface. Figure 4 (bottom) shows the same example as the middle except the vertices are optimized using extended charts.

4 Results

To have some measure of parameterization quality, we use three common metrics that measure the distortion of angles and area [Hormann and Greiner 2000; Degener et al. 2003], and the average L^2 stretch [Sander et al. 2001]. Letting $\sigma^k_{j,1}, \sigma^k_{j,2}$ be the singular values of the matrices

$$\left(t^k_{j,1}, t^k_{j,2}\right) \cdot \left(\hat{p}^k_{j,1}, \hat{p}^k_{j,2}\right)^{-1},$$

and

$$A^k_j = \|p^k_{j,1} \times p^k_{j,2}\|, \qquad \hat{A}^k_j = \|t^k_{j,1} \times t^k_{j,2}\|$$

be the area elements at p^k_j and t^k_j, then these metrics are defined as

$$E_{\text{angle}} = \frac{\sum_j A^k_j (\sigma^k_{j,1}/\sigma^k_{j,2} + \sigma^k_{j,2}/\sigma^k_{j,1})}{\sum_j A^k_j},$$

$$E_{\text{area}} = \frac{\sum_j A^k_j (1/(\sigma^k_{j,1}\sigma^k_{j,2}) + \sigma^k_{j,1}\sigma^k_{j,2})}{\sum_j A^k_j},$$

$$E_{\text{stretch}} = \sqrt{\frac{\sum_j A^k_j \left(1/(\sigma^k_{j,1})^2 + 1/(\sigma^k_{j,2})^2\right)/2}{\sum_j A^k_j}} \sqrt{\frac{\sum_j \hat{A}^k_j}{\sum_j A^k_j}}.$$

Notice that the minimum value for E_{angle} and E_{area} will be 2 assuming no distortion in either quantity and that E_{stretch} will be 1. For the level of subdivision k, we typically choose $k = 5$ to provide a fine discretization of the subdivision surface.

Figures 1 and 5 show examples of our method compared with standard polygon parameterization. In each figure, we calculate an ARAP parameterization of the control mesh using the method of Liu et al. [2008] (left). We also show the result of using our subdivision parameterization both without and with extended charts.

In all cases, our subdivision parameterization reduces both the angle and area distortion as well as the average stretch dramatically over standard polygon parameterization. Finally, we compare the results with applying as-rigid-as-possible polygon parameterization to the subdivided surface after $k = 3$ levels of subdivision. While this comparison is not fair as the number of degrees of freedom has increased by a factor of 64, we provide this comparison as a lower bound to what we could possibly achieve with our method. Even with far fewer degrees of freedom, our subdivision parameterization with extended charts comes remarkably close to this lower bound.

As is evident in these examples, the use of extended charts improves the parameterization as well. This phenomenon is especially apparent in Figure 4 where we show a chart segmenting part of the one-ring of a vertex of valence 24. In this situation, the boundary curve of the chart on the surface is not smooth. However, the subdivision rules applied to the texture create a smooth boundary. This disconnection between the subdivision rules creates a large amount of distortion in the parameterization near the extraordinary vertex.

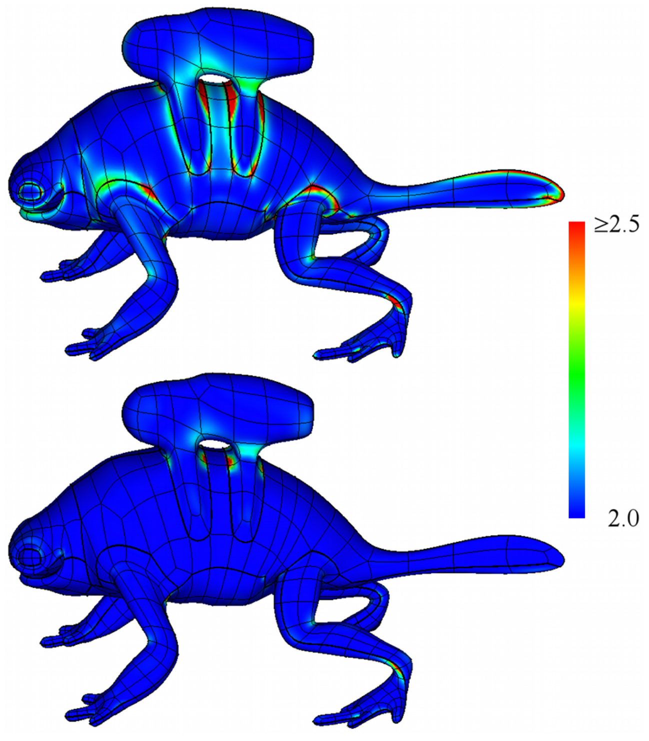

Figure 6: *The area distortion over a set of charts covering a surface. Blue represents no distortion (2.0) and red high distortion (≥ 2.5). We show the subdivision patch structure over the surface and chart boundaries drawn in bold. Top: ARAP parameterization of the control mesh with a total distortion of 2.048. Bottom: our parameterization with extended charts with a total distortion of 2.008.*

In contrast, extended charts have more degrees of freedom and can match the shape of the boundary curve precisely.

Figure 6 shows the area distortion over different regions of an example surface composed over several charts. When using polygon parameterization methods (top), the error in the distortion tends to be concentrated around chart boundaries. The error along the boundaries is especially prominent when passing through an extraordinary vertex, which Figure 4 demonstrates as well. However, extraordinary vertices themselves are also sources of error in the parameterization as can be seen in the saddle configurations near the top of the shape in Figure 6. Our method with extended charts (bottom) drastically reduces the distortion to almost negligible amounts over the vast majority of the surface.

Figure 7 shows the effect of the discretization level k in the resulting parameterization. As the level of subdivision increases, the error of the parameterization decreases. While the size of the system of equations that we solve remains constant, the time taken to build the system of equations depends on the subdivision level. Since R_j^k changes at each iteration, we must construct this system of equations for each iteration. For Figure 7, iterations of our optimization at $k = 1$ take only 0.11 seconds, 0.41 seconds at $k = 2$ and 1.77 seconds at $k = 3$ on an Intel Core i7 920. By starting our optimization at a low level of subdivision, we can converge to a rough solution quickly at very little cost. We gradually refine this solution and optimize at increasing level of subdivision until our solution converges.

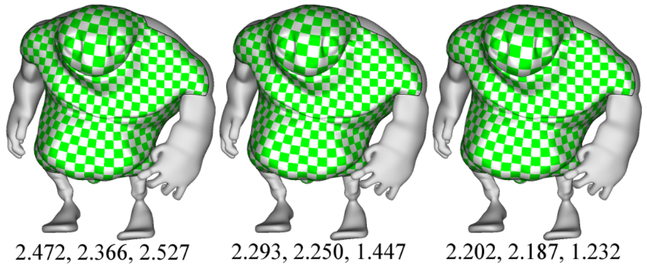

Figure 7: *Our parameterization with extended charts computed at $k = 1, 2, 3$ (left to right) and the distortion (angle, area, stretch) for each level.*

5 Conclusions

We have provided a simple yet effective technique for parameterizing subdivision surfaces. Our method out-performs polygonal parameterization methods and we have shown that using extended charts improves the parameterization errors as well.

Acknowledgements

We would like to thank Bay Raitt for the models used in Figures 6 and 7. The model in Figure 4 was provided by Charles Loop. This work was supported in part by NSF grant CCF-07024099.

References

AKSOYLU, B., KHODAKOVSKY, A., AND SCHRÖDER, P. 2005. Multilevel solvers for unstructured surface meshes. *SIAM Journal on Scientific Computing 26*, 4, 1146–1165.

BEN-CHEN, M., GOTSMAN, C., AND BUNIN, G. 2008. Conformal flattening by curvature prescription and metric scaling. *Computer Graphics Forum 27*, 2, 449–458.

BIERMANN, H., LEVIN, A., AND ZORIN, D. 2000. Piecewise smooth subdivision surfaces with normal control. In *Proceedings of SIGGRAPH*, 113–120.

CATMULL, E., AND CLARK, J. 1978. Recursively generated B-spline surfaces on arbitrary topological meshes. *Computer-Aided Design 10*, 6, 350–355.

DEGENER, P., MESETH, J., AND KLEIN, R. 2003. An adaptable surface parameterization method. In *Proceedings of the 12th International Meshing Roundtable*, 201–213.

DEROSE, T., KASS, M., AND TRUONG, T. 1998. Subdivision surfaces in character animation. In *Proceedings of SIGGRAPH*, 85–94.

DESBRUN, M., MEYER, M., AND ALLIEZ, P. 2002. Intrinsic parameterizations of surface meshes. *Computer Graphics Forum 21*, 3, 209–218.

DOMINITZ, A., AND TANNENBAUM, A. 2010. Texture mapping via optimal mass transport. *IEEE Trans. on Visualization and Computer Graphics 16*, 3, 419–433.

DRONE, S., LEE, M., AND ONEPPO, M., 2008. Direct3D 11 Tessellation. http://www.microsoft.com/downloads/details.aspx?FamilyId=2D5BC492-0E5C-4317-8170-E952DCA10D46.

FLOATER, M. S., AND HORMANN, K. 2005. Surface parameterization: a tutorial and survey. In *Advances in Multiresolution for*

Geometric Modelling, N. A. Dodgson, M. S. Floater, and M. A. Sabin, Eds. Springer, 157–186.

HORMANN, K., AND GREINER, G. 2000. MIPS: An efficient global parametrization method. In *Curve and Surface Design: Saint-Malo 1999*, P.-J. Laurent, P. Sablonnière, and L. L. Schumaker, Eds. Vanderbilt University Press, 153–162.

HORMANN, K., GREINER, G., AND CAMPAGNA, S. 1999. Hierarchical parametrization of triangulated surfaces. In *Proceedings of Vision, Modeling, and Visualization*, 219–226.

HORMANN, K., LÉVY, B., AND SHEFFER, A. 2007. Mesh parameterization: Theory and practice. In *SIGGRAPH 2007 Course Notes*, no. 2.

KÄLBERER, F., NIESER, M., AND POLTHIER, K. 2007. Quadcover—surface parameterization using branched coverings. *Computer Graphics Forum 26*, 3, 375–384.

KHAREVICH, L., SPRINGBORN, B., AND SCHRÖDER, P. 2006. Discrete conformal mappings via circle patterns. *ACM Transactions on Graphics 25*, 2, 412–438.

KREYSZIG, E. 1991. *Differential Geometry*. Dover, New York.

LAMBERT, J. H. 1772. *Beyträge zum Gebrauche der Mathematik und deren Anwendung*, vol. 3. Buchhandlung der Realschule, Berlin.

LÉVY, B., PETITJEAN, S., RAY, N., AND MAILLOT, J. 2002. Least squares conformal maps for automatic texture atlas generation. *ACM Transactions on Graphics 21*, 3, 362–371.

LIU, L., ZHANG, L., XU, Y., GOTSMAN, C., AND GORTLER, S. J. 2008. A local/global approach to mesh parameterization. *Computer Graphics Forum 27*, 5, 1495–1504.

LOOP, C., AND SCHAEFER, S. 2008. Approximating Catmull–Clark subdivision surfaces with bicubic patches. *ACM Transactions on Graphics 27*, 1, 8:1–8:11.

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

MERCATOR, G. 1569. Nova et aucta orbis terrae descriptio ad usum navigantium emendate accommodata. Duisburg.

PIETRONI, N., TARINI, M., AND CIGNONI, P. 2010. Almost isometric mesh parameterization through abstract domains. *IEEE Trans. on Visualization and Computer Graphics*. To appear.

PIPONI, D., AND BORSHUKOV, G. 2000. Seamless texture mapping of subdivision surfaces by model pelting and texture blending. In *Proceedings of SIGGRAPH*, 471–478.

RAY, N., AND LÉVY, B. 2003. Hierarchical least squares conformal maps. In *Proceedings of Pacific Graphics*, 263–270.

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

SHEFFER, A., LÉVY, B., MOGILNITSKY, M., AND BOGOMYAKOV, A. 2005. ABF++: fast and robust angle based flattening. *ACM Transactions on Graphics 24*, 2, 311–330.

SHEFFER, A., PRAUN, E., AND ROSE, K. 2006. Mesh parameterization methods and their applications. *Foundations and Trends in Computer Graphics and Vision 2*, 2, 105–171.

SORKINE, O., COHEN-OR, D., GOLDENTHAL, R., AND LISCHINSKI, D. 2002. Bounded-distortion piecewise mesh parametrization. In *Proceedings of IEEE Visualization*, 355–362.

SPRINGBORN, B., SCHRÖDER, P., AND PINKALL, U. 2008. Conformal equivalence of triangle meshes. *ACM Transactions on Graphics 27*, 3, 77:1–77:11.

TARINI, M., HORMANN, K., CIGNONI, P., AND MONTANI, C. 2004. PolyCube-Maps. *ACM Transactions on Graphics 23*, 3, 853–860.

WOO, A. 1998. Chordlength texturing of spline surfaces. *Journal of Graphics Tools 3*, 2, 15–19.

Appendix

Liu et al. [2008] show that the best rotation R_p in Equation (1) is $R_p = UV^T$, where the orthogonal matrices U and V are taken from the singular value decomposition of $\nabla\hat{\varphi}_p = U\Sigma V^T$, and additional care must be taken to guarantee that R_p has a positive determinant and thus is a true rotation. However, we prefer the following strategy for computing R_p.

Theorem 1. *The rotation $R \in \mathbb{R}^{2\times 2}$ that approximates $A = \left(\begin{smallmatrix} a & b \\ c & d \end{smallmatrix}\right)$ best in the Frobenius norm is*

$$R = \begin{pmatrix} a+d & b-c \\ c-b & a+d \end{pmatrix} \Big/ \sqrt{(a+d)^2 + (b-c)^2},$$

except in the special case $a = -d$ and $b = c$, when all rotations are equally good approximations to A.

Proof. Our goal is to find the rotation angle α that minimizes

$$f(\alpha) = \|R - A\|_F^2 = \left\| \begin{pmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{pmatrix} - \begin{pmatrix} a & b \\ c & d \end{pmatrix} \right\|_F^2$$
$$= (\cos\alpha - a)^2 + (\sin\alpha + b)^2 + (\sin\alpha - c)^2 + (\cos\alpha - d)^2.$$

With

$$v_\alpha = \begin{pmatrix} \cos\alpha \\ \sin\alpha \end{pmatrix} \qquad \text{and} \qquad w = \begin{pmatrix} b-c \\ a+d \end{pmatrix}$$

we can write the first derivative of f as

$$f'(\alpha) = 2\sin\alpha(a+d) + 2\cos\alpha(b-c) = 2v_\alpha^T w,$$

which is zero if and only if v_α is orthogonal to w, that is

$$v_\alpha = \pm\frac{w^\perp}{\|w\|} \qquad \text{with} \qquad w^\perp = \begin{pmatrix} a+d \\ c-b \end{pmatrix}.$$

The correct sign can be found be considering that the second derivative of f,

$$f''(\alpha) = 2\cos\alpha(a+d) - 2\sin\alpha(b-c) = 2v_\alpha^T w^\perp,$$

is positive if and only if $v_\alpha = +w^\perp/\|w\|$. Note that $f'(\alpha) = 0$ in the special case $w = \binom{0}{0}$, hence f is a constant function and all rotations are equally good approximations to A. □