3D PRINTING THE TREFOIL KNOT AND ITS PAGES

by

FREDERICK HOHMAN

A Thesis Submitted to the Honors Program of the University of Georgia
in Fulfillment of the Requirements for a 4990H or 5900H course

3D Printing the Trefoil Knot and its Pages

Frederick Hohman, *fred@fredhohman.com*
Dr. David Gay, *dgay@math.uga.edu*
The University of Georgia Department of Mathematics

Spring 2015

Contents

List of Figures

1 Abstract

Observing the physical geometry of shapes that are particularly tough to visualize in topology has been made easier by visualization techniques such as 3D printing. The work presented here describes the creation of a 3D printed puzzle of the open-book decomposition of the complement of the trefoil knot.

The open-book decomposition of the complement of the trefoil knot describes an object where the trefoil knot is equivalent to the spine of the book. Typical pages of an ordinary book are rectangular and are joined from one edge to the spine; when the equivalent pages of a book are added to the trefoil knot they become surfaces winding inside and outside of the knot.

After working through the mathematical theory behind the creation of the puzzle, utilizing various concepts such as stereographic projection, we created and printed primitive models, such as a thickened trefoil knot alone, a trefoil knot plus one thickened page, and a trefoil knot plus three thickened pages spaced equiradially.

Using Mathematica, a symbolic mathematics computer programming environment, high resolution 3D models of the knot and pages were created. With the aid of other modeling software, namely Blender, the inclusion of twelve pages was made possible by digitally cutting each page in half. From here, holes were removed from the 3D mesh of each half page to provide space for small magnets. Once printed, magnets are glued into the holes to allow each page and piece of the knot to join together to construct the desired puzzle. The puzzle can be taken apart and rebuilt in various ways that demonstrate properties of the trefoil knot and the accompanying surfaces.

2 Motivation and Background

The goal of this project is to create a 3D printed puzzle of the complement of the trefoil knot and its pages in order to better visualize the shape as a whole. The notion of a page will be explained in greater detail in the following sections.

To begin, consider the following function:

$$T : \mathbb{C}^2 \to \mathbb{C}$$

$$(u, v) \mapsto (u + iv)^2 - (u - iv)^3,$$

where $u = u_x + iu_y, v = v_x + iv_y \in \mathbb{C}, u_x, u_y, v_x, v_y \in \mathbb{R}$, and $i = \sqrt{-1}$.

As we will see throughout the following computations, this particular function's zero-set, i.e., the set of points $\{(u, v)|(u + iv)^2 = (u - iv)^3\}$, intersected with $\mathbb{S}^3$, i.e. with the extra constraint that

$$|u|^2 + |v|^2 = 1,$$

generates a trefoil knot through infinity. By saying "through infinity," we mean "through $(0, i)$," i.e. $(0, i)$ is in the zero-set. So our function T evaluated at the point $(0, i)$ is the following:

$$T(0, i) = (0 + i^2)^2 - (0 - i^2)^3 = 1 - 1 = 0.$$

In other words, closed curves in $\mathbb{S}^3$ that pass through the point $(0, i)$ when subjected to stereographic projection (method to be discussed in the following section) become knots though infinity.

By considering inverse images of certain subsets of $\mathbb{C}$ given our constraints, we will generate a thickened trefoil knot; however, this inverse image is a subset of $\mathbb{C}^2$ as described from our function above, so in order to 3D print we need a method to realize our object in $\mathbb{R}^3$.

2.1 Stereographic Projection

The method we will use is a generalized stereographic projection. As a preliminary example, consider the following qualitative description of standard stereographic projection. Suppose a sphere in $\mathbb{R}^3$ is sitting on a plane (think of a ball sitting on a table). From the top of the sphere (think North Pole), draw a line segment downward through the sphere and end at the plane. Notice that this line segment intersects both the sphere and table once. We can do this same line drawing method over and over to obtain a nearly one-to-one correlation from the sphere to the plane, i.e., every point except the exact top $(0, 0, 1)$ on the sphere can be mapped to the plane. This mapping is called stereographic projection.

Dr. Henry Segermen, **henryseg** on Thingiverse, has created a fantastic 3D printed model illustrating this method. By using light rays as the straight lines, one can hold a point light source at $(0, 0, 1)$ and watch as the pattern on model is projected into the plane with exact right angles.

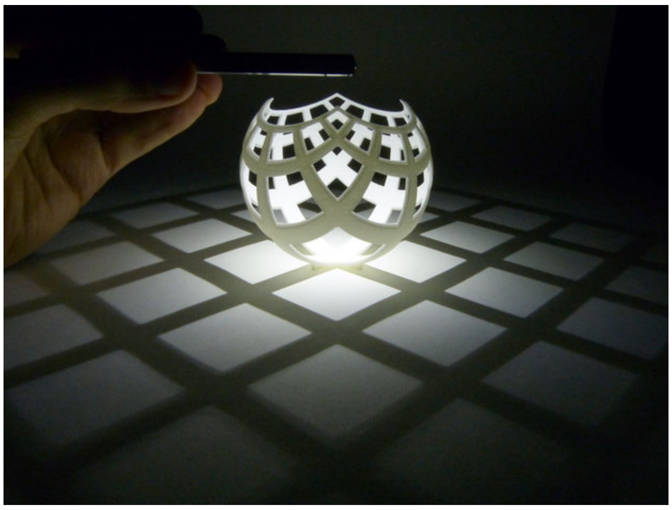

Figure 1: Henry Segermen's stereographic projection 3D print.

So stereographic projection provides a map from the surface of a sphere to the plane, both of which are two dimensional, but since the typical sphere is embedded $\mathbb{R}^3$, this has the effect of lowering the dimension of our considered subset by one. We can now consider a generalized stereographic projection and define it as the following function:

$$\pi : \mathbb{S}^3 - \{(0, 0, 0, 1)\} \to \mathbb{R}^3,$$

where

$$\pi(u_x, u_y, v_x, v_y) = \left(\frac{u_x}{1 - v_y}, \frac{u_y}{1 - v_y}, \frac{v_x}{1 - v_y} \right).$$

Most simply, this generalized stereographic projection function now takes in four numbers and produces three, once again lowering our dimension of consideration by one.

2.2 Composition of Functions

Recall that we wish to visualize our trefoil knot in $\mathbb{R}^3$. So consider the following function:

$$\pi^{-1} : \mathbb{R}^3 \to \mathbb{S}^3,$$

where

$$\pi^{-1}(x, y, z) =$$

$$\left(\frac{2x}{x^2+y^2+z^2+1}, \frac{2y}{x^2+y^2+z^2+1}, \frac{2z}{x^2+y^2+z^2+1}, \frac{x^2+y^2+z^2-1}{x^2+y^2+z^2+1}\right).$$

We can now compute the following evaluation of this new function π^{-1}:

$$\pi^{-1}(\pi(u_x, u_y, v_x, v_y)) = (u_x, u_y, v_x, v_y).$$

So this composition of functions evaluated at an arbitrary point has the effect of applying the identity function. So our stereographic projection function π has an inverse, therefore it is invertible, with inverse π^{-1}.

Since $\mathbb{S}^3 \subset \mathbb{C}^2$, which we can identify with $\mathbb{R}^4$, we define a new function that is the composition of π^{-1} with T. Call this function f, so

$$f = T \circ \pi^{-1} : \mathbb{R}^3 \to \mathbb{C}.$$

We can now take the inverse images of certain subjects of f so that, given a set $A \in \mathbb{C}$ we can visualize it in $\mathbb{R}^3$ by

$$f^{-1}(A) \subset \mathbb{R}^3.$$

We now have a single function that performs the transformation we want.

3 3D Model Generation

In order to 3D print our knots, we need to create digital 3D models. The most relevant file type we wish to use called an .stl file, which stands for **ST**ereo**L**ithography. These files describe only the surface geometry of a three-dimensional object regardless of common attributes typically found in other three-dimensional file types. There are a variety of software suites that can generate .stl files, but the work here will make use of three in particular: Mathematica, Blender, and MakerBot Desktop.

3.1 Mathematica Primer

RegionPlot3D, a built-in Mathematica function, plots regions in $\mathbb{R}^3$ using specified inequalities. In order to use RegionPlot3D, the user must specify the domain of values to plot over in the x, y, and z directions, as well as a mathematical inequality to satisfy. We are using this function instead of a typical 3D plot or parametric plot because RegionPlot3D carries out the exact process that we wish to perform, that is, it plots regions in $\mathbb{R}^3$ by considering regions in the plane.

As an example we can generate a sphere of radius 1 by the following code:

```
RegionPlot3D[
 (*Inequality*)
 x^2 + y^2 + z^2 < 1,

 (*Domain*)
 {x, -1, 1},
 {y, -1, 1},
 {z, -1, 1}
]
```

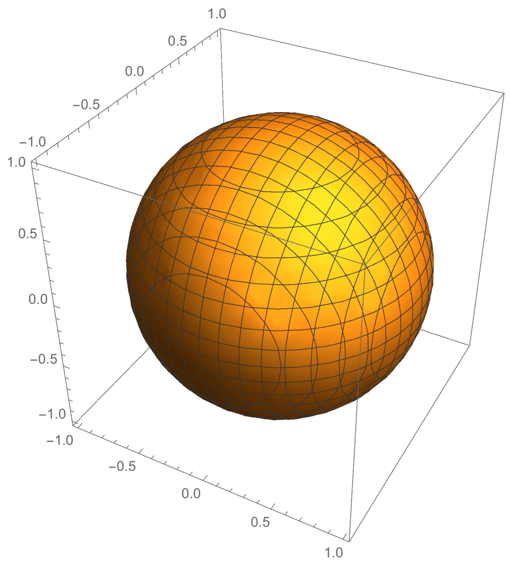

Figure 2: Mathematica code for a sphere.

Mathematica will plot all points (x, y, z) such that each point falls within the considered domain and obeys the given inequalities. As an extension of this idea, we can require our points to satisfy multiple inequalities using boolean operators such as **AND** and **OR**. As an example, consider the same sphere defined above, but restrict the region so that only points above the plane $z = 0$ are plotted.

```
RegionPlot3D[
 (*Inequality*)
 x^2 + y^2 + z^2 < 1

  (*Boundaries*)
  && z > 0,

 (*Domain*)
 {x, -1, 1},
 {y, -1, 1},
 {z, -1, 1}
]
```

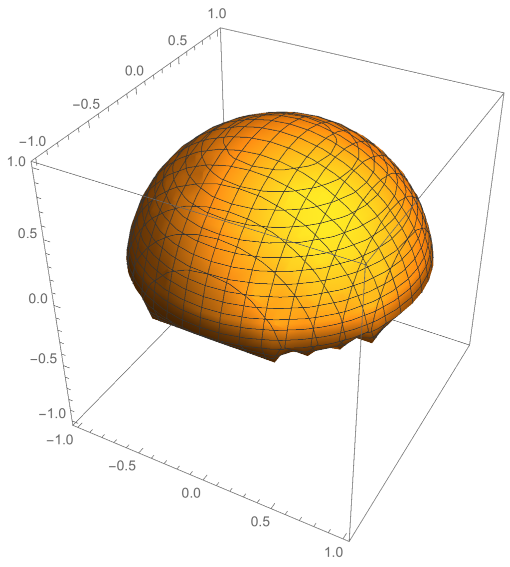

Figure 3: Mathematica code for a hemisphere.

Notice we could simply restrict the z range so that z goes from $(0, 1)$, but it is constructive to be able to manipulate a model without changing the overall plotted region. When models become more complicated this method allows us to control individual variables without altering other pieces.

3.2 Resolution Comparison

The primary characteristic to consider when creating .stl files is the 3D resolution. Mathematica achieves this through an option called **PlotPoints**. A higher PlotPoints value tells Mathematica to use more points to represent the plotted region, and ultimately more triangles to model the 3D mesh. However, as we increase PlotPoints, we also increase computation time. For example, say we used a PlotPoints of 50 and the generated mesh was unresolved. We can double our resolution and set PlotPoints to 100, but recall we are in $\mathbb{R}^3$, so by doubling the number of points in all directions x, y, and z we increase our computation by $2 \times 2 \times 2 = 8$. In other words, doubling a model's resolution increases computation time by a factor of eight.

In the example above, notice how choppy and non-circular the base of our hemisphere looks. Let us plot that same hemisphere with a specified PlotPoints of 100 and see what the bottom looks like.

```
RegionPlot3D[
  (*Inequality*)
  x^2 + y^2 + z^2 < 1

   (*Boundaries*)
   && z > 0,

  (*Domain*)
  {x, -1, 1},
  {y, -1, 1},
  {z, -1, 1},

  (*Options*)
  PlotPoints → 100
]
```

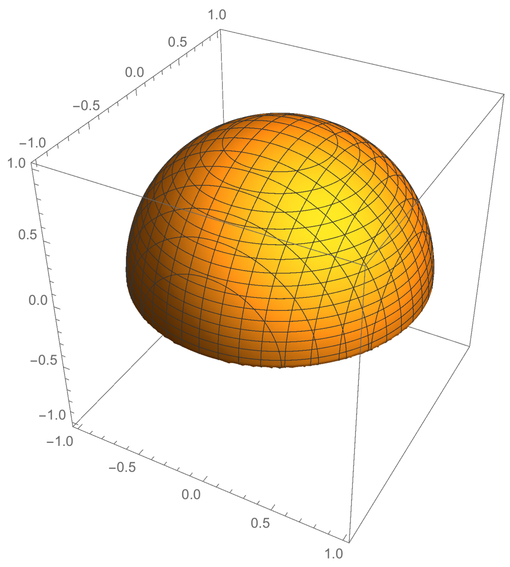

Figure 4: Mathematica code for a high resolution hemisphere.

We now see that the mesh is now much smoother, appearing to look like continuous curve. We can now see the effect PlotPoints and resolution has on 3D meshes.

3.3 Mathematica

If we were to take our subset of $\mathbb{C}$ to be just the point $\{(0, 0, 0, 0)\}$, then we would indeed generate a trefoil knot through infinity in $\mathbb{R}^3$, however this knot would be infinitely thin, and unsuitable for 3D printing. The primary Mathematica function written to visualize the trefoil knot, **inTinftube**, defines the trefoil knot mesh (the composition of our trefoil knot function with inverse stereographic projection explained above). This function takes in points (x, y, z) and "knot thickness" and returns a single number. The function then tests points using inequalities such that if the outputted number is less than zero, Mathematica includes the point in the plot, and if the number is greater than zero, Mathematica does not include the point in the plot. The "knot thickness" adds a tube around the trefoil knot that enables us to 3D print the knot.

We also need to define a boundary condition on our mesh, otherwise our model would stretch to infinity. We can include more inequalities that points must satisfy such that our boundary is a cylinder of radius 3, height 6, and is centered at the origin. With these parameters selected, the results below depict our "standard" thickened trefoil knot. This knot is the inverse image of a small disk centered at the origin in $\mathbb{C}$ to give the knot thickness.

```
RegionPlot3D[
 (*Trefoil Knot*)
 (inTinftube[x, y, z, 1.0] < 0)

  (*Boundaries*)
  && x^2 + y^2 < 9
  && z > -3
  && z < 3,

 (*Domain*)
 {x, -3, 3},
 {y, -3, 3},
 {z, -3, 3},

 (*Options*)
 AxesLabel → Automatic,
 ImageSize → 850,
 PlotPoints → 250,
 Mesh → None
]
```

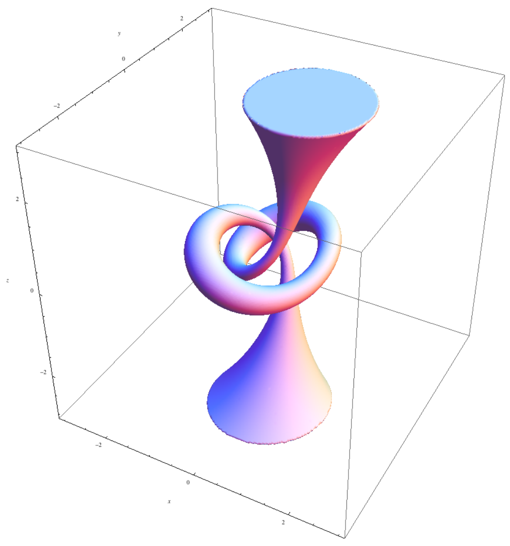

Figure 5: Mathematica code for a thickened trefoil knot.

Thus it has been computationally shown that the zero-set of the function T, plus the extra constraint from the intersection with $\mathbb{S}^3$, is indeed a trefoil knot through infinity.

4 Preliminary Examples

4.1 Open-Book Decomposition

It is now time to add the pages of the trefoil knot to our code so that we can generate new meshes. To get an intuitive notion of what a page is, consider the following example. Suppose we were to take the trefoil generated above and allow it to bend and deform. We could unravel the knot to form a long, skinny cylinder, much like a string. Now imagine that string is the spine of a book. If we were to add one page to this book, the page would attach to the spine along one of the four edges. If we were to then "re-tie" our spine back into the trefoil knot generated above, what would the page look like? This idea is called an **open-book decomposition**, and the following sections will describe a few preliminary models before presenting the final puzzle.

4.2 Trefoil Knot + 1 Page.

Similarly to that of the trefoil knot, we generally think of a page as being an infinitely thin surface, but once again, in order to 3D print something it needs to have thickness. So to add a page to our mesh, we will add extra regions in $\mathbb{C}$ before we apply our transformation function into $\mathbb{R}^3$. To do this, consider another Mathematica function, **onTinfpage**, that, much like inTinftube, takes in points and outputs a number; however, this time the function requires an angle as an extra input instead of knot thickness. This angle will define a ray in the plane starting at the origin that is rotated the specified angle in the mathematically positive direction. As before, we can have Mathematica plot points if the outputted number is less than zero, but remember we want a page of some physical thickness in order to 3D print. To do this we can call onTinfpage again and input an angle that is $\frac{\pi}{6}$ larger than before and have Mathematica plot points that are greater than zero. We have now included another region in the plane, namely a region that is between two rays defined by two angles.

So consider the following example where our two angles are $\frac{\pi}{6}$ and $\frac{\pi}{3}$. The new subset of $\mathbb{C}$ is seen in the image below.

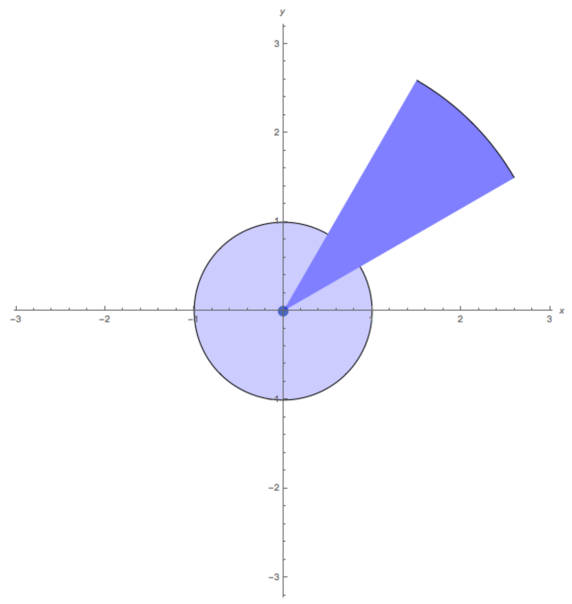

Figure 6: Trefoil knot + 1 page in $\mathbb{C}$.

Once again, the disk centered at the origin is the thickness of the trefoil knot and the region in the first quadrant is our page with $\frac{\pi}{6}$ thickness. That's the new region we want to include when using our function composition. We can now apply our same code to generate the following mesh.

```
RegionPlot3D[
 (*Trefoil Knot + 1 Page*)
 ((onTinfpage[x, y, z, Pi / 6] < 0
      && onTinfpage[x, y, z, Pi / 3] > 0)
    || inTinftube[x, y, z, 1.0] < 0)

  (*Boundaries*)
  && x ^ 2 + y ^ 2 < 9
  && z > - 3
  && z < 3,

 (*Domain*)
 {x, -3, 3},
 {y, -3, 3},
 {z, -3, 3},

 (*Options*)
 AxesLabel → Automatic,
 ImageSize → 850,
 PlotPoints → 250,
 Mesh → None
]
```

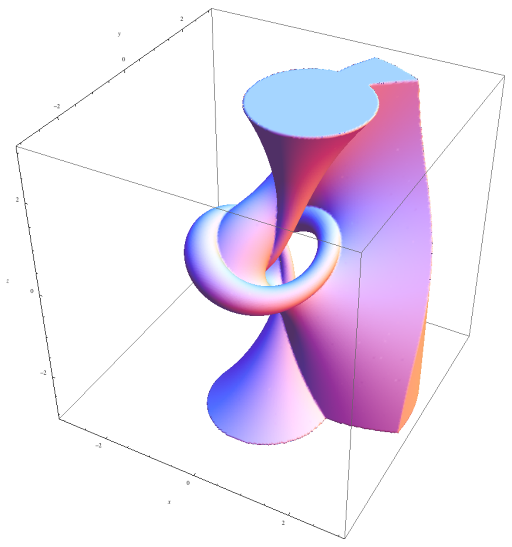

Figure 7: Mathematica code for a trefoil knot + 1 page.

So whereas a page is an infinitely thin surface, we are now looking at thickened page of $\frac{\pi}{6}$ thickness.

4.3 Other Examples

Now that we are able to add one thickened page we can continue to add as many as we would like, and at $\frac{\pi}{6}$ thickness, we should be able to fit twelve extra regions in our trefoil knot ($\frac{\pi}{6} \times 12 = 2\pi$). Some other examples are below, with the region in $\mathbb{C}$ shown on the left, and its image after applying our transformation in $\mathbb{R}^3$ shown on the right.

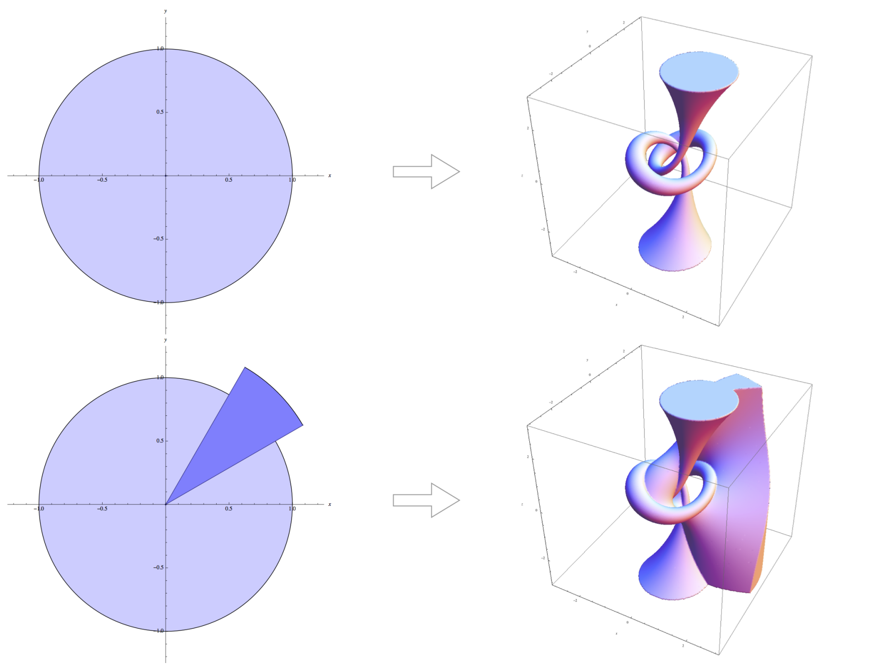

Figure 8: Trefoil knot transformation. Trefoil knot + 1 page transformation.

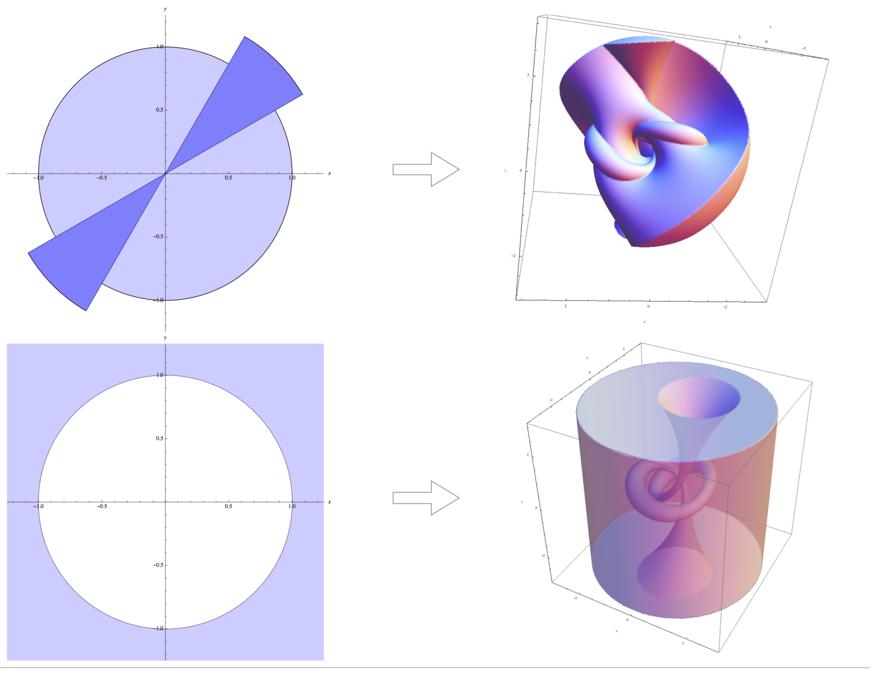

Figure 9: Trefoil knot + 2 pages transformation. Trefoil knot complement transformation.

We can now start to see the consequences of projecting our 4D function into $\mathbb{R}^3$, which is why our knots are visually unlike typical trefoil knots. Note that in the third example above that contains the trefoil knot and two pages, we are bounding the model by a sphere instead of the usual cylinder.

4.4 All 12 Pages

Now that we have the code to generate the pages of our knot, we can create a version with twelve thickened pages each of $\frac{\pi}{6}$ thickness in the complex plane. However, material tolerances for the PLA plastic used in 3D printing needs to be considered. So instead of making each page exactly $\frac{\pi}{6}$ thick, we can generate each thickened page of $\frac{\pi}{6} - \epsilon$ thickness. After experimental trial and error, a value of $\epsilon = \frac{\pi}{40}$ was sufficient. The exact subset of $\mathbb{C}$ can be seen below.

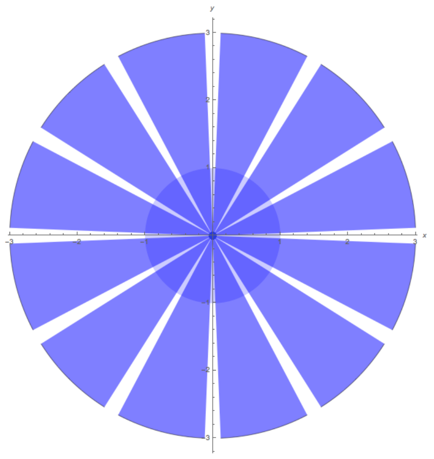

Figure 10: Trefoil knot + 12 pages in $\mathbb{C}$.

By letting Mathematica create each page with a PlotPoints of 200, we can import all twelve pages and the trefoil knot into Blender to view the unedited model. This will be the original mesh of the puzzle. Color has been added to represent the future 3D printed puzzle.

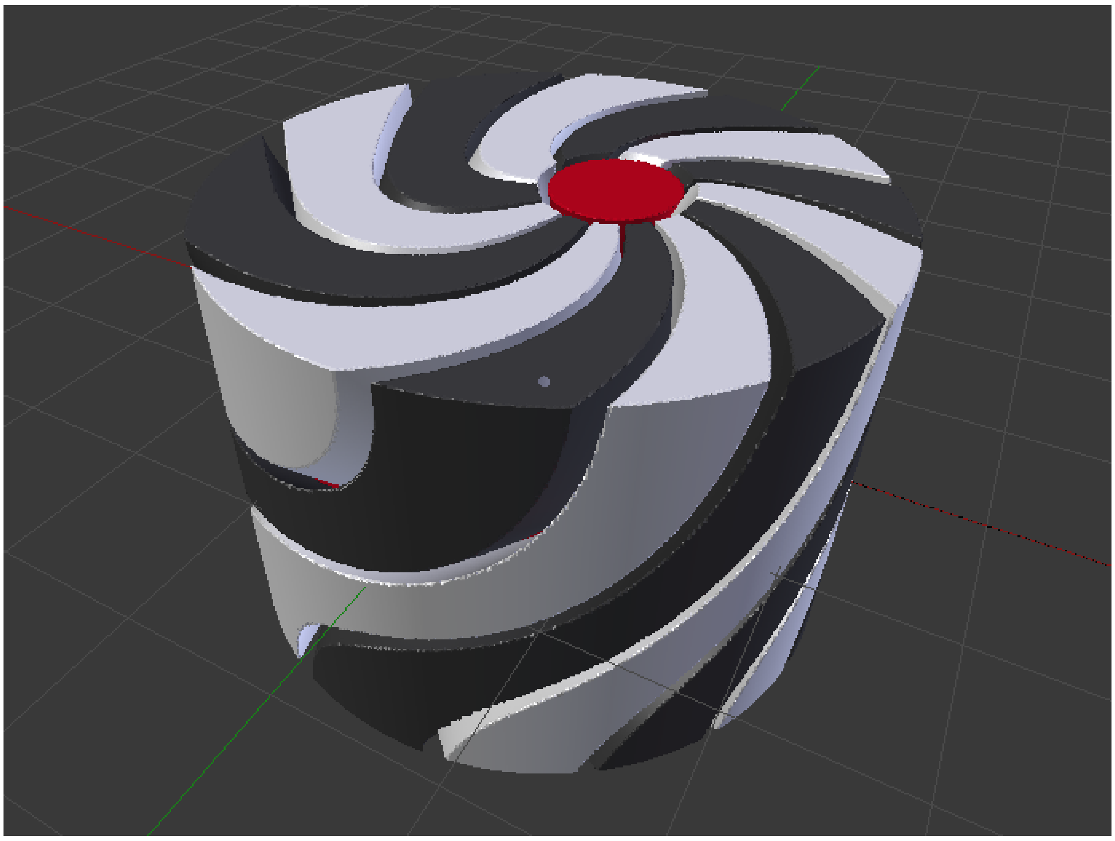

Figure 11: The original, unedited mesh of the trefoil knot + 12 pages.

5 Blender Manipulations

We now have thirteen unedited 3D meshes, one for each thickened page and one for the trefoil knot itself. In order to create the puzzle, we need to break up the original meshes in order to create our puzzle pieces. Ideally we should be able to place the trefoil knot at the core of the puzzle and attach any or all of the twelve pages to create multiple configurations. After much thought about the best way to achieve this, the most natural cut to make was along $z = 0$, i.e. splitting the model in two symmetric halves. This cut would allow one to pick up a completed puzzle and open it up to look at the internal structure. Once we established how we wanted to cut the model up into puzzle pieces, we also need to cut the trefoil knot so that it can lay inside each half. Furthermore, we need to label each piece in order to facilitate puzzle construction. Finally, holes need to be added to each piece in order to glue in magnets after 3D printing.

5.1 Horizontal Cut

Consider just the twelve page meshes and leave the trefoil knot whole for now. In order to cut the meshes in half, we can generate the infinitely thin plane $z = 0$ in Blender and add boolean modifiers to split the original models. Blender contains boolean modifiers such as intersect, difference, and union, all of which carry the usual mathematical meaning when applied to subsets of $\mathbb{R}^3$. So with a boolean modifier we are able to split these twelve meshes into twenty-four meshes.

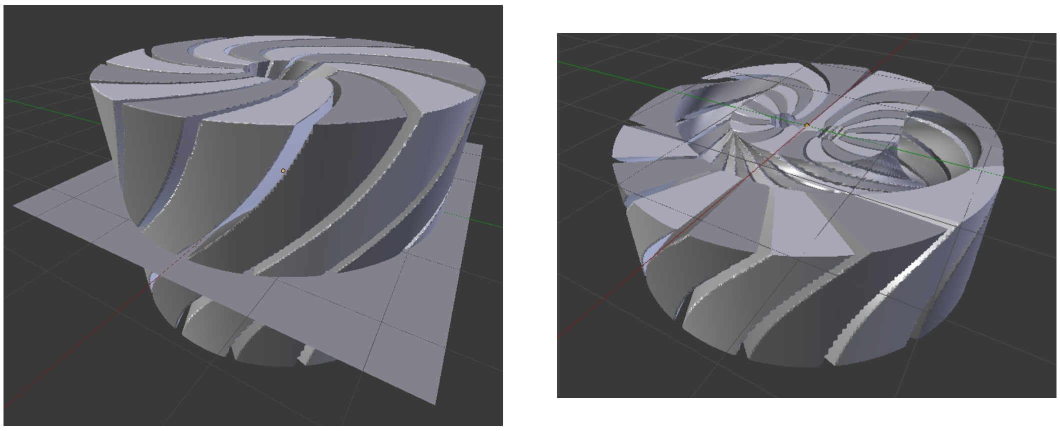
Figure 12: Before and after the horizontal cut.

5.2 Accounting for Pages 6 and 7

As stated before, cutting along $z = 0$ seemed like the most natural cut to make. If we inspect each page individually, we will notice that each page contains two holes. The $z = 0$ cut happened to cut open both holes on most pages, which allowed for us to place the trefoil knot inside one half of a page, and then connect the second half, trapping the trefoil knot inside. Unfortunately pages 6 and 7 (labeling to be discussed in the following section) were unique in that the two holes contained in each page were not split by $z = 0$. In order to get around this, we made another cut depicted below with the two new pieces dyed white.

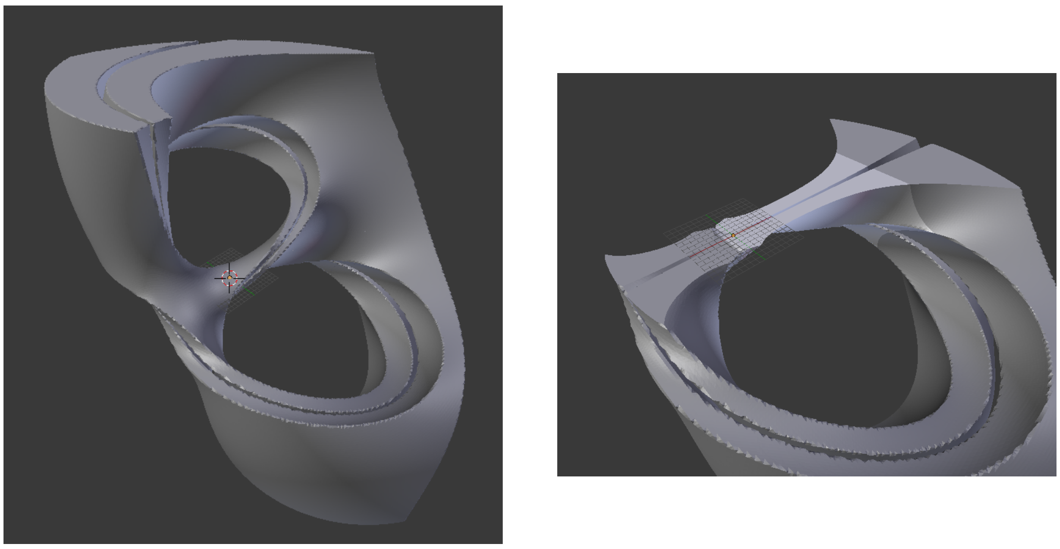
Figure 13: Pages 6 and 7 before and after the $z = 0$ cut with colored pieces.

5.3 Enumeration and Adding Holes

Now that we have twenty-four half pages with our correction of pages 6 and 7, we need a way to distinguish each half page from one another, thus providing a hint as to how each piece should fit together. Recall the final puzzle has twelve pages, so to distinguish each page from one another we can label each piece from 1 to 12, where page 1 is directly above the $\mathbb{R}$-axis in $\mathbb{C}$ and is bounded by the angles 0 to $\frac{\pi}{6}$ (neglecting the ϵ subtraction). The remaining pages move counterclockwise in numerically ascending order, until page 12 is a reflection of page 1 about the $\mathbb{R}$-axis.

Instead of adding an extra mesh to each page, we can use another boolean modifier to inset a numbered label into the piece. To do this, consider the 3D meshes of the numbers 1 to 12 that have already made by Thingiverse user **6brueder**. With the numbers 1 to 12 imported as 3D meshes, we can make use of the difference boolean modifier to inset the numbers into each half page as depicted in the example below.

In order to use magnets to hold together all the pieces of the puzzle, holes need to be added. Instead of manually drilling these after 3D printing, we can create a cylindrical mesh with slightly larger dimensions than the magnets to be used and perform a similar boolean difference operation on each piece to add an inset hole. On each half page, two holes were added to connect to its corresponding matching half page.

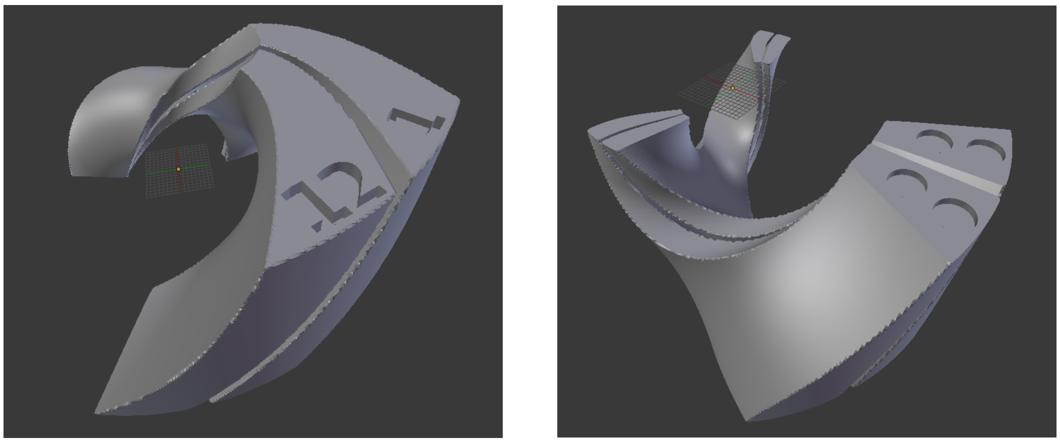

Figure 14: Pages 1 and 12 enumerated and added holes for magnets.

For the extra two pieces on pages 6 and 7, three more holes were added. Examples of where the holes were paced can be seen below in pages 6 and 7 and the extra two puzzle pieces.

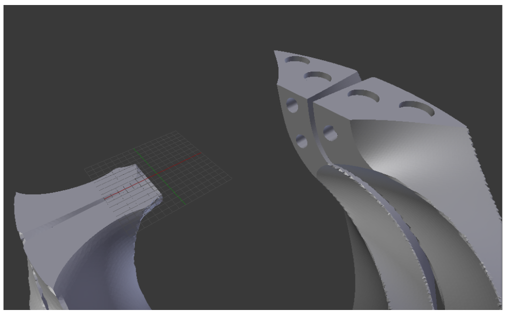
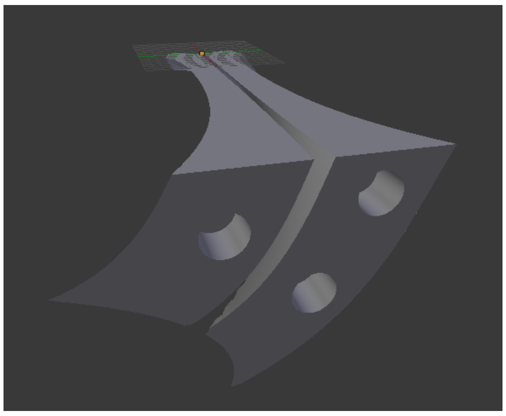

Figure 15: The extra holes needed for pages 6 and 7.

5.4 Trefoil Knot Cuts

Now that we have a complete set of twelve half pages with inset magnet holes and numerical labels, the last mesh to be edited is the trefoil knot itself. Following a similar process from the pages, we will first cut the trefoil knot with $z = 0$. Once we make this cut the trefoil separates into four pieces. We now need to determine the best way to cut the trefoil knot so that it fits inside the pages while minimizing the number of pieces as to not make the construction of the knot too complicated. This process was done primarily by experimentation. Ultimately, the best method came about by cutting the trefoil knot into six pieces total. The extra cut made can be seen below.

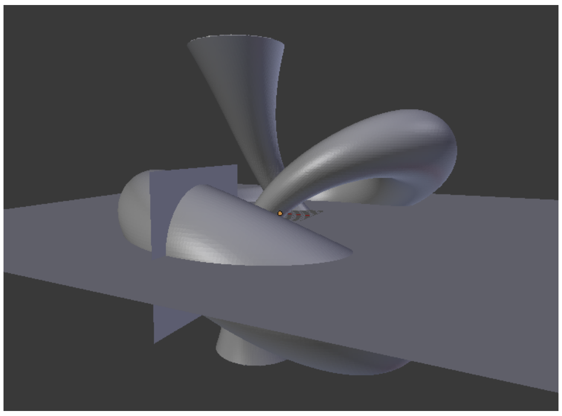

Figure 16: The two planes made to cut the trefoil knot.

Now that we have six trefoil knot pieces we need to add magnet holes similarly as we did to the half pages. Magnet holes were placed at every spot two pieces came together so that once assembled the knot would be sturdy.

6 Print Preparation

Once we have a completed model that is ready to print, we need a final piece of software that cuts the model into layers that the printer can recognize. MakerBot Desktop is the proprietary software that is bundled with the MakerBot Replicator and prepares a model for printing. Once imported, there is a list of options that can be edited to give specific control over certain characteristics of a print. The main options are

- Infill Percentage: how solid or hollow a print will be.
- Number of Shells: number of boundary walls on edges.
- Layer Height: how thick each layer of plastic will be extruded.
- Raft: provides base layer of plastic on which the object is printed on to ensure the print sticks to the build plate and prevent edge curling.
- Supports: provides built-in towers of plastic that hold up overhanging pieces of a print; removed when the print is completed.

Other options are included such as controls for temperature and print head speed, but those are left default for all prints considered.

Since the objects we are printing are puzzle-pieces and will be under user-stress during construction of the puzzle, the following option values used are specified below (if option is not listed assume default value):

- Infill Percentage: 15%.
- Number of Shells: 3.
- Layer Height: 0.20 millimeters.
- Raft and Supports: on.

7 Completed Puzzle

7.1 Gallery

The final, completed puzzle can be seen below. A full gallery can be seen by going to **fredhohman.com/projects**.

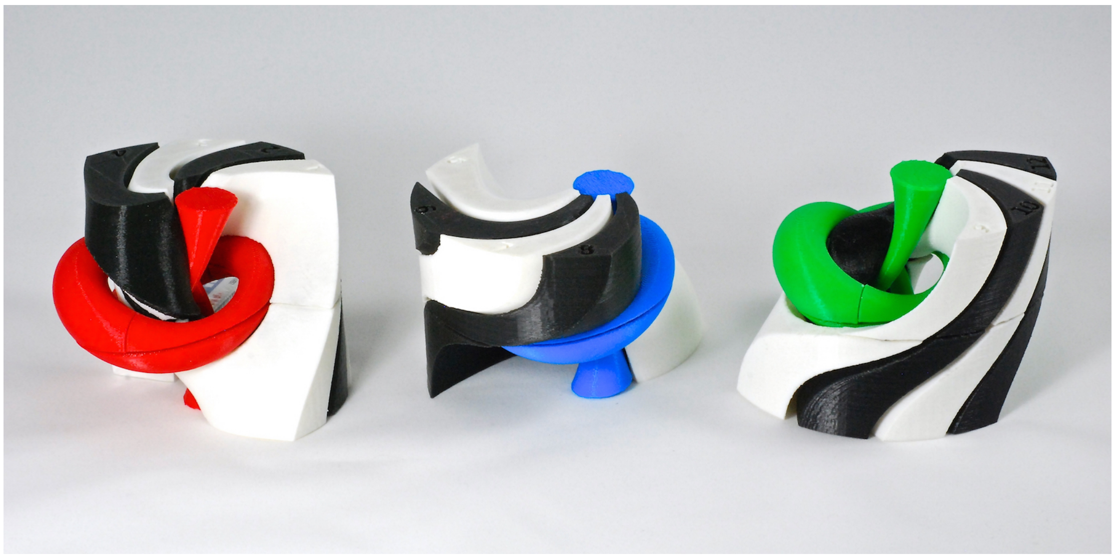

Figure 17: Trefoil knots + 4 pages.

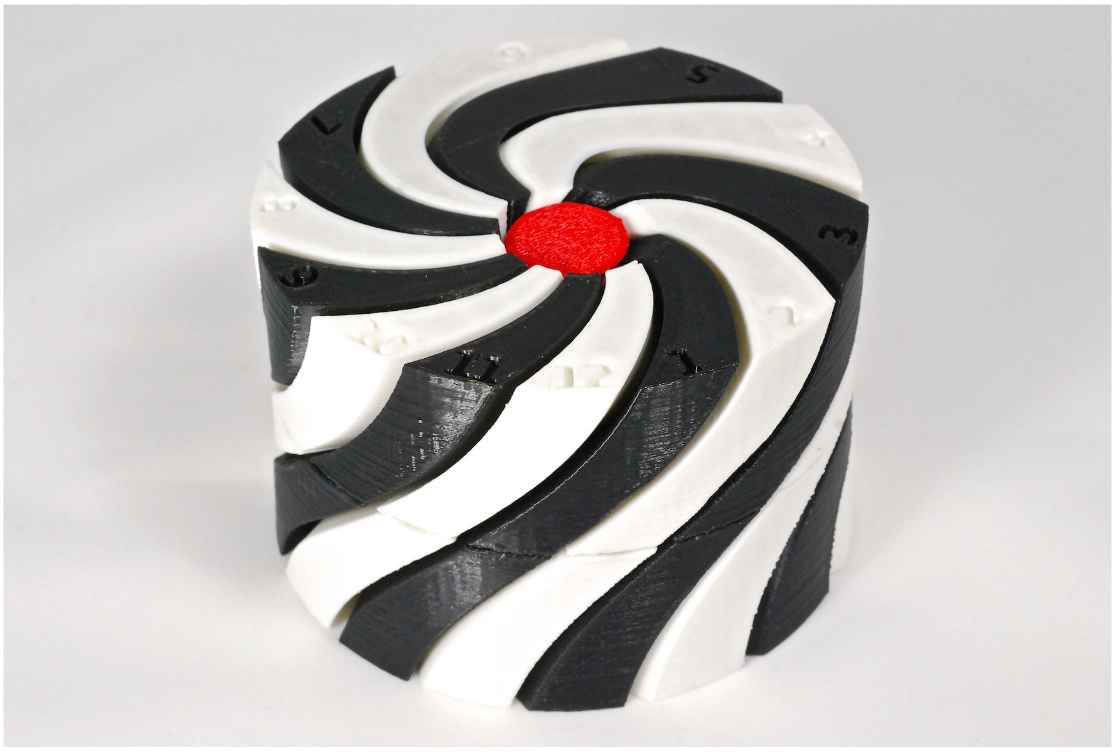
Figure 18: Trefoil knot + 12 pages.

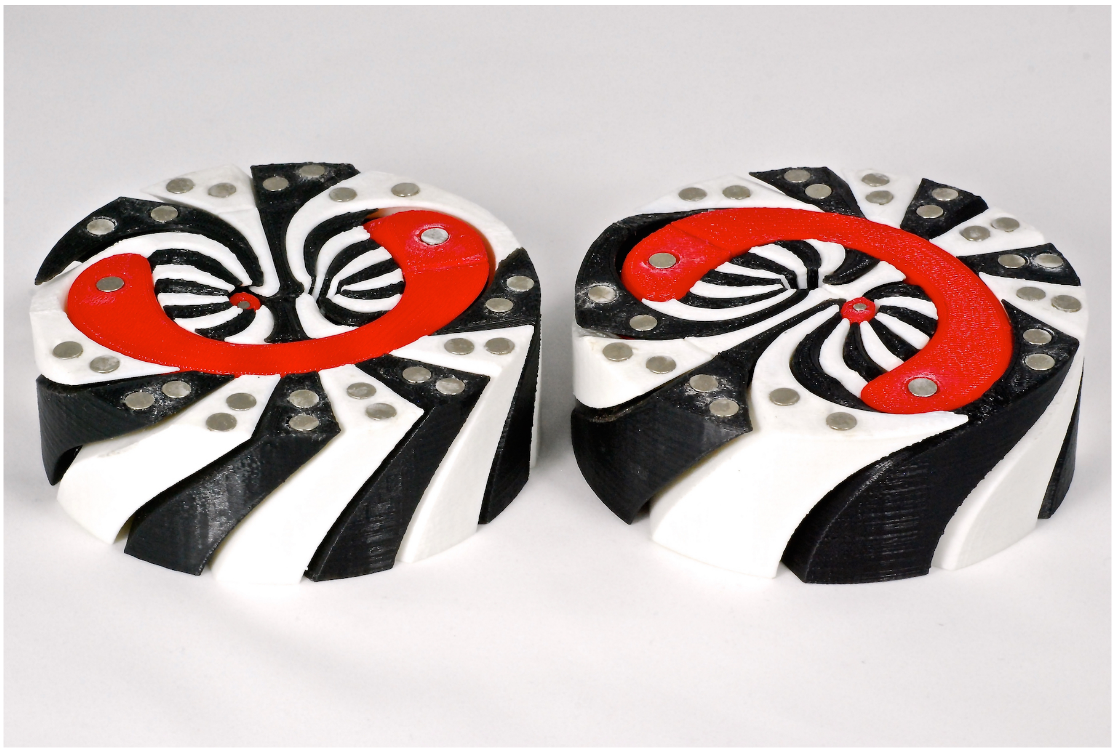
Figure 19: Trefoil knot + 12 pages in half.

7.2 Construction Instructions

While there is more than one way to put the puzzle together, the following method is known to work well:

- Identify piece numbered 5.
- Identify piece numbered 4.
- Tuck piece 4 into piece 5 until one side of each piece is flush together and the magnets are pointing the same direction.
- Continue this procedure using numbered pieces the in the following order: 3, 2, 1, 12, 11, 10, 9, 8, 7, and finally 6.
- If pieces 6 and 7 do not cooperate, it can sometimes be beneficial to try and insert both pieces at once.
- Locate six pieces of the trefoil knot (three unique pieces, two copies of each), and identify the two that contain multiple magnets.
- These two pieces should look different and will fit magnet-side up just as all the half page pieces numbered 1 through 12.
- Repeat this process for the second half of the puzzle.
- Combine both halves in the correct orientation using the magnets. One will hear a snap when the correct orientation has been found. As a hint, note that two half pages should connect if their number labels sum to 13.
- Take the last two pieces of the trefoil knot and place them into the top and bottom of the puzzle.

8 Topological Observations

Now that we have a completed puzzle and the .stl files to print more, we are able to notice some interesting characteristics of the puzzle, the trefoil knot, and its pages.

If we inspect the puzzle by itself, one observable symmetry is that each half of the puzzle (trefoil and pages included) are exact copies of each other. This way, if we were to print another complete puzzle, we could print just one half of the puzzle twice. Another consequence of this is that given just one half page of the puzzle, it has two other half pages that can complete the single page. In the above pictures we have alternated pages by using white and black plastic. Since each half of the puzzle is replicated, we can disassemble the puzzle and rebuild it such that instead of alternating white and black half pages, we can build one solid half of all black half pages and similarly one half of all white half pages.

Topologically we can examine more mathematical properties of one page. If we were to consider a page, i.e. a subset of $\mathbb{C}$ that is not $\frac{\pi}{6}$ thickness but infinitely thin, we know this page is a two-sided surface in $\mathbb{R}$. We know that every two sided surface is homeomorphic to some surface on the Classification of Surfaces list, so to identify this surface, we can make use of the rules given in Farmer, D.W. and Stanford, T.B.'s book *Knots and Surfaces: A Guide to Discovering Mathematics*. These rules, which are standard throughout topology, include bending and stretching without tearing, and making cuts and twists but reassembling the pieces so they fit together the same as originally constructed. This gives us a way to see how this surface is connected, as opposed to its size and distance.

If we inspect just one page in the image below, then we can perform the rules to draw the surface schematically in the plane. To do this, we can make four cuts that dissect the page into three pieces: one disk and two rectangles. The rectangles are the "handle-like" structures that are twisted. Note that we need to ensure when we make the cuts we preserve the way in which the two rectangles are connected to the main disk. Once we make the cuts, we can deform our pieces to reasonable sizes, and reattach the two rectangles at the appropriate locations. When we do this, we obtain the picture below of the page in the plane.

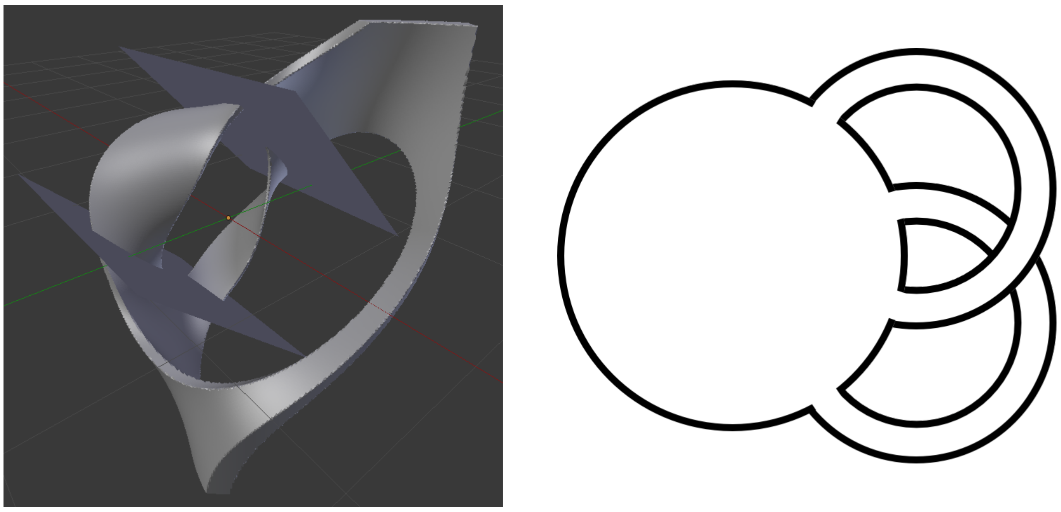
Figure 20: Approximately infinitely thin page and its reconstructed planar diagram.

From Farmer and Stanford, we know a surface can be described as the connected sum of some number of tori, each with some number of holes. To count the number of holes, we can count the number of boundary curves. To accomplish this, trace the edge of the surface until you arrive back at the starting point. When we do this for our surface, we cover every curve, leading us to say that our surface has only one boundary curve. Since we know our surface has one boundary curve, i.e. one hole in it, we know a single page is homeomorphic to *something* with one hole.

To determine what this *something* is, we draw a graph on our surface that breaks the surface into cells. Recall a cell is a region on a surface where every loop in contractable, i.e. can be shrunk down to a single point. In our case, the graph consists of four line segments visualized in light gray below, where the vertices are red dots.

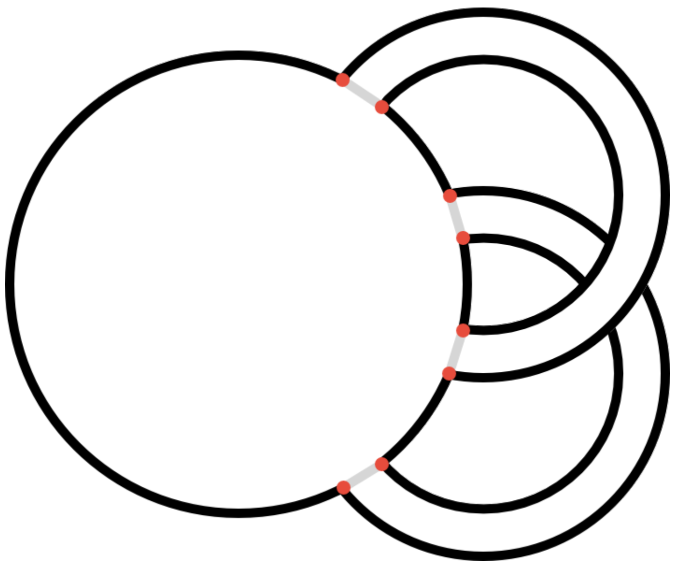
Figure 21: Single page planar drawing with imposed graph.

We can now compute the Euler Characteristic; this way we can distinguish between the sphere and the torus on the Classification list. Recall the Euler Characteristic is defined as

$$\chi = V - E + F,$$

where V is the number of vertices, E is the number of edges, and F is the number of faces. When we count the vertices, faces, and edges of our surface once we impose a graph on it, we find that

$$V = 8, \quad E = 12, \quad \text{and} \quad F = 3.$$

Therefore, we compute

$$\chi = 8 - 12 + 3 = -1.$$

So with one boundary curve and an Euler Characteristic of $\chi = -1$, we know that each page is homeomorphic to a torus with one hole.

9 Thanks

I would first like to thank Dr. David Gay for his continual guidance, suggestions, and mentorship over the past year and a half. Furthermore, I would like to thank Mo Hendon for giving me practice to present my results as I was performing research throughout the duration of my project, and Dr. Jason Cantarella for reviewing this thesis. Lastly, I would like to thank The University of Georgia for providing facilities, supplies, and monetary support for all the 3D printing experimentation that has been done.

10 References

[1] Farmer, D.W. and Stanford, T.B. *Knots and Surfaces: A Guide to Discovering Mathematics*. American Mathematical Society. 1995.

[2] N. D. Gilbert and T. Porter. *Knots and Surfaces*. Oxford University Press. 1996.

[3] Jeffrey R. Weeks. *The Shape of Space*. CRC Press. 2001.

[4] 6brueder. *All Alphabet Letters A-Z*. Thingiverse.com. 2011.

[5] henryseg. *Stereographic projection*. Thingiverse.com. 2013.