Interactive Design and Visualization of Branched Covering Spaces

Lawrence Roy, Prashant Kumar, Sanaz Golbabaei, Yue Zhang, *Member, IEEE*,
and Eugene Zhang, *Senior Member, IEEE*

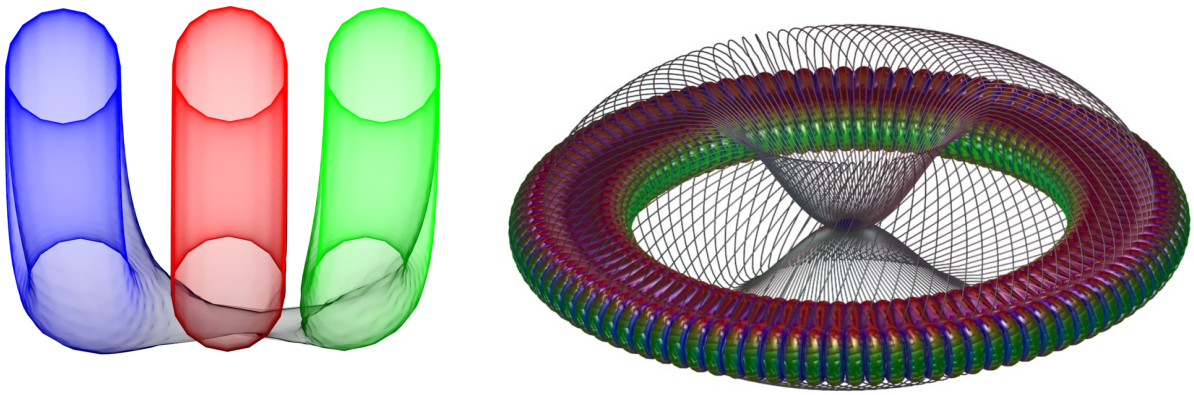

Fig. 1. Two branched covering spaces (BCS) generated using our design system: (left) a three-fold covering of the torus visualized using depth peeling [4] and (right) a 100-fold covering of the double torus using connecting tubes (Section 4.2.2). Our system enables researchers new to tensor field topology and quadrangular remeshing to develop geometric understanding of important properties of the BCSs.

Abstract—Branched covering spaces are a mathematical concept which originates from complex analysis and topology and has applications in tensor field topology and geometry remeshing. Given a manifold surface and an N-way rotational symmetry field, a branched covering space is a manifold surface that has an N-to-1 map to the original surface except at the *ramification points*, which correspond to the singularities in the rotational symmetry field. Understanding the notion and mathematical properties of branched covering spaces is important to researchers in tensor field visualization and geometry processing, and their application areas. In this paper, we provide a framework to interactively design and visualize the branched covering space (BCS) of an input mesh surface and a rotational symmetry field defined on it. In our framework, the user can visualize not only the BCSs but also their construction process. In addition, our system allows the user to design the geometric realization of the BCS using mesh deformation techniques as well as connecting tubes. This enables the user to verify important facts about BCSs such as that they are manifold surfaces around singularities, as well as the *Riemann-Hurwitz formula* which relates the Euler characteristic of the BCS to that of the original mesh. Our system is evaluated by student researchers in scientific visualization and geometry processing as well as faculty members in mathematics at our university who teach topology. We include their evaluations and feedback in the paper.

Index Terms—Tensor field topology, math visualization, branched covering spaces visualization, rotational symmetries, ramification points

1 Introduction

The *branched covering space* is a mathematical concept in topology which has found applications in tensor field topology and remeshing.

Tricoche [28] describes the behavior of a *degenerate point* in a 2D symmetric tensor field defined on some domain by converting the tensor field into a vector field, defined on the branched covering space of the domain, and by studying the behavior of the singularity in the vector field that corresponds to the degenerate point in the tensor field.

In geometry processing, the problem of quadrangular remeshing, i.e., the generation of a mesh of quads from an input triangle mesh with a guiding directional field, has gained much attention [5]. In quadrangular remeshing, the edges in the quads are often required to be approximately aligned with a given *cross* field that is usually derived from the principal curvature directions in the underlying surface. To prevent T-junctions from occurring in the quad mesh, Kälberer et al. [13] propose to lift the cross field in the input mesh to a vector field on the four-fold branched covering space of the surface. They then perform Hodge-decomposition to remove the divergence-free part of the vector field, thus preventing T-junctions in the remesh. Campen et al. [8] generate a coarse quad layout for the quadrangulation of a triangle surface with a cross field by computing crossing loops on the branched covering space. This leads to robust generation of quad layouts which facilitates pure-quad remeshing. Nieser et al. [17] perform hexahedral remeshing by also using the notion of branched covering spaces, this time of a volume and a 3D frame field.

The concept of branched covering spaces, which we will refer to

- *Lawrence Roy is with Roy Family Homeschool. E-mail: ldr709@gmail.com.*
- *Prashant Kumar is with Oregon State University. E-mail: kumarp@eecs.oregonstate.edu.*
- *Sanaz Golbabaei is with Oregon State University. E-mail: golbabas@eecs.oregonstate.edu.*
- *Yue Zhang is with Oregon State University. E-mail: zhangyue@oregonstate.edu.*
- *Eugene Zhang is with Oregon State University. E-mail: zhange@eecs.oregonstate.edu.*

Manuscript received 31 Mar. 2017; accepted 1 Aug. 2017.
Date of publication 28 Aug. 2017; date of current version 1 Oct. 2017.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TVCG.2017.2744038

1077-2626 2017 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

as *BCS* in the remainder of the paper, is important to researchers in the fields of tensor field visualization and remeshing. However, our experience is that it can be difficult for graduate student researchers in visualization and geometry processing to fully grasp the concept and important mathematical properties of the BCSs.

Mathematically, a BCS is the extension of the concept of covering space, with the additional notion of *ramification points*. While this addition can seem minor, the behaviors of a BCS are significantly more complex than that of a covering space and thus more difficult to understand. The additional complication of the BCSs has led to misconceptions such as that the BCS of a manifold surface is no longer a manifold surface due to the presence of ramification points. In addition, given an input surface and its branched cover, their topologies are linked by the *Riemann-Hurwitz formula* which depends on the number and type of ramification points. While this formula makes it possible to predict the topology of the BCS given an input surface and an N-way rotational symmetry field defined on it, developing an intuitive understanding of the Riemann-Hurwitz formula is a challenging task.

Existing research that requires some visual descriptions of a BCS [13, 28] often does so with hand-drawn illustrations of some patch on the BCS, usually around a ramification point. Moreover, such illustrations are usually in the form of static images, using shapes with relatively simple geometry and topology. Our experience with student researchers new to quad remeshing suggests a need to not only see the BCSs but also interact with them for models with more complex structures than a sphere or a torus.

There has been relatively little research in the construction and visualization of BCSs. Nieser et al. [16] develop an algorithm to construct the BCSs of a holomorphic function defined on the complex plane. They also develop a visualization technique of the BCSs by replicating the complex plane a number of times, placing them in parallel positions in 3D, and connecting them in the vicinity of ramification points. However, a naive adaptation of this approach to an arbitrary surface with complex geometry and topology is inadequate, due to the large amount of self-intersections in the BCS as demonstrated in Figure 1 (left). Moreover, while the topology of the BCS is determined by the topology of the input surface and the guiding rotational symmetry field, the geometric realization of the BCS has many degrees of freedom.

In this paper, we introduce an interactive design and visualization system for the BCSs given an arbitrary 3D surface and an N-way rotational symmetry field defined on it. Our system allows the user to not only visualize the BCS and the lifted vector field interactively with a number of options but also design the geometric realization of the BCS. To overcome the difficulties associated with self-intersections in the BCS, we introduce a method to connect different layers in the BCS using tubes that connect through *docking stations* (Section 4.2.2). We also allow the user to examine the neighborhood around a ramification point or a group of ramification points as well as that of a handle in the BCS through mesh deformation and animation. These techniques make it possible to visualize a number of important mathematical properties of a BCS, such as:

1. An N-way rotational symmetry (N-RoSy) field on the input mesh leads to an N-fold branched covering space of the original mesh.
2. The singularities in the N-RoSy field become the ramification points of the BCS.
3. Away from the ramification points, every point in the original mesh corresponds to N points in the BCS, each of which is assigned one of the vectors in the N-RoSy at the base point.
4. If the input mesh represents a manifold surface, then the BCS is also a manifold surface, i.e., the points in the BCS corresponding to a ramification point are manifold points.
5. The index of a singularity in the vector field on the BCS of the mesh is decided by the index of the corresponding singularity in the N-RoSy field.
6. *Riemann-Hurwitz formula*, which states that the Euler characteristic of the BCS is related to that of the base surface and the number of ramification points in the BCS.
7. The BCS is independent of the way the base mesh was cut open.
8. Any closed orientable surface with at least one handle is a two-fold cover for the sphere.

We also introduce the notion of *essential cut graph*, which is a *minimal* subgraph of the cut graph that is needed for BCS construction. Moreover, by investigating the structure of the essential cut graph, we are able to compute the correct structure in the docking station (Section 4.2.2).

We have conducted a user study in which graduate students in computer science as well as faculty members in mathematics evaluated our system. We report their evaluation in Section 10.

2 Related Work

The notion of branched covering spaces is first introduced into computer graphics in the context of quadrangular remeshing of surfaces [13].

Alliez et al. [1] point out the importance of quadrangular remeshing from a triangular mesh where the edges of the quads in the remesh follow the principal curvature directions of the underlying surface. To generate such a mesh, they adapt the evenly-spaced streamline placement approach by Jobard and Lefer [12] for vector fields to define the curvature tensor field of the surface. Ray et al. [21] point out that the distortion in the resulting quad mesh can be greatly reduced if the quads in the mesh are oriented according to the curvature tensor field, which has two mutually perpendicular directions (major and minor principal directions), and can be modeled as a four-way rotational symmetry (i.e., a cross). They further point out the difficulties associated with the singularities in the rotational symmetry field. Palacios and Zhang [18] provide a rotational symmetry field design system, with the ability to control the number and location of the singularities in any N-way rotational symmetry field for any $N \geq 1$. Ray et al. [22] introduce a mathematically rigorous algorithm that generates a smooth, per-face N-way rotational symmetry field given desired index and location of singularities in the mesh also for an arbitrary $N \geq 1$. Still, approaches such as tracing streamlines following an N-way rotational symmetry field will lead to *T-junctions*, leading to *quad-dominant* but not pure quad meshes.

Kälberer et al. [13] employ a global parameterization approach to quadrangulation that can eliminate T-junctions. The core of their approach is to lift the N-way rotational symmetry field on the input surface to a vector field on the branched covering space and then remove the divergence-free part in the vector field through Hodge-decomposition [20, 27]. This approach is reformulated into a mixed-integer quadrangulation approach [6]. Nieser et al. [15] apply a similar approach, but with 6-RoSy fields, to produce high-quality triangular meshes with control over the irregular vertices. Campen et al. [8] also use the notion of branched covering spaces to compute a coarse quad layout for the quadrangulation of a triangle mesh with a cross field.

In scientific visualization, the notion of branched covering spaces has been introduced by Xavier Tricoche [28] to explain the behaviors of the *degenerate points* in 2D symmetric tensor fields. These behaviors are understood by lifting the tensor field locally to a vector field on its branched covering space and examining the behavior of the singularity in the vector field corresponding to the degenerate point in the tensor field. A number of researchers [3, 16, 29] have constructed and visualized branched covering spaces where the base surface is the complex plane. While all this work provides an algorithm to construct the BCS of the complex plane (topologically a sphere minus the point ∞), it is not clear how such an approach can be extended to the surfaces of higher geometric and topological complexities without leading to a significant amount of self-intersections. Moreover, none of the branched covering space work provides the ability to design the geometric realization of the branched covering spaces.

In our work, we provide an interactive design and visualization system in which the branched covering space can be constructed for

high-genus surfaces and surfaces with high geometric complexities. In addition, anticipating potential applications of N-RoSy fields (N is not 1, 2, 4, or 6) in hyperbolic geometry, our system handles N-RoSy fields with an arbitrary $N \geq 1$. The user can design the geometric realization of the BCS, which allows the inspection of local behaviors around ramification points as well as handles. To overcome difficulties associated with a relatively large amount of self-intersections in the BCS, we also provide an approach in which the layers are connected through thin tubes as well as topological constructions we call "docking stations". This approach greatly reduces the amount of visual self-overlaps. In addition, we enhance the construction algorithm by using the concept of *essential cut graph*, which is a subset of the cut graph that existing BCS construction algorithms use to cut the mesh open. The essential cut graph not only reduces the computational cost during the BCS construction stage, but also is essential in enabling our tube-based visualization technique (Section 4.2.2).

3 Mathematical Background

In this section we review the background on (BCS). BCS is an extension of the notion of covering spaces, which we describe next [2].

Definition 1. *Let X be a topological space. A **covering space** of X is a topological space C together with a **continuous surjective** map: $p : C \to X$ such that for every $x \in X$, there exists an open neighborhood U of x, such that $p^{-1}(U)$ is a union of disjoint sets in C, each of which is mapped homeomorphically onto U by p. The map p is called the **covering map**, the space X is the **base space** of the covering, and the space C is called the **total space** of the covering. The pre-image of x is a set of discrete points in C, which are referred to as the **fiber** over x. The neighborhood U is referred to as an **evenly covered neighborhood**. Each homeomorphic copy in C of U is a* sheet *over U.*

An example covering space is $\mathbb{R}^1$, which provides an infinite cover of $\mathbb{S}^1$ (the unit circle in $\mathbb{R}^2$) through the map: $p(\theta) = \begin{pmatrix}\cos\theta \\ \sin\theta\end{pmatrix}$. Here $\mathbb{S}^1$ is the base space, while $\mathbb{R}^1$ is the total space.

A BCS extends the notion of covering spaces as follows:

Definition 2. *Let X and C be two topological spaces and $p : C \to X$ be continuous surjective map. C is said to be a **branched covering space** of X under p if there exists a nowhere dense set $\Delta \subset X$ such that $p|_{p^{-1}(X\setminus\Delta)} : p^{-1}(X \setminus \Delta) \to X \setminus \Delta$ is a covering mapping. The set $X \setminus \Delta$ is a **regular set** of the branched covering p, whereas Δ is the **singular set**.*

In the above definition "\" denotes set difference while "|" refers to when a function is restricted to a subset of its domain. Note that every covering mapping is also a branched covering mapping with an empty singular set. A less trivial example is $\mathbb{R}$ which covers the set of non-negative numbers $\mathbb{R}^+$ under the map: $p : \mathbb{R} \to \mathbb{R}^+$ $(p(x) = |x|)$. The singular set consists of the number 0 which has only one pre-image under p while other elements in $\mathbb{R}^+$ have two pre-images.

In geometry remeshing and tensor field topology, the branched covering mappings are usually induced from an input N-RoSy field on some base surface S.

Definition 3. *An N-RoSy is a set of N-vectors $s = \{\begin{pmatrix}R\cos(\theta + \frac{2k\pi}{N}) \\ R\sin(\theta + \frac{2k\pi}{N})\end{pmatrix} \mid 0 \leq k \leq N-1\}$. An N-RoSy field is a continuous N-RoSy valued function on the surface. A **singularity** in an N-RoSy field is a point in the domain where $R = 0$.*

A singularity can be characterized by its *singularity index*, which is defined in terms of the *winding numbers* and is a multiple of $\frac{1}{N}$.

Given an input surface S and an N-RoSy field defined on it, a branched covering space can be constructed that lifts the N-RoSy field to a vector field on the BCS in the following sense: (1) every regular point p in S corresponds to N points in the BCS, and (2) the collection of the vectors at points in the BCS that correspond to p is exactly the N-RoSy at p. The singularities in the N-RoSy field correspond to ramification points in the BCS, where the map between the base surface and the BCS has fewer-than-N pre-images. Figure 2 shows such a scenario when S is a torus with a 4-RoSy field with two singularities. Each singularity in the N-RoSy field is transformed into a ramification point in the BCS.

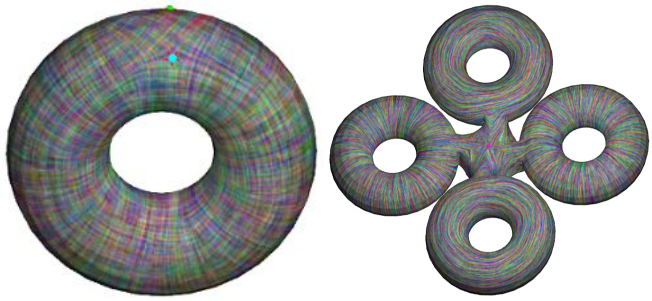

Fig. 2. A 4-RoSy field with one pair of singularities on the torus (left) leads to a covering space (right). Notice that the four directions at a point in the base mesh (left) are maintained by the four vectors, each at one of the corresponding points in the BCS (right). While static LIC images cannot convey the vector directions well, please see the supplementary video for dynamic flows that clearly differentiate between flows in opposite directions (e.g., corresponding points in top torus and bottom torus of (b)).

Branched covering spaces have a number of important properties:

1. A branched covering space is a manifold surface if the base surface is a manifold.
2. Every singularity of index k in the N-RoSy field is mapped to a singularity of index $Nk-(N-1)$ in the vector field in the branched covering space.
3. The Euler characteristic of the branched covering space is described by the *Riemann-Hurwitz formula* [11]: $\chi(B) = N \cdot \chi(S) - \sum_{p\in B}(e_p - 1)$, where $\chi(B)$ and $\chi(S)$ are the Euler characteristics of the branched covering space B and the base space S, respectively; and the summation term is over the set of pre-images of the singularity set where e_p is the index of each ramification point in the singularity set.
4. Every closed orientable surface with at least one handle is a two-fold branched cover for the sphere.

In the next sections we will describe our algorithm for the construction and interactive design of the branched covering space given an input surface with an N-RoSy field. In addition, we will provide details on a number of visualization and interaction techniques that allow users to gain intuitions about the aforementioned mathematical properties.

4 Branched Covering Spaces Construction

In this section, we describe the construction of the BCS given an input orientable manifold surface represented by a triangular mesh with a per-face N-RoSy field defined on it. That is, there is an N-RoSy defined on each face. Our algorithm consists of four stages. First, we compute the essential cut graph (Section 4.1) G on the input surface M and the *gap* for each edge in G, which is an integer between 0 and $N-1$ that describes how the vectors on different layers are assigned and how the layers are connected during the BCS construction process (Figure 3).

Second, we cut M open along G to obtain M'. Third, we replicate M' so that there are N layers $M_0, M_1, ..., M_{N-1}$, each of which is identical to M'. We then assign appropriate vector values at each face of each layer. Fourth, we connect M_0, M_1, M_{N-1} along their edges based on the gap computed for G, thus ensuring a manifold surface with a continuous vector field. Figure 3 illustrates this with an example.

On a high level, the four-step pipeline of our algorithm is similar to that of Nieser et al. [16]. However, there are a number of differences between the two approaches. Below we describe each step in detail and highlight the difference between our approach and that of Nieser et al. [16].

4.1 Essential Cut Graph Computation

Existing work in quad remeshing [6, 13] computes a cut graph G along which the base surface S is cut into a topological disk. This is achieved by region growing from a seed triangle, inside which a vector is selected from the N-RoSy of the triangle. During the region growing process, the chosen vector inside the seed triangle is parallel transported to the rest of the triangles in the mesh, which is used to select a vector from each of the remaining triangles to maximize vector field continuity across the edges. The boundary of the region (a topological disk) is the cut graph G, across which the selected vectors may not be consistent. Figure 4 (top row) illustrates this with an example in which the best match for the vector $V_{A,0}$ from the triangle A is $V_{B,1}$ in the triangle B. The *gap* from A to B is 1 in this case, which means that during the mesh stitching stage (Figure 4 (upper-right)), a duplicated triangle A_i is stitched to $B_{i+1 \bmod 4}$. More precisely, the gap from a triangle A to an adjacent triangle B is the j $(0 \leq j \leq N-1)$ where $V_{B,j}$ is the best match for $V_{A,0}$. Existing work then computes the gap across edges in the cut graph G. These algorithms then start the next stage of the pipeline, i.e., mesh cutting along the cut graph G. We observe that for the purpose of BCS construction, it is unnecessary to cut the surface S along every edge in G. This is because unlike quad meshing in which all cohomological basis is needed, BCS construction only requires handles produced by singularities in the field. Therefore, we only need to cut along the edges in G where the gap is not zero. To see this, consider the example in the bottom row of Figure 4. In this case A_i will be stitched with B_i, which makes it unnecessary to cut the mesh open along the edge shared by

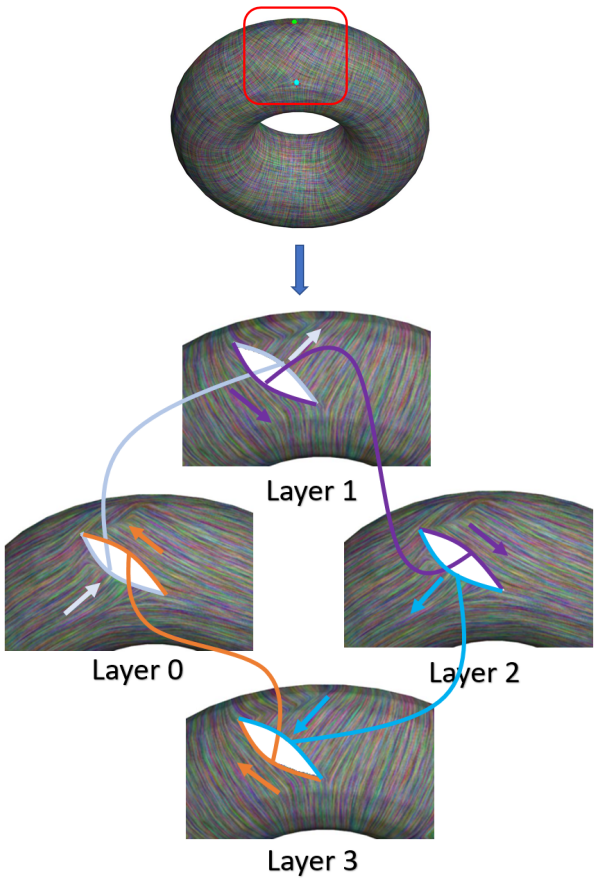

Fig. 3. Explanation of the construction process of branched covering spaces.

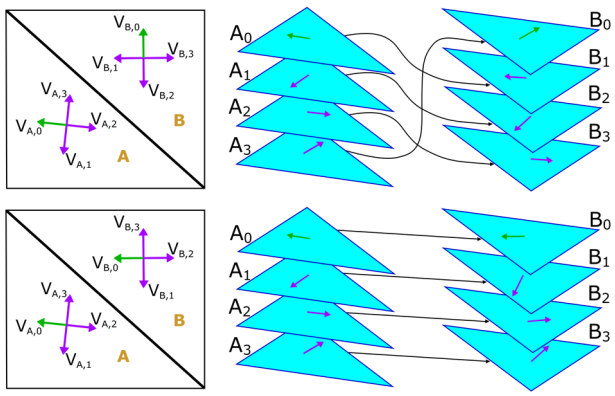

Fig. 4. This figure illustrates the concept of *gap* and its use in deciding how the layers are connected. In the left subfigure of the top row, the chosen vectors (green) in triangles A and B are indexed 0. The gap from A to B is 1 since the best matching vector in triangle B for $V_{A,0}$ is $V_{B,1}$. Therefore, during the layer stitching stage (right in the top row), A_i is stitched with $B_{i+1 \bmod 4}$. In contrast, the bottom row shows the case when the gap from A to B is zero. In this case, A_i will be stitched with B_i. Consequently, cutting the mesh open along the edge is unnecessary. We remove edges with a zero gap from the cut graph to obtain an *essential cut graph*, along which we cut the mesh open.

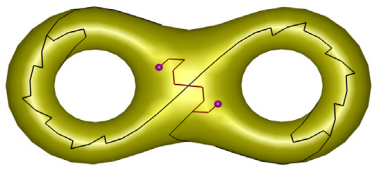

Fig. 5. The cut graph contains many edges over which the gap is 0 (black edges). The essential cut graph is a subgraph of the cut graph that consists of only edges whose gap index is not zero (red).

A and B. We define an essential cut graph G' to be a subgraph of a cut graph G where only edges with a non-zero gap are kept. Figure 5 compares the cut graph and the essential cut graph with an example. Essential cut graphs are *minimal* in the sense that removing any edge from an essential cut graph will lead to a topologically incorrect BCS. To extract the essential cut graph given a cut graph, we simply identify all the edges in the cut graph with a non-zero gap. Note that cut graphs are not unique given a surface and a RoSy field. Consequently, essential cut graphs are also not unique. In Section 4.2.2, we will discuss how the notion of essential cut graphs can be used to generate BCSs with fewer self-intersections and provide a more geometrically intuitive undertsanding behind the Riemann-Hurwitz formula.

Next we cut the mesh open along the essential cut graph computed in the first stage and replicate the cut-open mesh N times. For a triangle in the original mesh, its N-RoSy vectors are assigned to the corresponding triangles in each duplicated layer based on the aforementioned layer index. That is, the first vector is assigned to the triangle on the first layer, and the second triangle to the second layer, and so on. This results in a triangle mesh with N connected components and a vector field. Our algorithm in these two steps (cutting and replication) does not differ from existing work [16].

4.2 Layer Connection

In the last stage, we connect the N layers together to form the BCS. Our system provides two options for this step.

The first option is the same as [16], which is essentially an inverse process to the cutting stage except that there are more layers to stitch. The layers are initially placed in the 3D space. Given an edge on the original cut graph, the two triangles incident to the edge correspond

to $2N$ triangles in the replicated mesh. We will stitch triangle $t_{1,j}$ to $t_{2,L+j \bmod N}$ where L is the gap from triangle t_1 to t_2 and $0 \leq j < N$. Once all the edges have been glued together, we remove redundant vertices. This results in the BCS.

Note that stitching can introduce visual artifacts when the two layers to be stitched are relatively far apart. This is because the shared vertices, regardless of their final locations, will lead to some triangle with vertices located on both layers. While it is possible to place the shared vertices in between the two layers, the problem can still be visible. To remove such an artifact, ideally all the layers should be co-located, i.e., each vertex in the BCS has the same 3D coordinates as the corresponding vertex in the original input surface. However, in this case the BCS is identical to the input surface visually since all the layers are co-located.

To address these difficulties, we provide two approaches: *mesh deformation*, and *connecting tubes*.

4.2.1 Layer Connection through Mesh Deformation

In the first approach, the user can use our BCS design system to place the N layers anywhere in space with an arbitrary orientation and scaling. Next, the edges along which stitching occurs are also placed in space, usually between the layers. We then proceed to stitch the layers along these edges. However, we wish to maintain the overall separation among the layers while keeping their overall shape relatively undeformed so that they can be recognized as being similar to the base mesh and to each other.

To achieve this, our system computes a set of triangles in the base mesh that are relatively far away from the essential cut graph. These triangles are usually obtained by first computing the distance of each vertex in the base surface S to the essential cut graph and are selected based on user-provided threshold. The corresponding triangles in the layers will serve as hard constraints, along with the positions of the edges corresponding to the essential cut graph. The positions of vertices not defined by these hard constraints are then solved by Laplacian smoothing [26] with the hard constraints serving as the boundary condition.

The user can control the smoothness near the stitching by changing the position and orientation of each layer, the position and orientation of the edges to be stitched, and the number of triangles on each layer that will serve as hard constraints.

4.2.2 Layer Connection with Connecting Tubes

While the mesh deformation approach can lead to high-quality BCSs, such BCSs usually contain significant self-intersections which make the understanding of the BCS structure rather challenging. We now describe another approach which uses additional tubes to connect the layers.

Recall that the topology of the BCS does not depend on the cut graph. Consequently, given an N-RoSy field on the input surface, we can generate an essential cut graph that facilitates our layer connection. For educational purposes, we assume that no higher-order singularities exist in the N-RoSy field, i.e. the index of every singularity is either $\frac{1}{N}$ or $-\frac{1}{N}$. Let the *Euler characteristic* of the input surface be $\chi(S)$, then the set of singularities in the N-RoSy field can be considered as the disjoint union of two subsets. The first subset consists of $|N\chi(S)|$ singularities, each of which has an index of $sign(\chi(S))\frac{1}{N}$. The second set consists of $2M$ singularities ($M \geq 0$), half of which have an index of $\frac{1}{N}$ and the other half have an index of $-\frac{1}{N}$.

Our algorithm allows the user to edit the essential cut graph that it can not only shorten the total length of the edges in the essential cut graph but also increase the connectivity of the cut graph so that it will have $|\chi(S)|+M$ connected components. Each of the first $|\chi(S)|$ components consists of N singularities from the first subset, while each of the remaining M components consists of one positively-indexed singularity and one negatively-indexed singularity from the second subset. In this regard, the cut graph is further improved by connecting the singularity pairs with curves on the mesh surface following a geodesic path [14].

For each of aforementioned connected components in the essential cut graph, we will construct a *docking station* that is a terminal which helps in connecting the layers with tubes for each of the components of the essential cut graph.

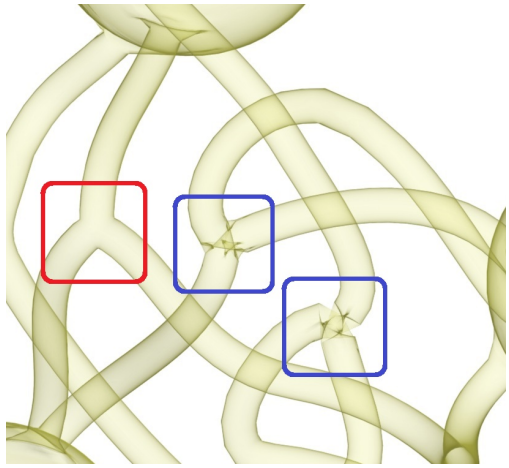

Fig. 6. An illustration of docking stations of different kinds. The docking station highlighted in red on the left is a topological sphere to which the tubes are connected. Others marked in blue are of higher genus.

Figure 6 shows the two types of docking stations with an example three-fold cover of the sphere. There are eight singularities in the original 3-RoSy field, forming three connected components in the essential cut graph: two groups of three positively-indexed singularities, and one group of one positively-indexed singularity and one negatively-indexed singularity. The docking station corresponding to the last group is relatively simple (enclosed by the red box), which is topologically three curves connecting two points that form three circular openings, with each opening connected to a layer through a tube. The first two groups each consists of three positively-indexed singularities, and their docking stations (enclosed by the blue boxes) are more complex. There are now three points in the docking station, representing the singularities in the connected component. There are three curves between the first and middle singularities on the cut graph, forming three holes (one per layer). There are also three curves connecting the middle and the last singularities, forming three other holes (again one per layer). Our system correctly generates the docking stations for both types of singularity groups for any arbitrary N-RoSy field.

5 Visualization of Branched Covering Spaces

Given a geometric realization of the BCS, our system provides a number of visualization options. Besides rendering the BCS as an opaque object using photorealistic shading of the normal maps [24], we also enable rendering of the BCS as translucent material using the dual depth peeling technique [4]. In addition, we allow the user to see the N-RoSy field on the original surface as well as the lifted vector field on the BCS using the technique of Palacios and Zhang [19], which is adapted from the original line integral convolution technique [7]. The texture can be further made dynamic to provide a sense of flow on the BCS based on the vector field.

These techniques can demonstrate important facts about the BCS, such as that there are mostly N copies of the original surface except at the ramification points and that the N-RoSy field is indeed lifted to a vector field on the BCS (Figure 2).

6 Interactive Design of Branched Covering Spaces

While the topology of the BCS is unique given the input mesh and the N-RoSy field defined on it, its geometric realizations in $\mathbb{R}^3$ form a large space. Unlike existing method [16] which provides an automatic method to generate the BCS, we provide an interactive design system that allows the user to create different geometric realizations of the BCSs.

At the very beginning, the user can modify the input N-RoSy field including the number and location of the singularities in the field using

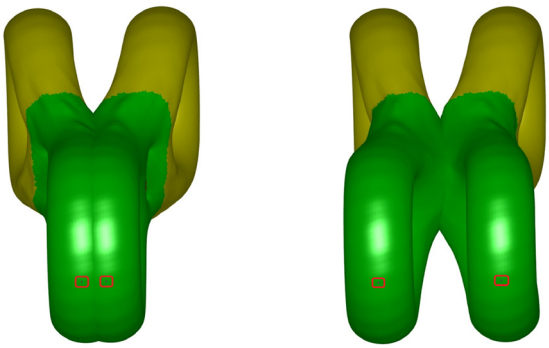

Fig. 7. Our system allows the BCS to be deformed: (left) regions to deformed (green) around the deformation handles (the purple triangles enclosed by the red boxes which are located at the end of the green parts of the surface), and (right) the deformed BCS.

the technique of Palacios and Zhang [18]. Next, the user can edit the essential cut graph by replacing an edge in the graph with the other two edges in the same triangle or two edges in the same triangle with the third edge of a triangle. The gap for the newly added edges is automatically computed. We support these operations even when some of the involved vertices are singularities in the field, as long as no singularity is removed from the essential cut graph.

Before layer connection, the user can specify the location, size, and orientation of each layer. With mesh deformation, the user can additionally specify the location and orientation of edges corresponding to the essential cut graph as well as part of the layers that will not deform. With connecting tubes, the user can specify the location, size, and orientation of the docking stations.

After the BCS has been generated, the user can further improve the geometric realization of the BCS.

When connecting tubes are used, the user has the option to deform the path that each tube takes to improve the smoothness of the path, to avoid collisions (intersection) with unrelated tubes, layers, and docking stations, as well as to reduce visual cluttering caused by the tubes. The construction of these curves is done by taking the strokes drawn by the user on screen as inputs which are filtered and used as the control points to draw Bezier curves [23].

In case connecting tubes are not used, i.e., layer connection is achieved through Laplacian smoothing, geometric realization of the BCS can be improved through mesh deformation.

During the deformation process, we strive to maintain the shape of each layer in order to maintain their recognizability. To achieve this, we reuse the framework of [25], which transfers the deformation from one surface (source) to another (target). Given an initial pose of the source mesh $P_{s,1}$ and the pose after deformation $P_{s,2}$, the framework seeks to extract the deformation and apply it to the initial pose of the target mesh $P_{t,1}$ to produce $P_{t,2}$. The vertex locations for $P_{t,2}$ are computed by minimizing an energy function which is the total squared difference between the deformation of every triangle in $P_{s,1}$ and its corresponding triangle in $P_{t,1}$.

In our case, we do not have two meshes. Therefore, to apply the framework, we let the user select a subset of triangles in the mesh which can serve as the deformation handle. The user can translate and rotate these triangles. The affine transformation from this subset of triangles and the rest of the triangles in the mesh will be maintained as much as possible during the deformation of the rest of the triangles. This leads to a similar framework.

When deforming each layer, we automatically compute a seed in the base mesh and use its pre-images in the BCS as the seed triangle for each layer. Since the self-intersections occur most around singularities, we wish to select a seed that is far away from the singularities in the base mesh. Figure 7 shows an example BCS before and after deformation. The seed triangles are enclosed in red boxes.

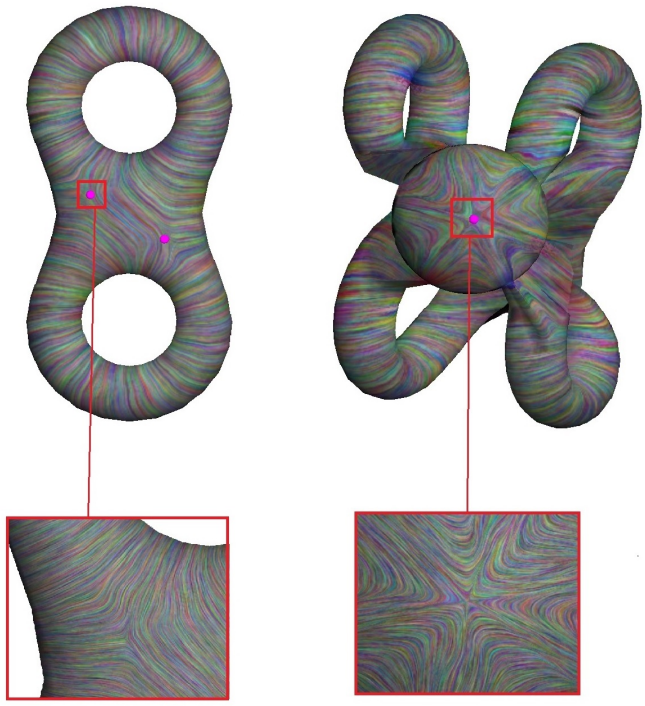

Fig. 8. An $-\frac{1}{2}$ indexed singularity in the 2-RoSy field on the double torus (left) corresponds to an -2 indexed singularity in the vector field in the BCS (right).

7 Animation

While the aforementioned visualization techniques can clearly show that the BCS is mostly N copies of the input surface except at the ramification points and that the N-RoSy field lifts to a vector field on the BCS, there are other important properties of the BCS that cannot yet be clearly demonstrated.

For example, the BCS is a manifold surface if the input surface is a manifold. That is, the ramification point is a manifold point. To address this, our system provides an operation that can take the neighborhood of a ramification point in the BCS and unfold it onto a hemisphere. The operation, which we term *ramification point inflation*, is achieved through a process similar to the aforementioned mesh deformation process. In this case, the user can specify a neighborhood R of the ramification point and a hemisphere H in the 3D space, and a correspondence between R and H is automatically established. This leads to a shape interpolation between R and H. A large neighborhood H' of H is then deformed to maintain the connectivity between the deformed H and the rest of the BCS (see Figure 17 in Section A in the supplementary material). This operation can show that the ramification point is indeed a manifold point in the BCS.

The same operation, when rendered using the Line-Interval-Convolutions (LIC) [7], can be used to show the connection between the index of a singularity and the index of its corresponding ramification point. In addition, we use LIC to show the relationship between a singularity's index I_p and the index of its corresponding ramification point I'_p, i.e., $I'_p = NI_p - (N-1)$ as illustrated in Figure 8.

Our system provides an initial location, orientation, and scale of the fully unfolded neighborhood. However, one can use our graphical interface to modify these in order to achieve an animation of optimal visualization. Please see the accompanying video for an example of this editing.

The Riemann-Hurwitz formula states that there are usually more handles in the BCS than $N \times g$, where g is the number of handles in the input surface. When using mesh deformation based layer connection, it is often difficult to see the additional handles as they are usually

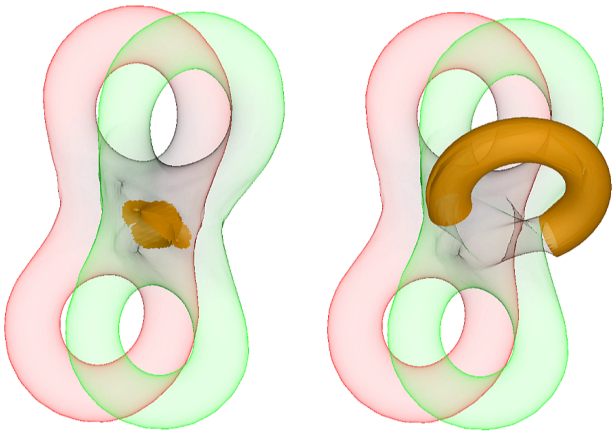

Fig. 9. Two frames in the animation of inflating a handle that is otherwise difficult to see due to the self-intersections in the BCS.

occluded due to self-intersections. To address this, we provide another operation, termed *handle inflation*, which can take the part of a handle and morph it to a cylinder, thus exposing the handle. To do so, we compute a homology generator [9, 10] per handle in the BCS using the technique from [30]. The homology generators are visualized on the BCS. However, some handles are difficult to see due to the self-intersections in the BCS. In this case, we allow the user to deform the BCS in order to make the handle visible using the aforementioned deformation framework. We then grow a topological cylinder from the homology generator by performing region growing from the a region of zero triangles and two boundaries (the homology generator is treated as two co-locating boundaries). The region is grown by adding one triangle at a time until a user-specified distance between the two boundaries is reached. The topological cylinder is then mapped to a canonical half torus (also a topological cylinder) of a user-given outer and inner radii. This is achieved by finding a shortest path between the two boundaries on the topological cylinder on the BCS and unfolding it onto a rectangle such that the top and bottom sides correspond to the shortest cut, and the left and right sides correspond to the two boundaries of the cylinder. We ensure that each vertex on the shortest path is mapped to two points on the top and bottom sides with the same X-coordinates. The interior vertices are then solved using the parameterization technique [24]. Since a mapping from a rectangle to a half torus is known, we have now mapped the topological cylinder to the half torus. Again, the user can choose to only see the topological cylinder deforming into a half torus, i.e., disconnected from the rest of the BCS, or deforming a transition region between the topological cylinder and the fixed region on the BCS. The user can also change the size of the topological cylinder, the transition region, as well as the inner and outer radii, orientation, and location of the half torus. Solid and translucent rendering as well as static and dynamic LIC are used to see the deformation. Figure 9 shows intermediate results of the animation.

8 The Riemann-Hurwitz Formula and Related Properties

The Riemann-Hurwitz formula is one of the most important properties of the BCS as it relates the topology of the base mesh and that of the BCS. In addition, the Riemann-Hurwitz formula leads to a number of important additional properties of the BCS, such as that the BCS is independent of how the mesh surface is cut open (i.e., the essential cut graph) and that any closed, orientable surface with at least one handle is a two-fold BCS for the sphere.

When being translated in terms of number of handles, the Riemann-Hurwitz formula states that the number of handles in the BCS is N times the number of handles in the base mesh with $(\frac{S}{2}-1)(N-1)$ handles. Here N is the degree of the RoSy field and S is the number of singularities in the field. Here we assume that the only singularities in the field are first-order, which is a common assumption in geometry processing.

While handle inflation can help verify the existence of additional handles in the BCS, it is difficult to build a geometric intuition on why the additional handles are needed for a valid BCS. We demonstrate this with a sequence of N-RoSy fields defined on the same surface so that each N-RoSy field in the sequence has two additional singularities than the preceding field. Figure 10 shows this where $N = 2$ and the base mesh is a sphere. The first 2-RoSy field has four first-order singularities, which is the minimum due to the Poincaré-Hopf theorem (the Euler characteristic being two). Moving down, each field has two more singularities (one positive-indexed and one negative-indexed so that the total index remains the same). The right column shows the corresponding BCSs. In the first row, the four singularities lead to two connected components in the essential cutgraph. By connecting the two layers with tubes, it is clear that each connected component leads to a bridge between the corresponding singularity pairs on the two layers. Therefore, there are two bridges between the layers, forming a new handle. Note that this is consistent with the above translated version of the Riemann-Hurwitz formula. Every time a new singularity pair is added to the field, a new connected component is introduced into the essential cut graph, thus a new bridge connecting the layers. This bridge, in the presence of existing bridges, leads to a new handle.

Figure 11 shows the same geometric intuition behind the Riemann-Hurwitz formula for higher-order RoSy fields. In this example, we show the BCSs corresponding to two 2-RoSy fields (top), two 3-RoSy fields (middle), and two 4-RoSy fields (bottom) on the torus. The fields for the BCSs in the left have only two singularities while the ones for the right have two additional singularities. Consequently, the BCSs for the right have one more docking station connecting to the N layers with bridges than the ones in the left. These additional N outgoing bridges leads to $N-1$ new handles, which is the consistent with the

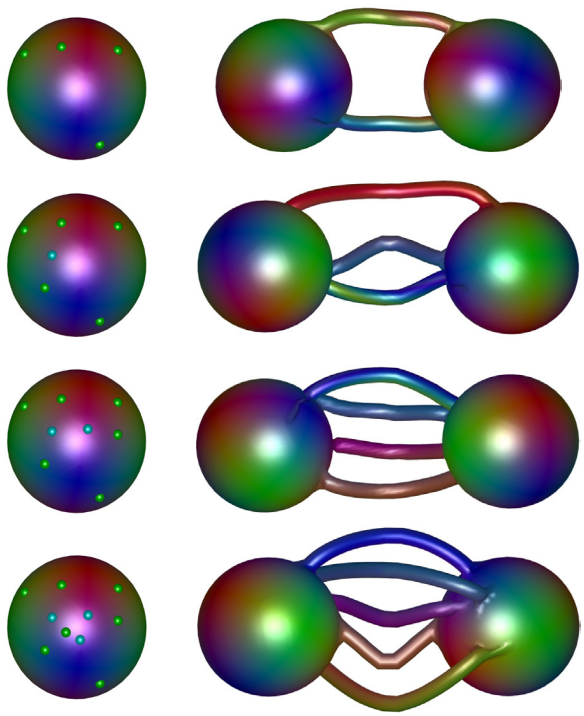

Fig. 10. Four 2-RoSy fields defined on the sphere (left) and their corresponding BCSs (right). From top to bottom, the 2-RoSy fields have four, six, eight, and ten singularities, respectively. Their corresponding BCSs have two, three, four, and five bridges, leading to respectively one, two, three, and four handles. In addition, this figure shows that any closed, orientable surface with at least one handle is a two-fold cover of the sphere.

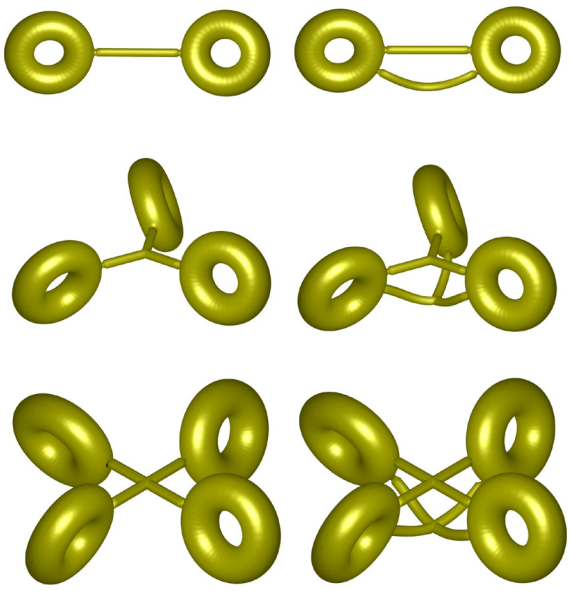

Fig. 11. BCSs of the torus with one (left) and two (right) pairs of singularities on the input N-RoSy field. From top to bottom, the BCS results are from the fields of 2-RoSy, 3-RoSy and 4-RoSy respectively on the input surface. These figures demonstrate the Riemann-Hurwitz formula which says that with addition of a singularity pair on the N-RoSy field on the input surface, N-1 handles are added in the resulting BCS.

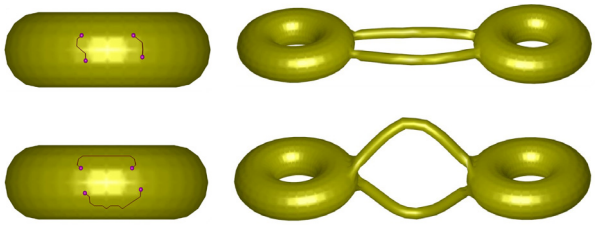

Fig. 12. A 2-RoSy field on the torus with four singularities leads to two different essential cut graphs (left). Their corresponding BCSs are topologically equivalent (three handles).

Riemann-Hurwitz formula.

To verify that the BCS is independent of essential cut graph, our system allows the user to edit the essential cut graph that can change which singularities are in the same connected component (Figure 12). Since the total number of singularities does not change when the essential cut graph is altered, the same number of bridges are formed, albeit from different parts of the layers. This leads to the same BCSs in terms of topological equivalence.

Finally, reading Figure 10 from right to left shows that any closed, orientable manifold with at least one handle is a two-fold cover of the sphere.

9 Performance

Our tool has been tested on a system with Intel(R) Xeon(R) CPU with 3.40 Ghz speed with a RAM of 64 GB and an NVidia Quadro K420 graphics card. Our LIC and dual depth peeling methods are implemented using the Shaders while the mesh deformation is achieved on the CPU. The time to perform one step in mesh deformation depends on the size of the input mesh and ranges from a few frames per second (e.g., sphere, torus, double torus, which have up to $10,000$ triangles) to two seconds (e.g., buddha and higher resolution mesh with $10,000-40,000$ triangles). Our BCS visualization and design system is robust, as demonstrated by Figure 18 (Section A in the supplementary material). Note these results are not for educational purposes. The 2-RoSy field on the torus (top) has 704 singularities, leading to a BCS with 353 handles. The budda model is complex both geometrically and topologically (six handles), and the 20-RoSy field leads to a BCS with 9202 handles. Figure 1 (right) shows a 100-fold cover of the double torus, which has 21188 handles. Our system is able to construct and visualize these BCS meshes.

10 Evaluation

Branched covering spaces is a concept in Topology that is often taught at the graduate level in Mathematics. Moreover, it is usually taught with simple illustrations.

To evaluate the effectiveness of our interactive and design system, we have conducted a user study with 75 participants, including three professors in mathematics who teach topology regularly, 7 math students, 22 computer science students, and 43 high school students who either have taken or are taking AP Calculus. Among the participants, 15 (including the three mathematics faculty members) chose to use our system instead of seeing a presentation that we prepared with our tool. The presentation is 35 minutes long, while the participants using our system did so in 45 minutes. At the end of the study, each participant filled a survey which includes three parts: (1) background information such as level (high school, university, faculty) and major, (2) the participant's evaluation of the effectiveness of our visualization system in terms of a seven branched covering space properties, and (3) suggestions and comments. The seven properties are:

- Q1: N-fold branched cover consists of N layers.
- Q2: BCS lifts an N-RoSy field to a vector field.
- Q3: BCS is a manifold, even around the singularities.
- Q4: The Riemann-Hurwitz formula.
- Q5: BCS is independent of cut graph.
- Q6: Any closed, orientable surface with at least one handle is a two-fold branched cover of a sphere.
- Q7: Index of the ramification point in the BCS relates to the index of the corresponding singularity in the N-RoSy field.

For each property, the participant was asked whether he/she thinks our visualization technique is effective in demonstrating the property. The rating system is: agree (3), neutral (2), and disagree (1).

Figures 13, 14, and 15 show the user study results by the high school students, computer science university students, and math university students, respectively. Figure 16 provides the result of the evaluation by the users who used our system, including the three math faculty members regularly teaching topology.

As shown in the results, the overall evaluation is positive and encouraging. We observe that the participants who chose to use our system in general had a higher rating of our visualization system than the participants who attended our presentation only. On the other hand, our visualization system is rated similarly among high school students, computer science university students, and math university students. Both the researchers on this project (research areas covering tensor field topology and quad-remeshing) and the math faculty participants agreed that three of the most essential tasks for BCS visualization are: (1) showing that the BCS is a manifold surface, (2) highlighting that the BCS is N copies of the original surface, and (3) demonstrating the relationship between the topology of the BCS and that of the original surface, i.e., the Riemann-Hurwitz formula. Regarding the effectiveness of our visualization techniques in terms of these tasks, most participants agreed that the tube-based visualization is more useful in showing the topology of the BCSs (being a manifold and satisfying the Riemann-Hurwitz formula). The non-tube based visualization is better in demonstrating that the BCS is exactly N copies of the base surface stitched together since no additional surfaces (tubes) are needed. All three mathematics faculty members stated in their comments that our design system can facilitate teaching topology and inspire new ideas in their research.

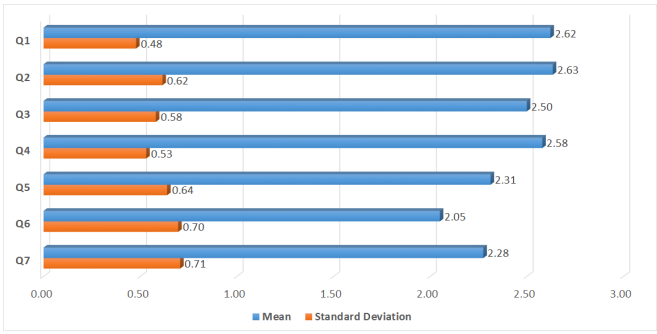

Fig. 13. Evaluation of effectiveness of our system on a scale of 1 to 3 (1 = not effective, 2 = neutral and 3 = effective) to understand key properties of BCS by high school students.

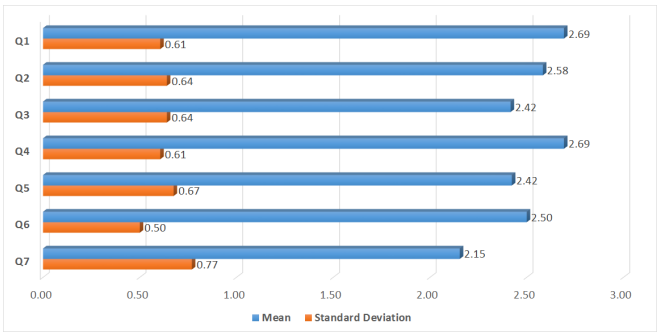

Fig. 14. Evaluation of effectiveness our system on a scale of 1 to 3 (1 = not effective, 2 = neutral and 3 = effective) to understand key properties of BCS by undergraduate and graduate computer science students.

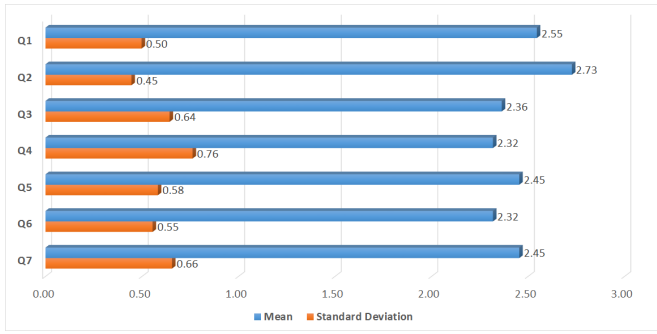

Fig. 15. Evaluation of effectiveness our system on a scale of 1 to 3 (1 = not effective, 2 = neutral and 3 = effective) to understand key properties of BCS by undergraduate and graduate mathematics students.

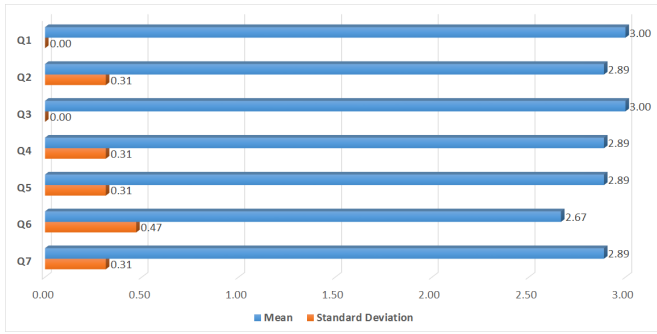

Fig. 16. Evaluation of effectiveness our system on a scale of 1 to 3 (1 = not effective, 2 = neutral and 3 = effective) to understand key properties of BCS by the users who interacted with our system.

11 Conclusion

In this paper, we describe an interactive design and visualization system for the BCS of a manifold surface. With various visualization, mesh deformation, and visualization techniques, our system allows a user to build intuitions on important properties of the BCS, such as its construction, the connection between the indices of a singularity and its corresponding ramification point, and the Riemann-Hurwitz formula. As part of our system, we introduce the notion of essential cut graph, which not only leads to improved efficiency in constructing the BCS, but also enables the use of docking stations that help reduce self-intersections in the BCSs.

Our system is not without limitations. When N is large, there are usually a large number of singularities in the N-RoSy field, leading to many connecting tubes which require a significant amount of user effort to generate a visually pleasing BCS. This can also happen when the input surface or the field has relatively complex topology. Visual cluttering often occurs as a result. To address these issues, we plan to explore automatic placement of the docking stations, layers and connecting tubes to improve the aesthetic of the BCS before any editing. In addition, we will investigate operations to bundle the tubes or handles in BCSs similar to *edge bundling* [31]. We also plan to add highlighting and filtering capabilities to our system.

Enhancing the visualization of the topological and geometric structures of docking stations provides a future research avenue. For example, instead of representing docking stations always as a sphere with holes, we will explore situations under which docking stations are better represented as a torus to which tubes are attached.

When connecting tubes are not used, a large amount of self-intersections usually occur and mesh deformation is needed to reduce the amount of self-overlaps. We wish to investigate automatic deformation strategies to optimize the shape of BCSs with minimal distortion and self-intersections. In addition, we plan to explore the use of regular texture patterns on BCSs to more clearly show the connection among different layers. Generating animations for ramification point inflation and handle inflation without moving the self-intersections is also a promising future research direction.

In this paper we have focused on fields with only first-order singularities. In the future, we wish to extend our system to handle higher-order singularities in the input fields. Constructing and visualizing BCSs for non-orientable surfaces such as the real projective space and the Klein bottle is of great interest to us. While visualizing BCSs for two-dimensional surfaces can be useful for researchers in quad remeshing, we plan to extend this work to visualize BCSs for volumetric frame fields. Finally, we will explore the use of our system for educational purposes.

Acknowledgments

We wish to thank our anonymous reviewers for their valuable, constructive suggestions. Professors Ren Guo, Dennis Garity, and Juha Pohjanpelto of the Department of Mathematics at Oregon State University provided valuable feedbacks and suggestions, which we greatly appreciate. We also thank Jonathan Palacios for the software on the design of rotational symmetry fields on surfaces. Tom Roy and Amy Roy have helped proofreading the paper, and we appreciate their support. Yue Zhang is partially supported by NSF award 1566236. Eugene Zhang is partially supported by NSF award 1619383.

References

[1] P. Alliez, D. Cohen-Steiner, O. Devillers, B. Lévy, and M. Desbrun. Anisotropic polygonal remeshing. In *ACM SIGGRAPH 2003 Papers*, SIGGRAPH '03, pp. 485–493. ACM, New York, NY, USA, 2003. doi: 10.1145/1201775.882296

[2] M. Armstrong. *Basic topology*. McGraw-Hill Book Co., 1979.
[3] M. Bátorová, M. Valíková, and P. Chalmovianský. Desingularization of ade singularities via deformation. In *Proceedings of the 29th Spring Conference on Computer Graphics*, SCCG '13, pp. 035:35–035:42. ACM, New York, NY, USA, 2013. doi: 10.1145/2508244.2508249
[4] L. Bavoil and K. Myers. Order independent transparency with dual depth peeling. Technical report, NVIDIA Developer SDK 10, Feb. 2008.
[5] D. Bommes, B. Lévy, N. Pietroni, E. Puppo, C. Silva, M. Tarini, and D. Zorin. Quad-mesh generation and processing: A survey. *Comput. Graph. Forum*, 32(6):51–76, Sept. 2013. doi: 10.1111/cgf.12014
[6] D. Bommes, H. Zimmer, and L. Kobbelt. Mixed-integer quadrangulation. In *ACM SIGGRAPH 2009 Papers*, SIGGRAPH '09, pp. 77:1–77:10. ACM, New York, NY, USA, 2009. doi: 10.1145/1576246.1531383
[7] B. Cabral and L. C. Leedom. Imaging vector fields using line integral convolution. In *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '93, pp. 263–270. ACM, New York, NY, USA, 1993. doi: 10.1145/166117.166151
[8] M. Campen, D. Bommes, and L. Kobbelt. Dual loops meshing: Quality quad layouts on manifolds. *ACM Trans. Graph.*, 31(4):110:1–110:11, July 2012. doi: 10.1145/2185520.2185606
[9] T. K. Dey, F. Fan, and Y. Wang. An efficient computation of handle and tunnel loops via reeb graphs. *ACM Transactions on Graphics (TOG)*, 32(4):32, 2013.
[10] T. K. Dey, K. Li, J. Sun, and D. Cohen-Steiner. Computing geometry-aware handle and tunnel loops in 3d models. In *ACM Transactions on Graphics (TOG)*, vol. 27, p. 45. ACM, 2008.
[11] R. Hartshorne. *Algebraic geometry*. Graduate texts in mathematics. Springer, New York, 1977.
[12] B. Jobard and W. Lefer. Creating evenly-spaced streamlines of arbitrary density. pp. 43–56, 1997.
[13] F. Kälberer, M. Nieser, and K. Polthier. Quadcover - surface parameterization using branched coverings. *Comput. Graph. Forum*, 26(3):375–384, 2007. doi: 10.1111/j.1467-8659.2007.01060.x
[14] D. M. Morera, L. Velho, and P. C. P. Carvalho. Computing geodesics on triangular meshes. *Computers & Graphics*, 29(5):667–675, 2005. doi: 10.1016/j.cag.2005.08.003
[15] M. Nieser, J. Palacios, K. Polthier, and E. Zhang. Hexagonal global parameterization of arbitrary surfaces. *IEEE Transactions on Visualization and Computer Graphics*, 18(6):865–878, June 2012. doi: 10.1109/TVCG.2011.118
[16] M. Nieser, K. Poelke, and K. Polthier. Automatic generation of riemann surface meshes. In B. Mourrain, S. Schaefer, and G. Xu, eds., *Advances in Geometric Modeling and Processing*, vol. 6130 of *Lecture Notes in Computer Science*, pp. 161–178. Springer Berlin / Heidelberg, 2010.
[17] M. Nieser, U. Reitebuch, and K. Polthier. Cubecover- parameterization of 3d volumes. *Comput. Graph. Forum*, 30(5):1397–1406, 2011. doi: 10.1111/j.1467-8659.2011.02014.x
[18] J. Palacios and E. Zhang. Rotational symmetry field design on surfaces. In *ACM SIGGRAPH 2007 Papers*, SIGGRAPH '07. ACM, New York, NY, USA, 2007. doi: 10.1145/1275808.1276446
[19] J. Palacios and E. Zhang. Interactive visualization of rotational symmetry fields on surfaces. *IEEE Transactions on Visualization and Computer Graphics*, 17(7):947–955, July 2011. doi: 10.1109/TVCG.2010.121
[20] K. Polthier and E. Preuß. Identifying vector field singularities using a discrete hodge decomposition. pp. 112–134. Springer Verlag, 2002.
[21] N. Ray, W. C. Li, B. Lévy, A. Sheffer, and P. Alliez. Periodic global parameterization. *ACM Trans. Graph.*, 25(4):1460–1485, Oct. 2006. doi: 10.1145/1183287.1183297
[22] N. Ray, B. Vallet, W. C. Li, and B. Lévy. N-symmetry direction field design. *ACM Trans. Graph.*, 27(2):10:1–10:13, May 2008. doi: 10.1145/1356682.1356683
[23] F. F. Samavati. Local filters of b-spline wavelets. In *Proceedings of International Workshop on Biometric Technologies 2004*, pp. 105–110, 2004.
[24] P. V. Sander, J. Snyder, S. J. Gortler, and H. Hoppe. Texture mapping progressive meshes. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '01, pp. 409–416. ACM, New York, NY, USA, 2001. doi: 10.1145/383259.383307
[25] R. W. Sumner and J. Popović. Deformation transfer for triangle meshes. In *ACM SIGGRAPH 2004 Papers*, SIGGRAPH '04, pp. 399–405. ACM, New York, NY, USA, 2004. doi: 10.1145/1186562.1015736
[26] G. Taubin. A signal processing approach to fair surface design. In *Proceedings of the 22Nd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '95, pp. 351–358. ACM, New York, NY, USA, 1995. doi: 10.1145/218380.218473
[27] Y. Tong, S. Lombeyda, A. N. Hirani, and M. Desbrun. Discrete multiscale vector field decomposition. In *ACM SIGGRAPH 2003 Papers*, SIGGRAPH '03, pp. 445–452. ACM, New York, NY, USA, 2003. doi: 10.1145/1201775.882290
[28] X. Tricoche. *Vector and Tensor Topology Simplification, Tracking, and Visualization*. PhD thesis, University of Kaiserslautern, 2002.
[29] M. Valíková and P. Chalmovianský. Visualisation of complex functions on riemann sphere. *The Visual Computer*, 31(2):141–154, 2015. doi: 10.1007/s00371-014-0928-3
[30] E. Zhang, K. Mischaikow, and G. Turk. Feature-based surface parameterization and texture mapping. *ACM Trans. Graph.*, 24(1):1–27, Jan. 2005. doi: 10.1145/1037957.1037958
[31] H. Zhou, P. Xu, X. Yuan, and H. Qu. Edge bundling in information visualization. *Tsinghua Science and Technology*, 18(2):145–156, 2013.