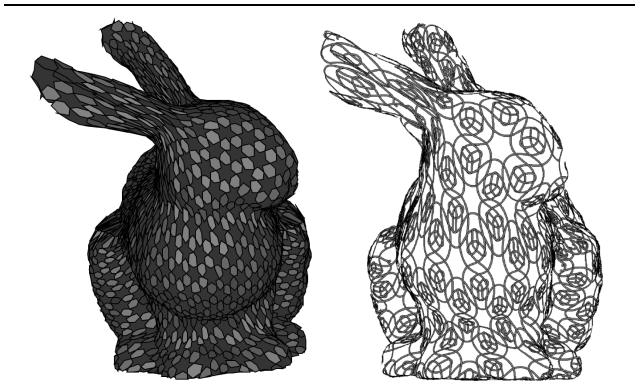
# Pattern Oriented Remeshing For Celtic Decoration

Matthew Kaplan Emil Praun Elaine Cohen
University of Utah

## Abstract

*Decorating arbitrary meshes with Celtic knots requires polygonal meshes with regular connectivity and close to regular face geometry. Because these properties are often irregular, especially in scanned or reconstructed models, the Celtic knots produced may be erratic and undesireable. In this paper we remesh models based on planar tilings defined by the user. Such pattern-oriented surfaces allow us to decorate models with attractive Celtic knots in a consistent fashion and may be applicable to a large number of algorithms that are sensitive to mesh structure.*



## 1. Introduction

Object decoration has been a primary outlet for artistic expression in many cultures yet the research towards computer generated decorations has not been proportional to its influence. Computer graphics scientists have typically concentrated their efforts on more traditional media such as pen-and-ink and painting. The decoration of objects with Celtic knotwork (a form of artwork that consists of rhythmically interwoven threads) has become widespread in recent years. It is seen frequently in jewelry, tattoos and as ornament on many everyday items.

The goal of this work is to decorate the surfaces of 3D meshes with Celtic knotwork. Kaplan and Cohen [12] describe methods to produce Celtic artwork in both 2D and 3D. The edges of a planar graph (in 2D) and the 2D manifold mesh (in 3D) define the pattern of the output knots. In earlier work, allowing the user to design repetitive tilings of geometric shapes was shown to be the key to producing attractive knots. Designing tilings is easy for a planar surface but is extremely difficult for surfaces which are 2D manifold. Therefore, in 3D, the edges of the original input meshes were used to compute the knots. In such input meshes the geometric properties such as face structure and edge connectivity are typically arbitrary and the resultant knots seemed random and lacked the repetitive stylized quality that makes Celtic knotwork attractive, as seen in Figure 1. Therefore, we desire a process by which the user can design infinite planar tilings in 2D and transfer such tilings onto 3D meshes for the calculation of Celtic ornament.

We propose a remeshing technique that imposes shape and connectivity contraints on the resulting mesh. Our technique samples the original mesh with a pattern defined by a planar tiling thus allowing us to transfer designs in a plane to the mesh surface. The Celtic knots computed on such meshes accurately transfer the original artistic intent of the user in a way not previously possible. Furthermore, there may be a large class of artistic algorithms that can benefit from such meshes. The resulting meshes may be more attractive and artistic, due to the patterns embedded on the surface, than meshes produced using other techniques.

### 1.1. Approach Overview

Given a surface mesh we obtain a spherical parameterization of the surface as described by Praun and Hoppe [15]. We unfold the spherical parameterization into a geometry image. We define rules that allow the user to tile the plane with an infinite lattice of a specific pattern of interlocking polygons. Since the geometry image exists in a 2D plane, we construct such a lattice within the geometry image domain. The vertices of this lattice define the sampling pattern used to construct a new mesh from the geometry image. Finally, we compute Celtic knots from the new surface mesh.

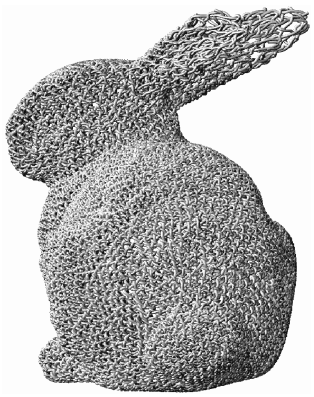


**Figure 1. An example of the knots produced on the original bunny model is shown. Notice that the thread pattern seems essentially random.**

## 2. Related Work

### 2.1. Surface Parameterization

Associating a surface with a planar domain has traditionally consisted of partitioning the surface into charts and packing the charts into a texture atlas. Other approaches have used the connectivity of the surface charts to form a domain complex (semi-regular remeshing), such as that described by Khodakovsky et al. [13]. Because we are attempting to transfer an infinite tiling of a planar domain, chart based methods which do not result in a planar parameterization may not be appropriate. Gu et al. [7] introduced geometry images, in which the geometry is resampled into a completely regular 2D grid. Praun and Hoppe [15] and Gotsman et al. [6] both describe methods for mapping genus zero objects to a sphere to produce a spherical parameterization.

### 2.2. Remeshing

A great number of remeshing schemes have been proposed. A good review is given by Alliez et al. [2]. Most of these combine vertex optimization with mesh simplification but place no constraint on the local shape of mesh elements. Alliez et al. [1] have proposed a remeshing scheme that samples the surface anistropically to produce a mesh whose edge connectivity follows the lines of maximal and minimal curvature. While this method is the closest in spirit to ours, it shares very little in common procedurally with our algorithm. Gu et al. [7] and Praun and Hoppe [15] have both shown how to remesh from geometry images. While they remesh with quadrilaterals, we will demonstate how to remesh arbitrary geometries taking into account the boundary rules they describe.

### 2.3. Ornament and Tilings

Kaplan and others have examined the generation and application of shapes in several contexts such as Islamic artwork [9], symettrohedra [10], and Escherized shapes [11]. Both the Escherization and Islamic art systems produce tilings of interesting polygonal shapes in a 2D plane but are too specific to their stated problems to fully characterize the set of patterns we might wish to produce. The symmetrohedra produce polyhedra by embedding polygons on a sphere. While this seems ideal for the spherical parameterization, it does not encompass a large enough class of patterns to be sufficient for our needs and offers no ability for higher level control over the patterns produced. Neyret and Cani [14] create pattern based ornament by constructing a tiling of equilateral triangles on the original mesh. Their method was limited to triangular primitives and required extensive user design to achieve texture primitives that matched along triangle boundaries.

### 2.4. Celtic Knotwork

The need for quality meshes has been noted extensively in the context of computer generated Celtic knots which most often have relied on grid meshes to achieve interesting knots, largely due to the regularity of the structure [5, 12]. Browne [4] attempts to construct semi-regular meshes around 2D shapes in order to impose order on the resulting knots and notes the importance of quality mesh structure for font generation. We use the method described by Kaplan and Cohen [12] due to its ability to construct knots around arbitrary mesh configurations and to construct coherent knots in 3D.

## 3. Parameterization

Since we are attempting to transfer a planar tiling pattern onto a 3D surface we require a parameterization that associates a planar domain with the 3D mesh. While obtaining a parameterization is not the goal of this work it is a critical component of our technique. The infinite planar tiling imposes the constraints that the choice of parameterization model should be continuous in the plane and should be able to expand to an infinite plane. If a parameterization does not expand to the potentially infinite plane, there will be seams over which the continuity of the tiling pattern will be broken.

These constraints rule out the use of chart based methods such as Khodakovsky et al. [13] which have the desireable properties of low distortion and the ability to handle ar-

bitrary genus. They evaluate to a planar domain but their atlases are not continuous. Traditional texture atlases also lie in the plane but are not continuous. Global conformal maps [8] don't work well for genus zero objects since they introduce cuts and do double covering, changing the object to a higher genus. They would work for higher genus models producing several infinitely tiled planes with transition cuts between them. These transitions make them difficult to work with but might be a useful avenue for future work. The original geometry images described by Gu et al [7] would not allow us to tile the infinite plane and have irregular boundary conditions which it would make it difficult to apply the tiling across the seams. We use the spherical parameterization described by Praun and Hoppe [15]. Their parameterization method transforms the original mesh into a geometry image which defines a square planar domain with dimensions $[0 \rightarrow 1]$ in both $x, y$. This meets our requirements of planar continuity and the ability to expand to an arbitrarily large plane. This method does constrain us to use genus zero surfaces but this describes a large enough class of models to be considered useful. The boundary conditions of the geometry image may introduce seams but they are regular and predictable and can be handled as a special case.

We will describe the basics of the spherical parameterization but refer the reader to [15] for more information. Given a surface mesh $M$, a spherical parameterization of the surface is created, $(S \rightarrow M)$ by forming a single continous invertible map from the unit sphere to the mesh. Each mesh edge is mapped to a great circle arc and each mesh triangle is mapped to a spherical triangle bounded by these arcs. Next, a spherical parameterization $(S \rightarrow D)$ of a domain polyhedron, $D$, is created from either a tetrahedron, octahedron or cube. Then the domain is unfolded into the geometry image $(D \rightarrow I)$. Because the domain of the polyhedron parameterization matches that of the spherical parameterization, the spherical parameterization of $M$ can now be mapped to the geometry image. The geometry image (as shown in Figure 2), is essentially a 2D square representation of the original object. This method is quite robust, but will introduce distortion near narrow extremities.

We use an unfolded octahedron as the polyhedron to specify the domain map with. This unfolds cleanly into a square domain map and permits simple extension rules that allow traversal over the boundaries in a continous manner. The key is to extend the grid by rotating it $180^\circ$ around the midpoints of the boundaries, leading to the topology shown in Figure 3.

## 4. Tiling Patterns

Prior to the remeshing operation, we tile the plane with a user-defined pattern of specific polygonal shapes within the domain of the geometry image.

**Figure 2. The bunny and gargoyle models and their corresponding geometry images.**

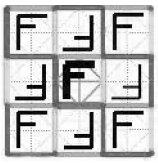


**Figure 3. Boundary conditions for a spherically parameterized geometry image using an octahedral unfolding. The geometry image is represented by the central square.**

The tiling patterns we show are periodic and can be represented as translational units and translation vectors. A translational unit is a set of interlocking polygons that represents the basic pattern. Translation vectors define how to translate "copies" of each translational unit to tile the plane with the pattern. To perform the tiling, we draw an initial translational unit within the geometry image domain. Then we create additional copies of the translational units and translate them by the translation vectors. We continue this until the domain space is completely occupied. Any polygons whose edges fall outside the domain boundary are not added. In order to align the pattern appropriately with the geometry image the bounds of the translational unit should be integral divisors of the domain dimensions.

For examples of patterns laid over geometry images see Figure 9.

### 4.1. Boundary Conditions

The octahedral unfolding that produces the geometry image creates special boundary conditions. Regard Figure 5a. The unfolding process essentially "slices open" the octagon along the edges from one corner vertex. The result of this is that the corner vertex, $E$, actually occurs at all four corners of the geometry image and each edge adjacent to vertex $E$ is "split" and occurs symmetrically about the midpoint of each geometry image boundary edge.

Because of this, any vertex or edge crossing that oc-

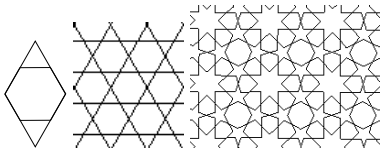


**Figure 4. At left is a simple translational unit with translation vectors in each of the four cardinal directions. The planar tiling of the translatioal unit is shown in the middle. A more complex pattern created with our system is shown at right.**

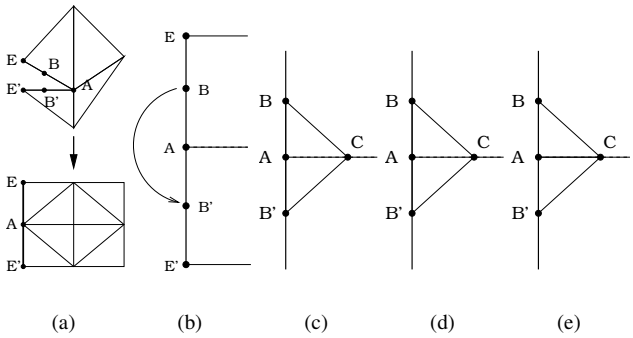



**Figure 5. a)Unfolding an octahedral domain into a planar domain. b) Domain boundary symmetry requirements. The red lines represent the boundary of the geometry image domain. *A* is the midpoint of the left edge of the geometry image. Any vertex $B$ along a domain boundary edge must be matched by an equivalent vertex 180° about $A$. Edges such as $AB$ and $AB'$ are geometrically identical. c-e) Error conditions near boundary midpoints.**

curs along the domain boundary edge along the segment $AE$ must also occur symmetrically about vertex $A$ along the segment $AE'$, as seen in Figure 5b. If this condition does not hold, there will be gaps in the new mesh where the edges $AE$ and $AE'$ connect since the the new geometry will not approach the edge uniformly. Because the tilings must maintain this 180° rotational symmetry about the midpoints of the boundary edges, our patterns tend to have lengths and widths that are integral divisors of the domain dimensions and expand uniformly in the x and y directions. Rotating the tiling patterns by angles that are not multiples of 90° may not be appropriate in many cases since changing the axis of expansion won't preserve the rotational symmetry.

Additionally, there are several cases in which patterns may preserve the above properties may still evaluate to a non-manifold surface. The symmetry about the midpoint along any boundary edge means that vertices that are equidistant from the midpoint along the boundary edge evaluate to the same position, as we would expect. Examine the case shown in Figure 5c. In this case, both vertices that compose edge $BB'$ are really the same vertex. Therefore, this edge evaluates to a single vertex so it is removed from the polygonal face.

When a figure has a similar topology to Figure 5c but there is a vertex at the midpoint, as seen in Figure 5d, the edges on either side of the midpoint $A$ are really the same edge. Geometrically, this evaluates to a polygon that has a set of edges that move towards the interior and double back on each other. To make this face valid, we remove both of these edges from the polygon.

In Figures 5c-d, edge removal left us with a polygon with only two edges which cannot be a closed polygonal face. We remove such faces from the output mesh and the resulting boundary seam on either side will be $BC$.

Another problem that may occur is when two shapes are geometrically identical in 3D despite occupying different areas of the domain in 2D. In Figure 5e, both faces have matching boundary vertices and identical interior vertices. Since both of these shapes are identical, we end up with a geometrically flat area on the remeshed model, where both faces end up being essentially flip sides of a coin. We detect such situations and subdivide the non-boundary edges $CB$ and $CB'$ to differentiate between the two faces.

We apply polygons across domain boundaries so that the pattern fully tiles the remeshed model. In some instances, however, polygons applied across the boundary do not match up with the available space on the opposing side of the boundary. This may occur when patterns do not maintain the required rotational symmetry or are drawn off-axis or at the wrong scale. Such cases should still be handled since there may be many patterns that we would like to use that do not meet the previously stated requirements.

When this does occur, it leaves holes in the new mesh as seen in Figure 6. While there may be no perfect solution to this, besides reorienting the pattern so that it is situated correctly, there are several methods that will allow us to obtain useable meshes. First, we can pull all vertices near the domain boundary to actually lie on the domain boundary. This has the effect of filling the remaining space and preserving the topology of the pattern though shapes may become larger than they would normally be and the transitions between shapes along the boundary may not be preserved. Second, we can fill the remaining open space near the boundaries by inserting polygons in that space. These polygons are formed by walking along the remaining open edges of the pattern and creating polygons between vertices

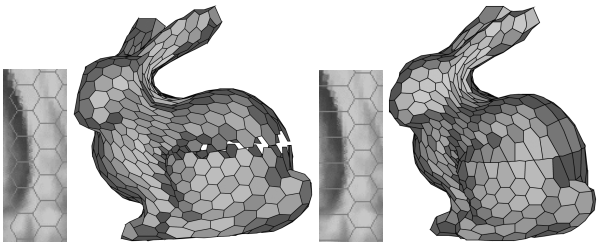


**Figure 6. If a pattern does not approach the boundary correctly (as in the two leftmost images), a variety of techniques may be used to handle the boundaries appropriately. In this case, the vertices near the boundary have been dragged to the boundary and adjacent polygonal faces across the domain boundary have been stiched together. Here, the left edges of the rectangular images are the left boundary edges of the geometry image.**

that touch the domain boundary. This minimizes distortion of the pattern but does not preserve the pattern topology.

Because the domain boundaries require rotational symmetry about their midpoints, if the pattern of vertices and edge crossings between segments $AE$ and $AE'$, as seen in Figure 5b, do not match, then there will be discontinuities in the new mesh. To handle this, for each vertex in $AE$ that is not matched in $AE'$ we insert a new vertex at the appropriate location and vice versa. The edges and polygons along $AE$ and $AE'$ are remapped to take the new vertices into account. This method does not preserve the topology of the pattern but is important for producing closed models.

## 5. Remeshing

The vertices of the tiling pattern are the locations at which we sample the geometry image and the edges of the pattern define the connectivity of the new mesh. For each sample vertex, we find the barycentric coordinates of the triangle that the sample location lies within in the geometry image. The new vertex position is the location defined by the barycentric coordinates within the triangle in the original model that the geometry image maps to. The result of this algorithm is a 2D manifold mesh with the pattern of shapes embedded into its structure.

We have shown simple periodic tilings here since that is the class of patterns that work well with the Celtic knots algorithms. There is no reason that other patterns that fill the domain space could not be used to perform the remeshing.

At this point, we have essentially remeshed twice; once to obtain the geometry image and a second time to embed the pattern. It is possible to instead transform the pattern vertices back through each of the mappings (parametrization and geometry image) to the original model instead of doing a linear approximation directly on the geometry image with the trade off of slightly more complex code. We chose not to do this since the geometry images we worked with were extremely high resolution and the results we obtained worked well with the Celtic Knots program and were visually appealing.

## 6. Knotwork

Celtic knotwork is a type of art that consists of intertwined threads that maintain a characteristic over-under pattern at each corssing. The method of producing Celtic knots described by Kaplan and Cohen [12], uses the midpoints of each edge in the mesh to define the crossing positions and connectivity of the threads in the knot. The thread paths connect adjacent crossing positions which is the reason that the shape of the edges is of critical importance in designing Celtic knots (see Figure 7).

An important question to ask is: if we have a planar parameterization of the mesh and can calculate planar knots easily, why remesh at all? Why not use the parameterization as texture coordinates and texture map the ornament? Why should the mesh structure bear this load? The answer is that the knots program constructs splines paths for the threads based on the midpoints of each edge. If we texture map a planar knot onto the surface, the knot threads will appear disjoint and "painted" onto the surface of the object whereas calculating knots based on the new mesh will result in smooth knots that approximate the surface but do not exactly lie on it.

The crossing patterns of the Celtic knots can be changed by associating a breakpoint value at every edge. These breakpoint values (0,1 or 2) are used to change the thread paths of the knot and give interesting variations on knot patterns. With large meshes, it is very time consuming to input these breakpoints by hand. We associate breakpoint values with each edge of each polygon defined in the pattern application system. We save the breakpoint information along with the model file and they are read as a pair into the Celtic knots program which computes the knots automatically, as seen in Figure 9 bottom.

## 7. Results

The results of this algorithm may be evaluated from two perspectives: the quality of the remeshed models and the quality of the knots produced from these meshes though it must be stressed that the remeshed models are a pleasant by-product rather than our goal.

A variety of remeshed examples appear in Figures 6- 9. The class of tilings that we can successfully create and em-

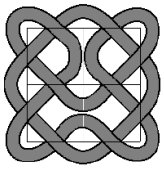


**Figure 7. An example of a 2D Celtic knot produced on a graph of 2x2 squares. Notice that crossings occur at each edge midpoint and connect adjacent edge midpoints except at the breakpoint edges. Regular edges are colored black. Breakpoint edges are colored blue and yellow.**

bed that maintain the boundary constraints is large and encompasses all of the patterns we transferred from the 2D Celtic knot program. With the techniques we have presented for handling boundary conditions, we are able to handle all 2D patterns, though the behavior near the boundaries is not guaranteed to preserve the topology or minimize distortion.

The quality of the approximation can be affected for three reasons: parameterization, pattern scale (i.e. sample volume) and pattern shape. We do not measure the distortion of the geometric properties of the new mesh since that is a problem with parameterization and as parameterization techniques improve, so will the results of our method. Distortion does occur on narrow extremities consistent with the distortion in the spherically parameterized geometry image.

The images presented here were done with relatively coarse patterns. To minimize the error between the original and new meshes, the tiling patterns can be drawn at as fine a scale as desired. Because the sampling pattern produced by the tiling may not be uniform, different parts of the remeshed model may approximate the original model less well than others depending on the density of samples. Also, because the polygons of each face are not tesselated and may be arbitrarily complex (depending on user preference), the faces are most probably non-planar for any non-triangular faces. Fortunately, this is not a problem for us since the Celtic knots program does not require planar faces.

The 3D knotwork produced with this technique is a significant improvement over previous results. An example of knots produced with the previous method is shown in Figure 1 at left. Note that the threads are essentially random and contain no repetitive patterns or motifs (a hallmark of Celtic design). Examples of the type of results possible using the the remeshed models are shown in Figure 9. The new images show a variety motifs taken from classical Celtic knotwork repeated over the surface of the models. The threads are coherent and create repeated figures that traverse the entire surface in a consistent fashion.

Because the output of the Celtic knots program is so dependent on the quality of the mesh, prior work only showed quality knots calculated over uniform polyhedra since the quality of more interesting models was so poor. The remeshing procedure presented here has made knot decoration of 3D meshes a viable tool for animators and designers.

## 8. Conclusion and Future Work

We have presented techniques for creating 2D tilings of user-defined patterns of polygons, for remeshing models with those patterns and for applying the results of these methods to the automated construction of Celtic knotwork.

Our remeshed models are 2D manifold and correctly embed the desired pattern of shapes yet there are several natural avenues for further research. Currently, our system only handles genus zero models. We would like to extend this to include other methods of parameterizations that include models of higher genus. Remeshing has been a widely researched area in recent years and we believe that the applications relevent to pattern-oriented remeshing will grow as such meshes become available. Our technique naturally compresses the mesh but is almost definitely not an optimal compression.

This area of research has a number of natural extensions to texture synthesis techniques. Each vertex of the remeshed model can be used as coordinates in a texture parameterization with texture coordinates could be generated automatically from the tiling application.

The 3D Celtic knotwork produced in this framework is significantly better than 3D results from previously published work. The ability to embed arbitrary patterns of shapes into the meshes allows us to transfer previously designed motifs and designs from 2D onto models allowing us a large measure of control in designing specific decorations. The extension of this system to include non-genus zero models would allow us to use models of all the character symbols. This would allow us to decorate an entire font set ( a traditional Celtic decoration motif) automatically by remeshing all the character models with a single pattern and computing knots on the remeshed models automatically.

Pattern-oriented remeshing has further implications for transferring decorative toolsets from 2D to 3D. It seems natural to extend the work of Wong et al. [16] in floral decoration, Kaplan and Salesin [11] in escherization and others to compute 3D decorations using such meshes. This may have far reaching implications for creating a wide variety of ornament directly on 3D models which could be a useful tool for artists and animators.

## References


[1] P. Alliez, D. Cohen-Steiner, O. Devillers, B. Levy, and M. Desbrun. Anisotropic polygonal remeshing. *SIGGRAPH*, pages 485–493, 2003.

[2] P. Alliez, M. Meyer, and M. Desbrun. Interactive Geometry Remeshing. *SIGGRAPH*, pages 347–354, 2002.

[3] B.Grunbaum and I.Steward. Tilings and patterns, 1986.

[4] C. Browne. Font decoration by automatic mesh fitting. In *RIDT*, pages 22–43, 1998.

[5] A. Glassner. *Andrew Glassner's Other Notebook*. A.K.Peters, 2002.

[6] C. Gotsman, X. Gu, and A. Sheffer. Fundamentals of spherical parameterization for 3d meshes. In *SIGGRAPH*, pages 358–363, 2003.

[7] X. Gu, S. Gortler, and H. Hoppe. Geometry images. In *SIGGRAPH 2002, Proceedings*, 2002.

[8] X. Gu and S.-T. Yau. Global conformal surface parameterization. In *Proceedings of the Eurographics/ACM SIGGRAPH symposium on Geometry processing*, pages 127–137. Eurographics Association, 2003.

[9] C. S. Kaplan. Computer generated islamic star patterns. In *Bridges*, 2000.

[10] C. S. Kaplan and G. W. Hart. Symmetrohedra:polyhedra from symmetric placement of regular polygons. In *Bridges 2001*, 2001.

[11] C. S. Kaplan and D. H. Salesin. Escherization. In K. Akeley, editor, *SIGGRAPH 2000*, pages 499–510, 2000.

[12] M. Kaplan and E. Cohen. Computer generated celtic design. In *Eurographics Symposium on Rendering*, 2003.

[13] A. Khodakovsky, N. Litke, and P. Schröder. Globally smooth parameterizations with low distortion. In *SIGGRAPH 2003*, 2003.

[14] F. Neyret and M.-P. Cani. Pattern-based texturing revisited. In *SIGGRAPH 99 Conference Proceedings*, pages 235–242. ACM SIGGRAPH, Addison Wesley, Aug. 1999.

[15] E. Praun and H. Hoppe. Spherical parameterization and remeshing. In *SIGGRAPH 2003*, pages 340–9, 2003.

[16] M. T. Wong, D. E. Zongker, and D. H. Salesin. Computer-generated floral ornament. *Computer Graphics*, 32:423–434, 1998.


**Figure 8. Examples of pattern-oriented remeshed models.**

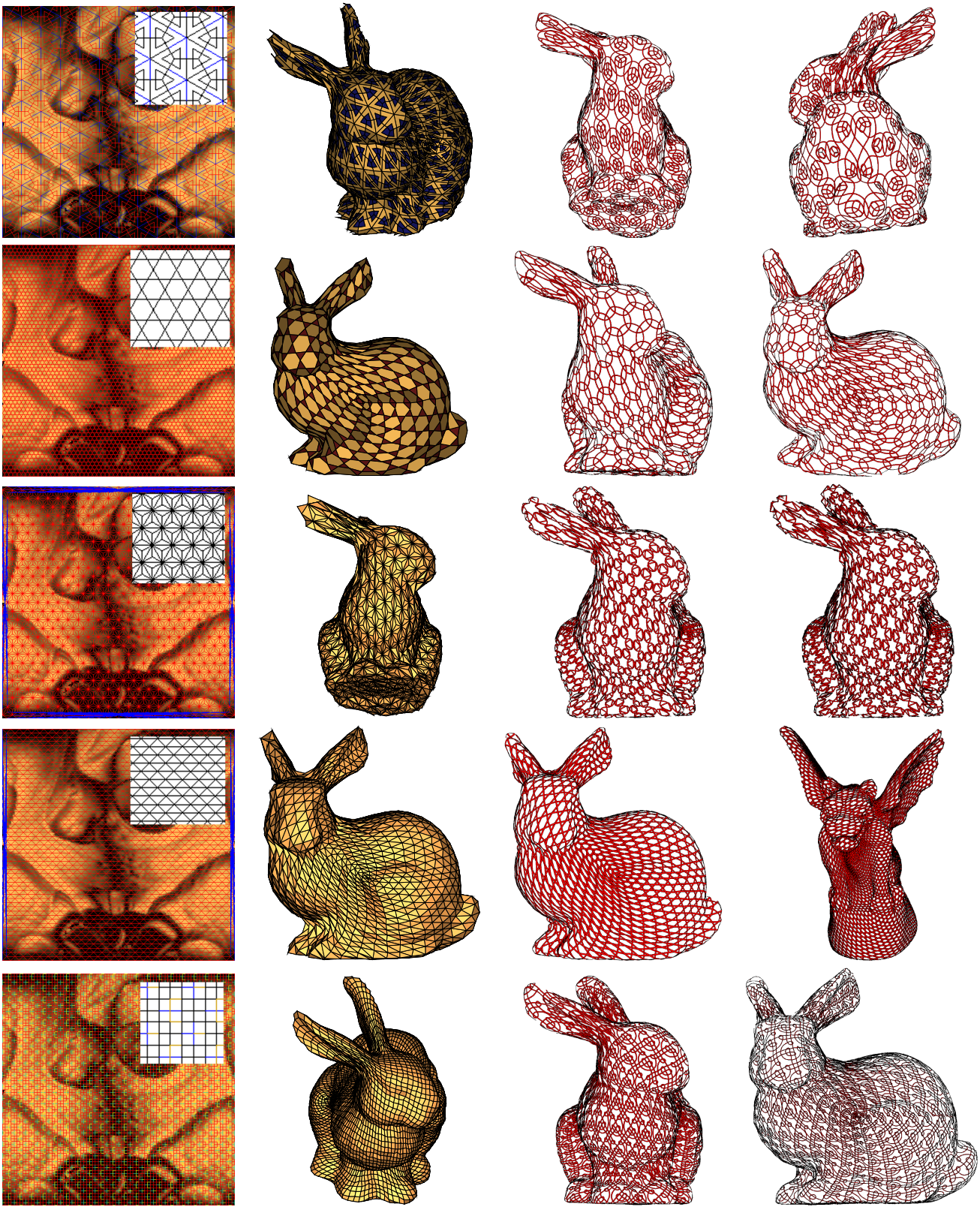


**Figure 9. Each row represents: At left are the geometry images with tiling patterns overlaid. Inset are larger views of the tiling patterns. Blue areas near the borders are edges that overlap the domain boundary. Next are the results of a remeshing performed with the tiling pattern. At right are examples of Celtic knots computed on the remeshed models.**