Parameterization of Tubular Surfaces on the Cylinder

Toon Huysmans
Vision Lab, Dept. of Physics
University of Antwerp
Groenenborgerlaan 171
Belgium 2020, Antwerp
toon.huysmans@ua.ac.be

Jan Sijbers
Vision Lab, Dept. of Physics
University of Antwerp
Groenenborgerlaan 171
Belgium 2020, Antwerp
jan.sijbers@ua.ac.be

Brigitte Verdonk
Dept. of Math. and CS.
University of Antwerp
Middelheimlaan 1
Belgium 2020, Antwerp
brigitte.verdonk@ua.ac.be

ABSTRACT

In this paper we develop a method to parameterize tubular surfaces onto the cylinder. The cylinder can be seen as the natural parameterization domain for tubular surfaces since they share the same topology. Most present algorithms are designed to parameterize disc-like surfaces onto the plane. Surfaces with a different topology are cut into disc-like patches and the patches are parameterized separately. This introduces discontinuities and constrains the parameterization. Also the semantics of the surface are lost. We avoid this by parameterizing tubular surfaces on, their natural domain, the cylinder. Since the cylinder is locally isometric to the plane we can do calculations on the cylinder without loosing efficiency. For speeding up the calculation we use a progressive parameterization technique, as suggested in recent literature. Together, this results in a robust, efficient, continuous, and semantics preserving parameterization method for arbitrary tubular surfaces.

Keywords

parameterization, remeshing, geometry images, texture mapping

1. INTRODUCTION

Surface parameterization is a technique to convert a mesh, described using primitives like triangles, quadrilaterals, or polygons, into a parametric description of the surface. In most applications the surface is two-dimensional and it is embedded in a three dimensional space. Thus, a parameterization is a map from a two-parameter domain onto the three-coordinate surface.

During the last ten years, parameterization has become an important topic in computer science and especially in computer graphics. It has a variety of applications such as: texture-mapping [LPRM02, SGSH02], rendering acceleration [GGH02], morphing [Ale02, ZSH00], remeshing and level of detail [EHL$^+$95, PH03, AMD02], surface fitting [BGK95], surface description [SD02] and form analysis [Sty01].

Most of the techniques in the literature are concerned with the parameterization of topological discs. The parameterization of surfaces of other topology is addressed by cutting the surface into one or more patches, of disc topology, and parameterizing the patches separately. This cutting constrains the parameterization process from the beginning and it also introduces discontinuities into the parameterization. For some applications, like global form analysis, morphing, and surface fitting, this is undesirable. The only way to parameterize a surface of non-disc topology, without cutting it, is by parameterizing it on a domain that has the same topology as the surface. For example, surfaces with spherical topology can be parameterized on the sphere [PH03, GGS03, GWC$^+$03]. In [KS04] and [SAPH04] triangle surfaces are parameterized onto other triangle surfaces that share the same topology.

We are interested in parameterizing surfaces with cylindrical topology onto the cylinder. This is done by Zöckler et al. in their paper on morphing [ZSH00], where they parameterize the cylindrical surface in two stages: first they cut the surface and parameterize it onto the plane, and then the parameterization is glued back together and optimized on the cylinder. Since the surface is cut, distortions are introduced in the first optimization and therefore they have to optimize the parameterization a second time. For complex surfaces this method might not find the optimal parameterization. Our algorithm is different; we directly parameterize onto the cylinder without cutting the surface. This way our algorithm is capable of parameterizing

The Journal of WSCG, Vol.13, ISSN 1213-6964
WSCG2005, January 31-February 4, 2005
Plzen, Czech Republic.
Copyright UNION Agency–Science Press

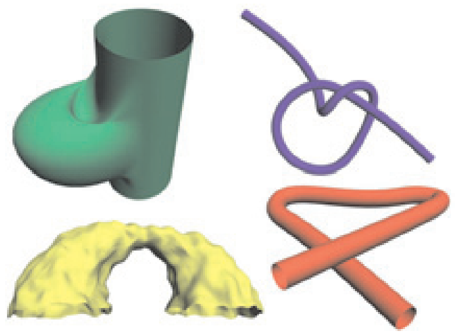
Figure 1: Some examples of tubular surfaces.

virtually any tubular surface with low distortion.

Goal The goal of this work is to find a one-to-one mapping from the surface of the cylinder to an arbitrary tubular surface. With 'tubular surface' we mean any elastic deformation of a sphere with two holes (boundaries), see figure 1 for a number of examples. The upper boundary of the cylinder should map to one of the boundaries of the tubular surface and the lower boundary of the cylinder should map to the other boundary of the tubular surface. The interior of the surface of the cylinder then has to be mapped to the interior of the tubular surface. This is illustrated in figure 2.

There are an infinite number of maps possible between the cylinder and a tubular surface, but we desire a map that is a balanced tradeoff between the following two properties. First, we require that the semantics of the cylinder are ported to the tubular surface. By this we mean that axial lines on the cylinder are mapped to lines that run in the axial direction on the tubular surface and that radial curves of the cylinder are mapped to curves that run in the radial direction on the tubular surface (see figure 3). Second, we also want that a uniform distribution of points on the cylinder is mapped to a quasi uniform distribution of points on the tubular surface. The results in section 4. will show that minimizing the stretch [SGSH02] of the map, will produce a map with a balanced tradeoff between the semantics and the uniformity property.

The remainder of this paper is divided into the following sections: Section 2. contains some theory about parameterizations and the cylinder that is important for the rest of the paper. Section 3. explains our approach to the parameterization of tubular surfaces on the cylinder. Section 4. shows some results obtained with an implementation of our technique. Some surfaces together with their cylindrical parameterization and also some cylindrical geometry images are shown. Section 5. concludes the paper and suggests directions of future research.

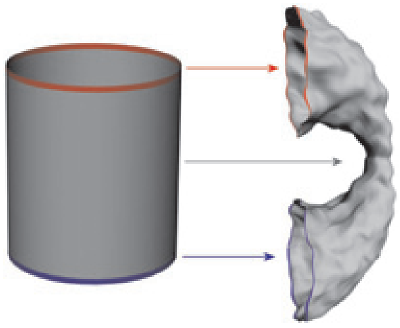
Figure 2: The upper boundary of the cylinder is mapped to the red boundary of the tubular surface and the lower boundary is mapped to the blue boundary of the tubular surface. The interior of the cylinder surface is mapped to the (grey) interior of the tubular surface.

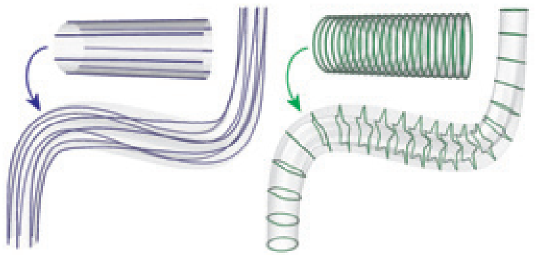
Figure 3: Semantics of the cylinder: Axial and radial lines on the cylinder are mapped to axial and radial lines on the tubular surface.

2. THEORY

This section introduces some basic differential geometry notions and explains some of the geometric properties of the cylinder, which are important to understand the rest of the paper. Most of this can be found in an elementary differential geometry book, for example [dC76].

Parameterization Informally, a parameterization of a surface $\mathcal{M}$ is a bijective map from a domain $\mathcal{D}$ to the surface $\mathcal{M}$. Mostly, $\mathcal{D}$ is a simple mathematical surface, for example the plane [SGSH02] or the sphere [PH03].

More formally, a parameterization of the surface $\mathcal{M}$, on the domain $\mathcal{D}$, is a *homeomorphism* $\mathbf{\Phi}$ between $\mathcal{D}$ and $\mathcal{M}$. The domain $\mathcal{D}$ is chosen so that it is *homeomorphic* to $\mathcal{M}$. This leaves us to explain the terms homeomorphic and homeomorphism: suppose $\mathcal{D}$ and $\mathcal{M}$ are topological spaces, and $\mathbf{\Phi}$ is a function from $\mathcal{D}$ to $\mathcal{M}$. Then $\mathbf{\Phi}$ is a homeomorphism iff the following holds:

- $\mathbf{\Phi}$ is a bijection;
- $\mathbf{\Phi}$ is continuous;
- the inverse function $\mathbf{\Phi}^{-1}$ is continuous.

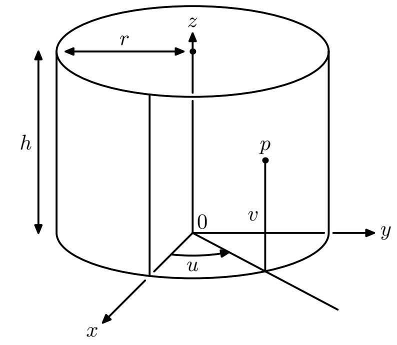

Figure 4: The right circular cylinder of radius r and height h. A point p on the lateral surface of the cylinder is defined by a cylindrical coordinate pair (u, v).

If there exists a homeomorphism $\mathbf{\Phi} : \mathcal{D} \mapsto \mathcal{M}$, then $\mathcal{M}$ is said to be homeomorphic to $\mathcal{D}$; $\mathcal{D}$ is also homeomorphic to $\mathcal{M}$, since $\mathbf{\Phi}^{-1}$ is a homeomorphism.

In our specific setting of parameterization of tubular surfaces, we choose $\mathcal{D}$ as the surface of the cylinder and $\mathcal{M}$ the surface of the tubular object. We say that $\mathbf{\Phi}$ is a cylindrical parameterization of the tubular surface $\mathcal{M}$.

So, in this paper we are concerned with the automatic construction of such a homeomorphism for any surface homeomorphic to the cylinder.

The Right Circular Cylinder There are many definitions for the concept cylinder, but we choose a specific one: the right circular cylinder. This cylinder is depicted in figure 4. The base of the right circular cylinder is a circle of radius r and the centers of the sections form a straight line perpendicular to the base of the cylinder. We choose the lateral surface of the cylinder as our parameterization domain $\mathcal{D}$; it is parameterized by $\mathbf{c} : U \mapsto \mathbb{R}^3$:

$$\begin{aligned} U &= \{(u, v) \in \mathbb{R}^2 | 0 \leq u < 1, 0 \leq v \leq 1\} \\ \mathbf{c}(u, v) &= (r\cos(2\pi u), r\sin(2\pi u), v). \qquad (1) \end{aligned}$$

Geodesic Triangulation of the Cylinder In this work we are only concerned with the parameterization of piecewise linear triangle surfaces. This has the interesting side effect that we do not have to calculate the parameterization $\mathbf{\Phi}$ for every point explicitly. If we define the parameterization for the vertices and the edges of the triangle surface, then the parameterization of all other points can be found using interpolation.

We choose the parameterization of an edge between two vertices on the surface to be a geodesic of the cylinder that connects the parameterization of those two vertices. We choose a geodesic because it is a locally length minimizing curve. On the cylinder, each geodesic γ is a helix, a circle parallel to the base, or a line perpendicular to the base, defined by:

$$\begin{aligned} \gamma(t) &= (r\cos{(at + b)}, r\sin{(at + b)}, ct + d), \\ & \quad a, b, c, d \in \mathbb{R}. \end{aligned}$$

There are an infinite number of geodesics between any two points on the cylinder, each with a different number of turns or a different direction. In figure 5 there are three geodesics (helices) all connecting the same two points. The solid line is the shortest geodesic of all possible geodesics between p_1 and p_2. It is important to specify the geodesic for each edge in the parameterization. How we do this will become clear in section 3.

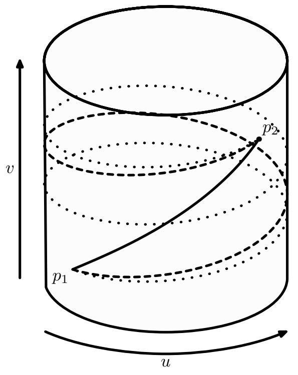

Figure 5: Three geodesics of the cylinder between points p_1 and p_2. The solid is the shortest geodesic, the dashed adds one turn in the positive u-direction and the dotted adds two turns.

If we parameterize the vertices and the edges of the surface onto the cylinder with the same connenctivity as the surface mesh and if the resulting triangles on the cylinder are not overlapping, then we get a triangulation of the cylinder. This, we call a *geodesic triangulation* because the edges of the triangles are geodesics of the cylinder. Such a triangulation induces a map from points on the cylinder to points on the surface. It is clear that this map is bijective, continuous, and its inverse is also continuous: it is a homeomorphism and thus also a parameterization.

Local Cylinder-Plane Isometry As we already mentioned in the introduction, most parameterization algorithms have the plane as their parameterization domain; calculations done in this plane are mostly fast and easy. When the parameterization domain is not flat, the computations can be harder. For example in [PH03] the domain is the sphere and calculations involve numerical integration which slows down the parameterization process.

A surface is flat if it has zero gaussian curvature, for example the plane. To check that the cylinder is flat, we compare the first fundamental form of the cylinder with the first fundamental form of the xy-plane.

If they coincide, then the cylinder is isometric to the plane. As a concequence the cylinder has zero gaussian curvature and therefore is flat.

The cylinder is parameterized by $\mathbf{c}$ in (1) and the xy-plane on the other hand is parameterized by $\mathbf{p} : \mathbb{R}^2 \mapsto \mathbb{R}^3$:

$$\mathbf{p}(u, v) \quad = \quad (u, v, 0) \tag{2}$$

The first fundamental form of the cylinder is given by:

$$E_c = \left|\frac{\partial \mathbf{c}}{\partial u}\right|^2 = r, \quad F_c = \frac{\partial \mathbf{c}}{\partial u} \cdot \frac{\partial \mathbf{c}}{\partial v} = 0, \quad G_c = \left|\frac{\partial \mathbf{c}}{\partial v}\right|^2 = 1$$

The first fundamental form of the xy-plane is given by:

$$E_p = \left|\frac{\partial \mathbf{p}}{\partial u}\right|^2 = 1, \quad F_p = \frac{\partial \mathbf{p}}{\partial u} \cdot \frac{\partial \mathbf{p}}{\partial v} = 0, \quad G_p = \left|\frac{\partial \mathbf{p}}{\partial v}\right|^2 = 1$$

We can see that the only difference between the first fundamental forms is between $E_c = r$ and $E_p = 1$, but if we choose the radius of the cylinder to be $r = 1$ then the first fundamental forms coincide.

This means that the cylinder and the plane are *locally* isometric, yet they are not *globally* isometric because the plane and the cylinder are not homeomorphic. This local isometry can be grasped visually: by cutting the cylinder along a line perpendicular to the base, the cylinder can be unfolded to the plane without distortion. This property has several interesting consequences.

First of all, due to the isometry, every geodesic of the cylinder corresponds to a geodesic of the plane and vice versa. The geodesics of the plane are all straight lines, so a geodesic triangle of the cylinder corresponds to a straight-line triangle in the plane. Now, if we have to apply an algorithm to geometry on the cylinder, we can simply transform the geometry from the cylinder to the plane by the isometry and apply ordinary algorithms to the planar geometry. Once the result is obtained in the plane, it can be transformed to the result on the cylinder.

Another advantage of working with the corresponding plane geometry, is that we can use ordinary 2d-optimization algorithms, like the conjugate gradient algorithm, for optimization of the vertex positions. Also, during the optimization of the vertex positions, we have to calculate the distortions of a geodesic triangle caused by the parameterization. But thanks to the isometry, this distortion can be calculated using the corresponding triangle in the plane. This means that we can calculate the distortion, using the formulas from planar parameterization algorithms as in [SGSH02].

3. METHOD

The parameterization can be computed in two steps: first, find a geodesic triangulation of the cylinder using the connectivity of the tubular surface so that we have a homeomorphism. Second, optimize the positions of the vertices on the cylinder so that the distortion of the parametrization is minimized. Although this method is correct, it has the disadvantage that the optimization step is very hard and that it will probably get stuck in a bad local minimum. It is better to construct the parameterization in a hierarchical way, as in [HGC99] and [SGSH02]. The hierarchical parametrization utilizes the progressive mesh of the tubular surface and proceeds as follows: first the base mesh is parameterized and then we iteratively split the vertices and locally optimize their placement while avoiding foldovers. This method is outlined in the following algorithm:

Algorithm 1 Parameterize($\mathcal{M}$)

```
1: ($\mathcal{M}_0$, {vsplit$_1$, ..., vsplit$_m$}) = ProgMesh($\mathcal{M}$);
2: $\mathcal{P}_0$ = ParameterizeBaseMesh($\mathcal{M}_0$);
3: $i_{prev} = 0$;
4: for $i = 1$ to $m$ do
5:     $\mathcal{P}_{i-1} \xrightarrow{\text{vsplit}_i} \mathcal{P}_i$;
6:     place new vertex $v$ inside kernel of its 1-ring;
7:     OptimizePlacement($v$);
8:     if $\#P_i > factor \times \#P_{i_{prev}}$ then
9:         OptimizePlacement() for all $v$ in $\mathcal{P}_i$;
10:        $i_{prev} = i$;
11:    end if
12: end for
13: OptimizePlacement() for all $v$ in $\mathcal{P}_m$;
14: return $\mathcal{P}_m$;
```

We will now go into more detail:

Progressive Mesh Construction We first construct the progressive mesh [Hop96] of our surface $\mathcal{M}$ using a quadratic error metric. A progressive mesh is constructed by successively collapsing an edge of the mesh; the next edge to collapse is chosen so that the introduced quadratic error metric is minimal and that the collapse does not violate any constraints. We impose three constraints:

- Both boundaries of the tubular surface should have at least three vertices.
- Collapse a boundary vertex only into a vertex of the same boundary. This avoids that a vertex of one boundary is collapsed into a vertex of the other boundary, which would generate a degenerate mesh. We also require this out of convenience, because this way we know that an internal vertex can never be split into a boundary vertex which eases the parametrization process.
- The third constraint says that there may be no triangles with all three vertices on one boundary,

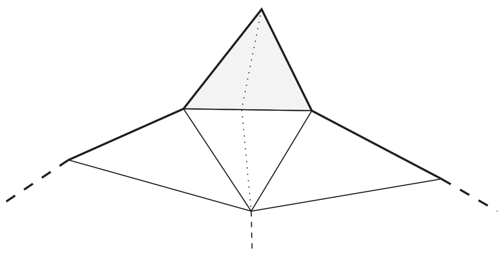
Figure 6: The shaded triangle is violating the third constraint because its three vertices are on one boundary (bold line). We remedy this by splitting the edge that is not on the boundary, this results in two extra triangles.

because the parametrization of such a triangle would result in a triangle of zero area, which is undesirable.

We have added these constraints to the progressive mesh construction algorithm. We also require that the original surface does not violate any of the above constraints. If the first or the second constraint is violated in the original surface, then we reject the mesh. When the third constraint is violated in the original surface, we have a remedy: split the edge of the triangle that is not on the boundary, this is depicted in Figure 6.

The progressive mesh is represented by the base mesh $\mathcal{M}_0$ and a set of vertex splits $\{\text{vsplit}_1, \text{vsplit}_2, \ldots, \text{vsplit}_m\}$, which are the reverse operations of the edge collapses in reversed order.

Base Mesh parameterization If we construct the progressive mesh of a tubular surface, as explained in the previous section, then the base mesh $\mathcal{M}_0$ will be an open prism with a triangle as its base. This mesh is depicted in Figure 7. Each of the three square sides of the base mesh consists of two triangles. This mesh is parameterized on the cylinder by separating the three points on both boundaries by 120 degrees. Then the vertices on one of the boundaries are rotated until three of the edges, connecting both boundaries, are perpendicular to the base of the cylinder. The (u, v)-coordinates of the parameterized base mesh are displayed in Figure 7.

We also have to determine the parameterization of the edges; the parameterization of an egde is a geodesic of the cylinder. A geodesic can be determined by specifying its direction (negative or positive u-direction) and its number of turns (0,1,2,...). The parameterization of an edge $((u_1, v_1), (u_2, v_2))$ of the base mesh is always a geodesic with 0 turns, because the length of the edge is at most $1/3$ in the u-direction, and its direction is positive if $u_1 <= u_2$ and negative otherwise.

In this way we obtain a geodesic triangulation of the

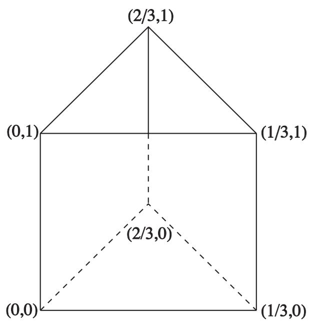

Figure 7: Base mesh of the progressive mesh for a tubular object, together with the (u, v)-coordinates of its parameterization.

cylinder with the connectivity of the base mesh, and thus we have found the parameterization $\mathcal{P}_0$ of the base mesh.

Next Level Parameterization Once we have the parameterization of the base mesh, we start by iteratively refining the resolution of the parameterization using vertex splits until we end up with the parameterization $\mathcal{P}_m$ of $\mathcal{M}_m = \mathcal{M}$. The step we explain here is generic and takes us from a parameterization $\mathcal{P}_i$ to the parameterization $\mathcal{P}_{i+1}$.

We start by applying vsplit_i to $\mathcal{P}_i$, this results in a new vertex v. In order to avoid foldovers we have to put this vertex inside the kernel of the polygon formed by the triangles of its 1-ring. We will put the vertex v in the center of this kernel. The kernel is computed in the plane using the isometry. But first we will have to transform the polygon to the plane. We set the y-coordinates of the planar polygon equal to the v-coordinates of the cylindrical polygon. We then choose one point of the polygon as a reference and set its x-coordinate to 0. Then we determine the x-coordinate of the next vertex in the polygon by calculating the u-length of the geodesic edge between this vertex and the reference vertex(taking into account the direction and the number of turns of the geodesic). Then the u-coordinate of the next vertex is determined relative to the previous vertex until al vertices are assigned a u-coordinate and we have obtained the planar version of our geodesic polygon.

We construct the kernel of this planar polygon using line clipping and calculate its geometric center. Then we transform the center to the cylinder and use this coordinate as the placement for v. We transform the center from the plane to the cylinder using a vertex of the polygon as a reference. We also update the direction and number of turns of each of the edges incedent to the vertex v. We now have a parameterization of $\mathcal{M}_{i+1}$.

In order to obtain a parameterization that is a balanced trade-off between the semantics and the uniformity property (see section 1.), we have to optimize the parameterization. After we have split a vertex, the placement of the new vertex v will be optimized, then we optimize the placement of each of its neighbours and we end with optimizing the new vertex v again. Also, when the number of vertices in the parameterization has increased with a factor (for example 1.5), we do this optimization for each of the vertices of the parameterization. A single vertex is optimized by the following steps:

1. transform the vertex v and its 1-ring polygon to the plane;
2. use the current position of v as an initial guess for the optimization;
3. minimize the symmetric version of the geometric stretch of the barycentric map summed over the 1-ring triangles as defined in [SGSH02]. The calculation of geometric stretch is based on the Jacobian of the barcentric map, since the Jacobian is invariant to isometry we can calculate the stretch using the planar triangles instead of the geodesic triangles of the cylinder. The optimization of the metric is also done in the plane using a standard 2D-optimization routine, while constraining the position of v to the kernel of the 1-ring polygon in order to avoid foldovers;
4. in the end, transform the optimized position of v back to the cylinder and update the direction and number of turns of each geodesic incident to the optimized vertex.

There is one remark we have to make: when we parameterize tubular objects that are very long in the axial direction compared to the radial direction, the parameterization gives bad results since the triangles are compressed in the axial direction to fit on the cylindrical domain of length 1. This can be remedied by changing the length of the cylindrical domain. For example when parameterizing a tubular surface that is twice as long in the axial direction as it is in the radial direction, we have to set the length of the cylindrical domain to the double of the radius of the cylindrical domain. Currently this length has to be estimated by the user, in the future we hope to automate this.

Sampling the Parameterization Up till now we have only defined the parametrization of the vertices and the edges. If we would like to sample the parametrization at arbitrary points of the cylindrical domain, then we have to define the parameterization at every point. As we have seen in the previous section, the interior of a triangle is parameterized using the barycentric map. Therefore if we want to sample the parameterization on the point (u, v) we only have to find the geodesic triangle on the cylinder that contains the point (u, v), the value of the parametrization is then determined by the barycentric map from that triangle to the corresponding triangle on the tubular surface. To find the triangle containing the point (u, v), we utilize a point location technique using bounding volume hierarchies [GLM96].

surface	# faces	h	time (s)
knot	12768	7.0	55
pipe	23248	3.0	117
head	11538	1.0	64
bow	33702	2.0	143
spring	19152	10.0	89
screwdriver	53782	3.0	268

Table 1: parameterization results of surfaces from $11K$ to $50K$ faces within 1 to 5 minutes. The height of the cylinder (h) ranges from 1 to 10 times the radius of the cylinder.

4. RESULTS

We have tested an implementation of the algorithm on different tubular surfaces, the results are summarized in Table 1. We have parameterized surfaces with $11K$ to $50K$ faces, within 1 to 5 minutes on a $1.2GHz$ computer. In Figure 8 the parameterized surfaces are displayed. The parameterization is revealed by the texture of the surface, the blue and the red lines on the surface are the iso-parameter lines for respectively the u and v parameter. Also, the quality of the parameterization can be derived from this figure. First, the semantics of the cylinder are ported to the surfaces because the red lines (iso-u) are running in the radial direction and the blue lines (iso-v) are running in the axial direction. Second, the distortion is kept low, which we can see because the iso-parameter lines form squares or rectangles. However, the size of the squares or rectangles can vary on the same surface (for example on the screwdriver), which tells us that the parameterization suffers from scale distortion. This is unavoidable when parameterizing onto the cylinder. This is also the reason why we did not add the stretch of the parameterizations in Table 1, there would be no point in comparing them.

Once a parameterization is obtained it is also possible to generate a geometry image [GGH02] of the surface. Geometry images are a completely regular image based surface representation with implicit connectivity. They have a number of applications: hardware accelerated rendering, adaptive remeshing, compression, etc. Our geometry images are constructed by

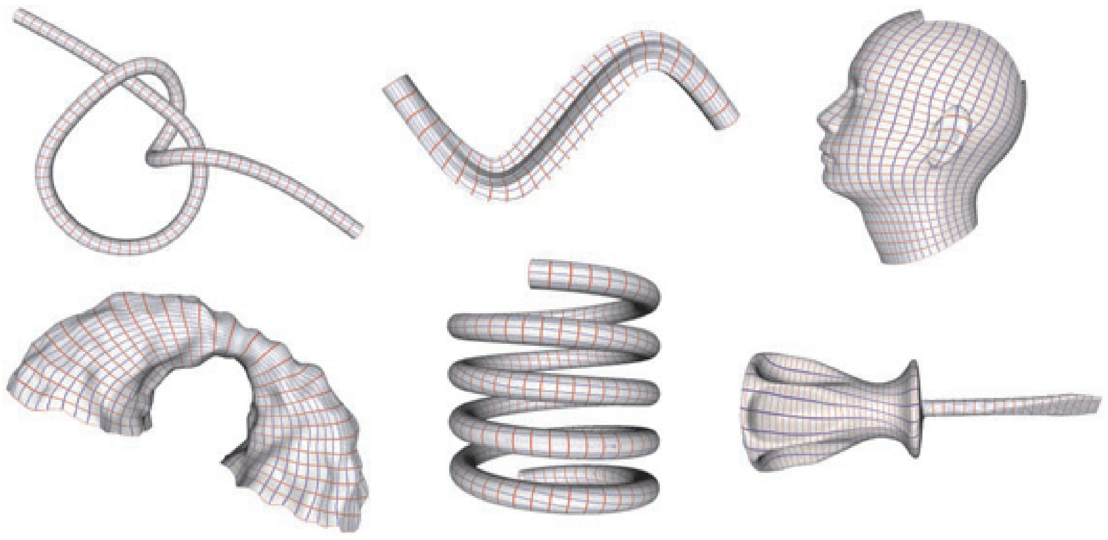

Figure 8: parameterization results, from left to right and from top to bottom: a knot, a pipe with a cross section that morphs from a circle to a star and back to a circle, a head with the bottom of the neck open and a square hole in the top, a bow, a spring, and a screwdriver with a hole in the tip and in the top. The texture visualizes the iso-parameter lines of the parameterization.

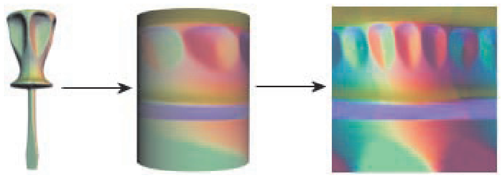

Figure 9: A geometry image is generated by parameterizing the surface on the cylinder and unfolding the cylinder to the plane.

sampling the cylindrical parameterization on a regular (u, v)-grid and unfolding this grid to the plane, this process is visualized in Figure 9. One side-effect of cylindrical geometry images is that the u and v resolution can be controlled separately. This results in rectangular geometry images, which can be useful for elongated surfaces. Figure 10 displays the cylindrical geometry image and normal map of the bow surface at different resolutions.

5. CONCLUSION

In this paper we propose a new method to parameterize tubular surfaces. We parameterize the surfaces on their natural domain i.e., the cylinder, which avoids cutting. By minimizing our symmetric stretch metric we obtaine a parameterization with a balanced trade-off between cylindrical semantics and uniform sampling. We test the algorithm on several surfaces and summarize the results. We also propose a new kind of geometry images for cylindrical surfaces and show some results.

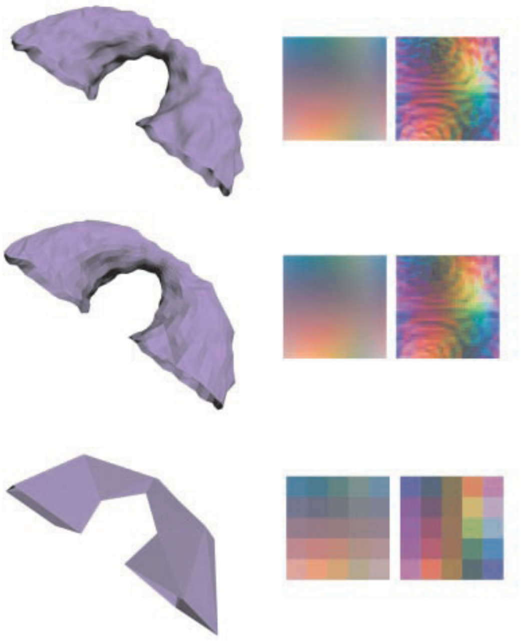

Figure 10: Cylindrical geometry image and normal map of the bow surface at following resolutions: 257×257, 33×33, 5×5. The remesh of the geometry image at each resolution is displayed on the left and is flat shaded.

Future directions of research: we would like to find a method to automatically determine the optimal length of the cylinder when parameterizing a surface or adapt the parameterization method so that we can use a cylinder of unit length without artifacts. We would also like to extend the parameterization method for feature correspondance. This should enable us to use the tubular parameterization for shape analysis of biological tubular objects as for example the human cochlea.

ACKNOWLEDGMENTS

This work was financially supported by the Flemish Institute for the Promotion of scientific and Technological Research in the Industry (I.W.T.) - Brussels, Belgium and the Fund for Scientific Research (F.W.O.) - Flanders, Belgium.

References

[Ale02] Marc Alexa. Recent advances in mesh morphing. *Computer Graphics Forum*, 21(2):173–196, June 2002.

[AMD02] Pierre Alliez, Mark Meyer, and Mathieu Desbrun. Interactive geometry remeshing. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, pages 347–354. ACM Press, 2002.

[BGK95] Ch. Brechbühler, G. Gerig, and O. Kübler. Parametrization of closed surfaces for 3-D shape description. *Computer Vision and Image Understanding: CVIU*, 61(2):154–170, March 1995.

[dC76] Manfredo P. do Carmo. *Differential Geometry of Curves and Surfaces*. Prentice-Hall, 1976. 503 pages.

[EHL+95] Matthias Eck, Hugues Hoppe, Michael Lounsbery, Tom Duchamp, Tony DeRose, and Werner Stuetzle. Multiresolution analysis of arbitrary meshes. *Proc. Conf. on Computer Graphics (SIGGRAPH '95)*, pages 173–182, January 1995.

[GGH02] Xianfeng Gu, Steven J. Gortler, and Hugues Hoppe. Geometry images. In John Hughes, editor, *SIGGRAPH 2002 Conference Proceedings*, Annual Conference Series, pages 335–361. ACM Press/ACM SIGGRAPH, 2002.

[GGS03] C. Gotsman, X. Gu, and A. Sheffer. Fundamentals of spherical parameterization for 3d meshes. *ACM Transactions on Graphics*, 22, 2003.

[GLM96] S. Gottschalk, M. C. Lin, and D. Manocha. Obbtree: A hierarchical structure for rapid interference detection. In *Proc. SIGGRAPH '96*, pages 171–180, 1996.

[GWC+03] X. Gu, Y. Wang, T. Chan, P. Thompson, and S. Yau. Genus zero surface conformal mapping and its application to brain surface mapping. In *Information Processing Medical Imaging 2003*, 2003.

[HGC99] K. Hormann, G. Greiner, and S. Campagna. Hierarchical parametrization of triangulated surfaces. In B. Girod, H. Niemann, and H.-P. Seidel, editors, *Proceedings of Vision, Modeling, and Visualization 1999*, pages 219–226, Erlangen, Germany, November 1999. infix.

[Hop96] Hugues Hoppe. Progressive meshes. *Proc. 23rd Int'l. Conf. on Computer Graphics and Interactive Techniques (SIGGRAPH'96)*, pages 99–108, January 1996.

[KS04] Vladislav Kraevoy and Alla Sheffer. Cross-parameterization and compatible remeshing of 3d models. *ACM Trans. Graph.*, 23(3):861–869, 2004.

[LPRM02] Bruno Levy, Sylvain Petitjean, Nicolas Ray, and Jerome Maillot. Least squares conformal maps for automatic texture atlas generation. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, pages 362–371. ACM Press, 2002.

[PH03] Emil Praun and Hugues Hoppe. Spherical parametrization and remeshing. *ACM Transactions on Graphics*, 22(3):340–349, July 2003.

[SAPH04] John Schreiner, Arul Asirvatham, Emil Praun, and Hugues Hoppe. Inter-surface mapping. *ACM Trans. Graph.*, 23(3):870–877, 2004.

[SD02] Jan Sijbers and Dirk Van Dyck. Efficient algorithm for the computation of 3D fourier descriptors. In Guido M Cortelazzo and Concettina Guerra, editors, *Proceedings of the 1st International Symposium on 3D Data Processing Visualization and Transmission (3DPVT-02)*, pages 640–643, Los Alamitos, CA, June 29–21 2002. IEEE Computer Society.

[SGSH02] Pedro V. Sander, Steven J. Gortler, John Snyder, and Hugues Hoppe. Signal-specialized parametrization. In *13th Eurographics Workshop on Rendering*. Eurographics Association, 2002.

[Sty01] Martin Styner. *Combined Boundary-Medial Shape Description of Variable Biological Objects*. PhD thesis, University of North Carolina, Dept. of Computer Science, June 2001.

[ZSH00] Malte Zöckler, Detlev Stalling, and Hans-Christian Hege. Fast and intuitive generation of geometric shape transitions. *The Visual Computer*, 16(Issue 5):241–253, 2000.