Semester Project Report: Computational Design of Stable Crochet

Tal Rastopchin

June 15, 2024

1 First Crochet Experiments

I started the semester project by collecting and exploring relevant papers on crochet, knitting, and other related methods of fabrication in the visual computing / graphics / geometry modelling literature. After completing the literature review, I met with Professor Sorkine-Hornung to discuss what I learned, as well as brainstorm some ideas for tackling the problem of accurately and stably reproducing highly-curved surface patches. The most important takeaway from the discussion is that we needed to get a more comprehensive understanding of what is possible. Namely, we wanted to get a better understanding of the design space of accurate, stable, and highly-curved crocheted surface-patches.

In order to explore this design space, we first identified a couple parameters we hypothesized might be related to a crocheted object's accuracy and stability:

1. Crochet parameters: yarn material, yarn weight, hook width, gauge (determined by the combination of yarn weight and hook width).
2. Shape parameters: Gaussian curvature, mean curvature, minimality, open / closedness.

1.1 Crochet Parameters

We hypothesized that using the least-elastic yarn would promote stability of the resulting crocheted object. I did a bit of research, and I found that the elasticity of a yarn is mostly determined by curvature of the underlying individual strands. Jillian Moreno's blog post on yarn's drape and elasticity explains that "the more crimp in a fiber, the more elastic it is. Crimp can range from intense zig zags all the way to waves. Crimp acts just like a spring, the more crimp the more elasticity." Many sources that I found explain that wool gets its elasticity from the crimp of the underlying natural fiber. I could not find a specific source claiming that cotton fibers have less crimp, but I did find many sources claiming that cotton is less elastic compared to other common fibers materials.

We also hypothesized that using a tighter gauge, which can be achieved by using a hook with a smaller width than recommended for the given yarn weight, would promote rigidity, and hopefully stability, of the resulting crocheted object. Fiber artists know that using a tighter gauge makes the spaces in knitted fabric smaller, which decreases the fabric drape, resulting in a more rigid fabric.

Hence, for the first set of crochet experiments, I chose to use cotton yarn to crochet the shapes of interest. I used "katia easy knit cotton", which is 100% cotton and recommends a 5.5-6mm crochet hook. Each ball is 109 yards and I used about two balls in total.

I used a slightly smaller hook width than recommended, but I did not further vary the gauge. I used both inline and tapered crochet hooks with widths between 4.5-5mm. (For the second set of crochet experiments, I further varied the gauge by using an even smaller hook width.)

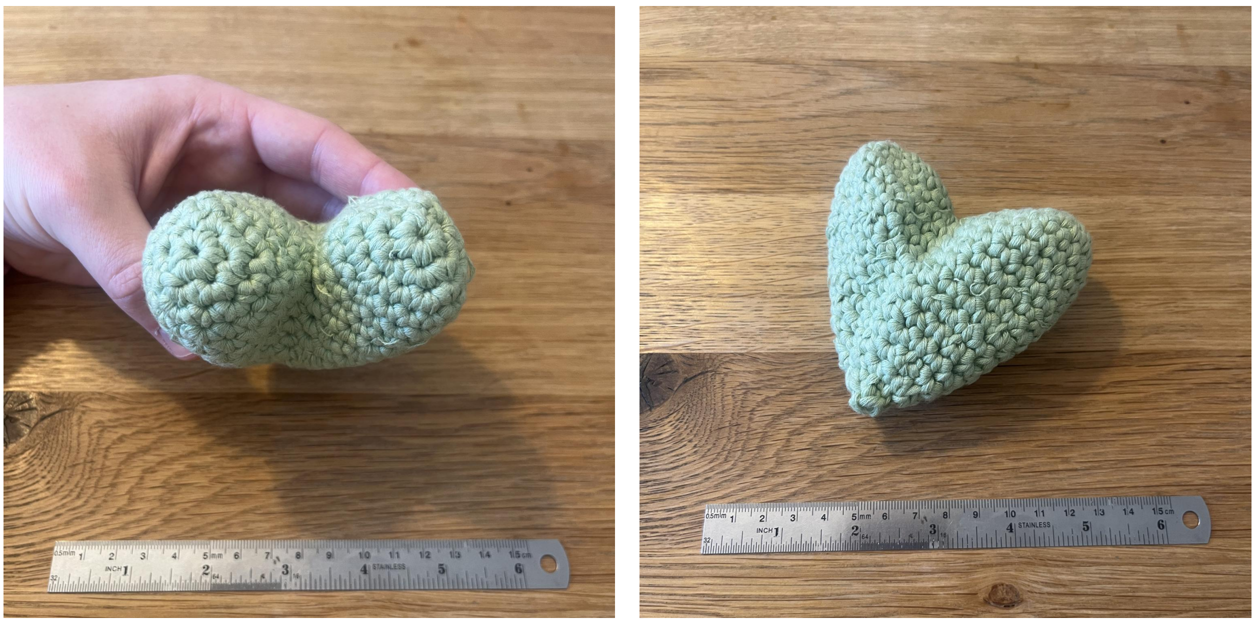
1.2 Shape Parameters

We hypothesized that shapes with high curvature would inherently be more stable than shapes with low curvature. In their paper on developable metamaterials, Signer et al. explain that they control the resistance to deformation of their metamaterials by introducing curvature: "The curvature introduces bending energy (thereby elastic energy), which is proportional to the mean curvature squared" [5]. Professor Sorkine-Hornung further hypothesized that a closed shape might be more stable than an open shape, due to the lack of a boundary. Hence, we wanted to explore how a shape's Gaussian curvature, mean curvature, and minimality, and closed / openness might relate to its crocheted accuracy and stability. To do this, we identified three shapes of interest, which cover our shape parameters:

1. A classic saddle which exhibits negative Gaussian curvature, and is open.
2. A heart, which exhibits both negative and positive Gaussian curvature, and is closed.
3. A catenoid, which has negative Gaussian curvature, is a minimal surface, having zero mean curvature, and is open.

1.3 Heart

I used Mignon Crochet's heart amigurumi pattern to create the heart. I used a 5mm inline hook. Because we're interested in stability, I did not stuff the object, and left it empty.



(a) Top view (b) Side View

Figure 1: Heart

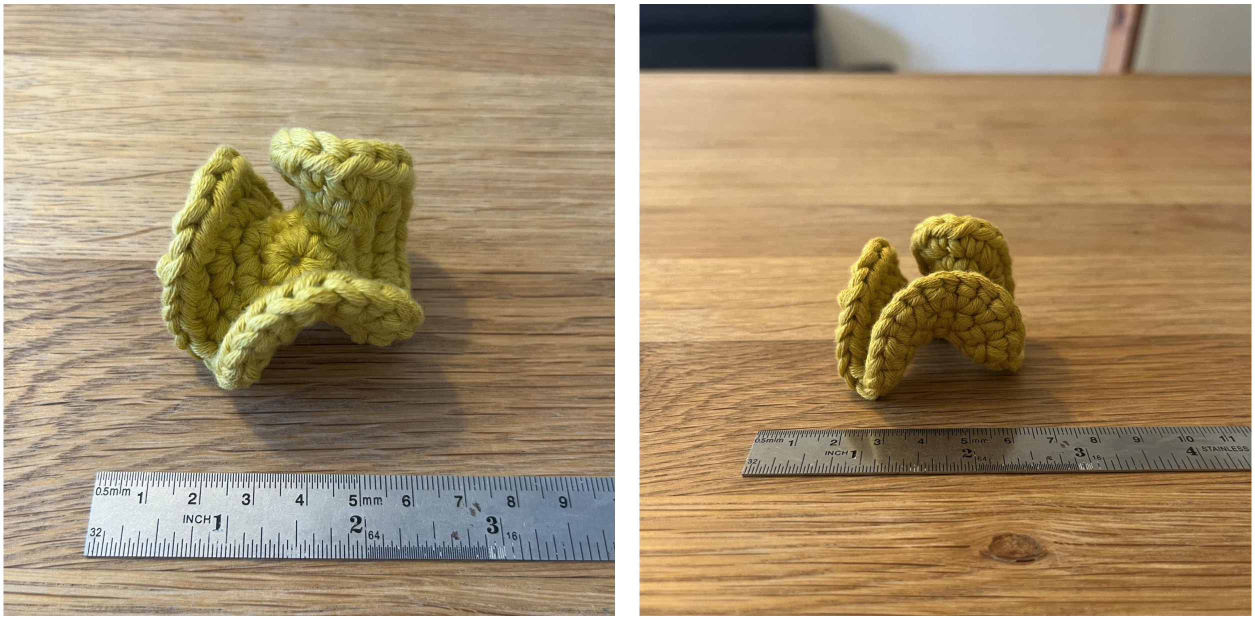
While the heart does maintain some of its shape, it easily flattens to the classical 2D representation of a heart. Before I took the pictures I squeezed it so that it would retain some of it's 3D shape. The crochet and shape parameters of this experiment did not produce a crocheted result that was stable.

1.4 Saddles

I used Daina Taimina's recipe for crocheting the pseudosphere in order to generate the patterns for three saddles. Each saddle starts with an adjustable ring consisting of 6 or 8 crochet stitches, and the subsequent number of stitches in each row grows according to a constant multiplicative rate. My intuition tells me that it's a lot easier to obtain stability at a smaller scale than at a larger scale. So, I crocheted three different saddles with decreasing growth rates, with the goal of better understanding how different curvatures might produce different levels of stability.

1.4.1 Saddle A

The first saddle consists of 6 starting stitches, a growth rate of 2, and 4 rows. I used a 4.5mm tapered crochet hook.



(a) Top view (b) Side view

Figure 2: Saddle A

You can see that it has a high curvature which at the smaller scale corresponds to a higher rigidity and stability. It easily holds itself up.

1.4.2 Saddle B

The second saddle consists of 8 starting stitches, a growth rate of 1.5, and 6 rows. I used a 5mm tapered crochet hook.

Even though this saddle has a lower curvature than the previous, it is still quite rigid and stable, and still holds itself up.

While the previous saddle's pattern was easy to generate since every stitch was an increase, I had to get more creative to determine the placement of the increases on each row. I implemented a very simple saddle pattern generator in Python that computes the number of stitches in each row and uses a greedy algorithm to determine the location of the increases. I got the resulting pattern stitch-by-stitch, which was not friendly to follow, and so I implemented very simple loop folding, inspired by AmiGo's approach [2]. It took me quite some time to get the script to generate a pattern that made sense and that was human-readable, but it was very worth it because it served as a foundation to crochet the rest of the items in this series. I use the same script to create Saddle C.

1.4.3 Saddle C

The third saddle consists of 8 starting stitches, a growth rate of 1.38, and 9 rows. I used a 5mm tapered crochet hook.

Just like with the second saddle, even though it has a lower curvature than the previous, is still quite rigid and stable, and still holds itself up.

1.5 Catenoids

I crocheted the same catenoid at two different scales. The catenoid is parameterized as a surface of revolution, where it's height (cosh) runs from $-\operatorname{arccosh}(2)$ to $\operatorname{arccosh}(2)$. Inspired by Daina Taimina's algorithm for generating her catenoid / helicoid pattern, I first made a gauge in order to estimate my single crochet stitch width and stitch height. Then I extended the saddle pattern generator by

(a) Top View (b) Side View

Figure 3: Saddle B

(a) Top View (b) Side View

Figure 4: Saddle C

1. implementing the connection of rows that decrease in the number of stitches, and
2. (brute-force) discretizing the catenoid along its generatrix (cosh) and along its radiuses (circles) using the measurements determined from my gauge.

This again took some time to get working, but it was very worth it, as it gave me a human-readable pattern for the desired shape of interest.

1.5.1 Gauge

The gauge consists of a 10 by 10 single crochet swatch. I used a 5mm tapered crochet hook. Measuring the swatch, I determined a stitch width and height of 7.7 cm / 10 stitches = 0.77 cm.

Figure 5: Gauge

1.5.2 Catenoid A

The first catenoid is scaled by a factor of 10/4. I used a 5mm tapered crochet hook. (I didn't use the 10 x 10 swatch to estimate my single crochet stitch height and stitch width; I used a smaller gauge, which gave me a stitch height of 1.00 cm and a stitch width of 0.77 cm.)

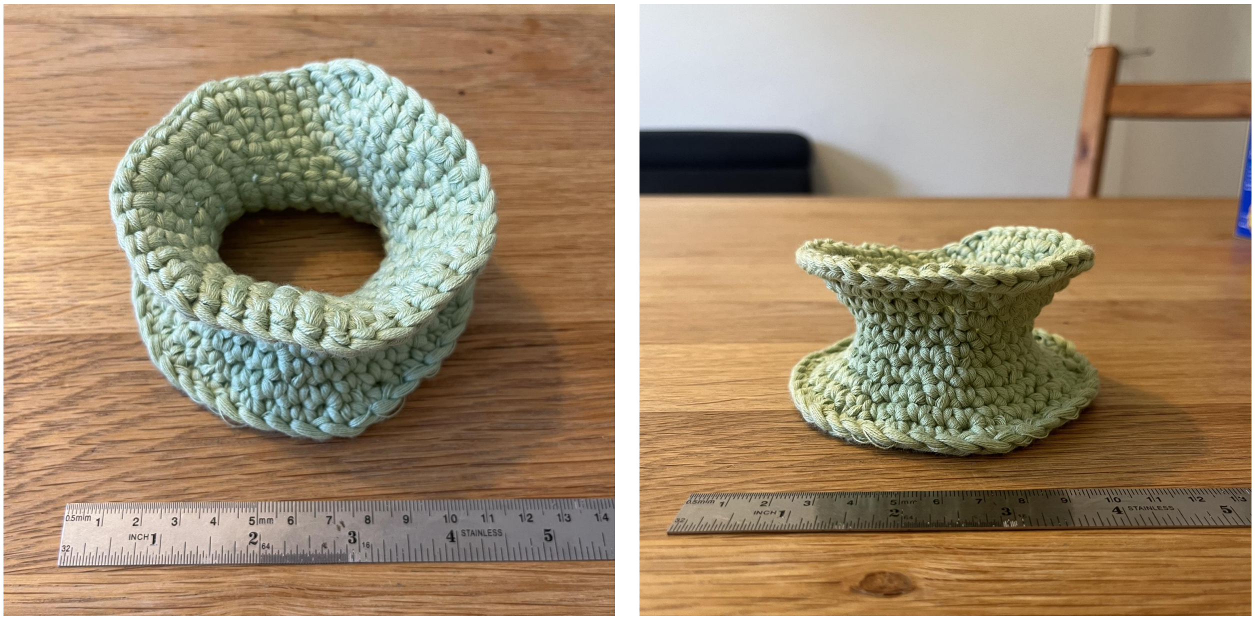
I chose the first smaller scale factor of 10/4 both to verify that my generated pattern would approximate the catenoid and second to create a smaller catenoid. You can see that it closely resembles the catenoid and that it appears quite rigid and stable.

1.5.3 Catenoid B

The second catenoid is scaled by a factor of 5. I used a 5mm tapered crochet hook.

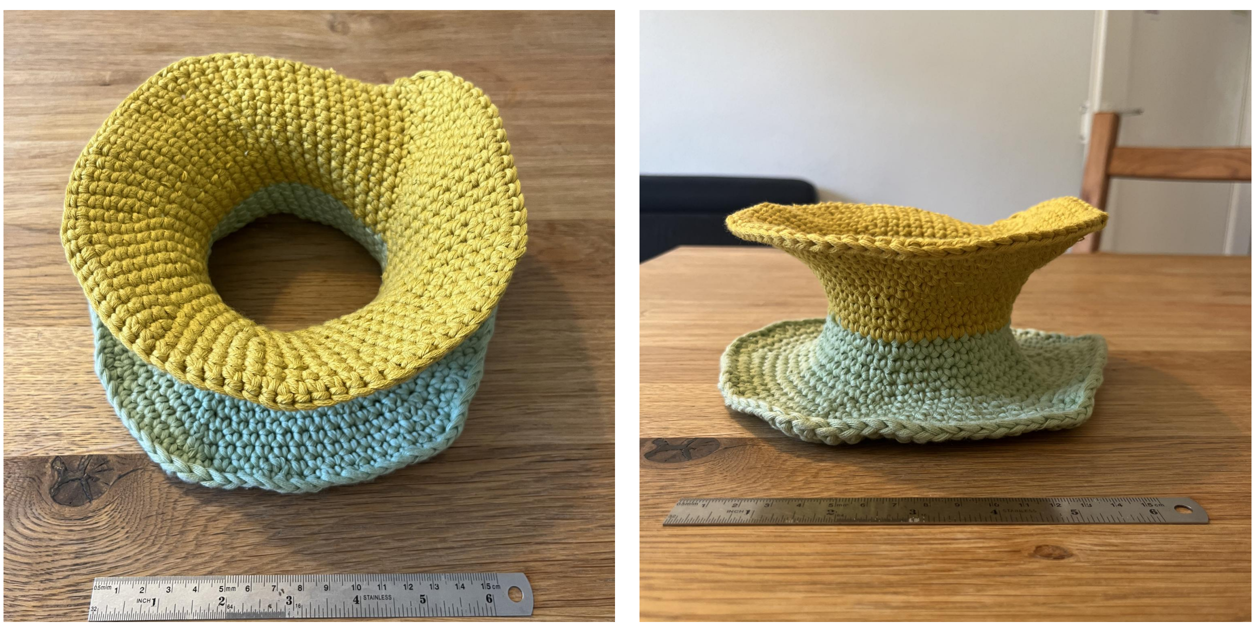
I chose a scale factor of 5 in order to push the last crocheted object in this series to its limit. Since my intuition told me that it's easier to crochet smaller stable objects, I wanted to push the scale to its limit to see if we start to lose accuracy or stability. The resulting bigger catenoid is less accurate and stable than its smaller counterpart. Namely, the weight of the object pulls the lower half of the object to the surface of the table. Furthermore, the top boundary also droops down a bit. Regardless, it does overall hold its shape quite well, especially away from the boundaries.

While crocheting the object, I noticed that the evolving boundary was always drooping until a couple more rows were added. This matches up with Professor Sorkine-Hornung's hypothesis that a closed shape



(a) (b)

Figure 6: Catenoid A



(a) Top view (b) Side view

Figure 7: Catenoid B

might be more stable than an open shape, due to the lack of a boundary. My hypothesis is that because stitches along the boundaries don't have the influence of surrounding stitches in every direction of the tangent plane, forces are missing that would be required to approximate the isometry of the surface.

2 Second Crochet Experiments

I met with Professor Sorkine-Hornung to discuss the results of the first crochet experiments, and we refined our exploration direction.

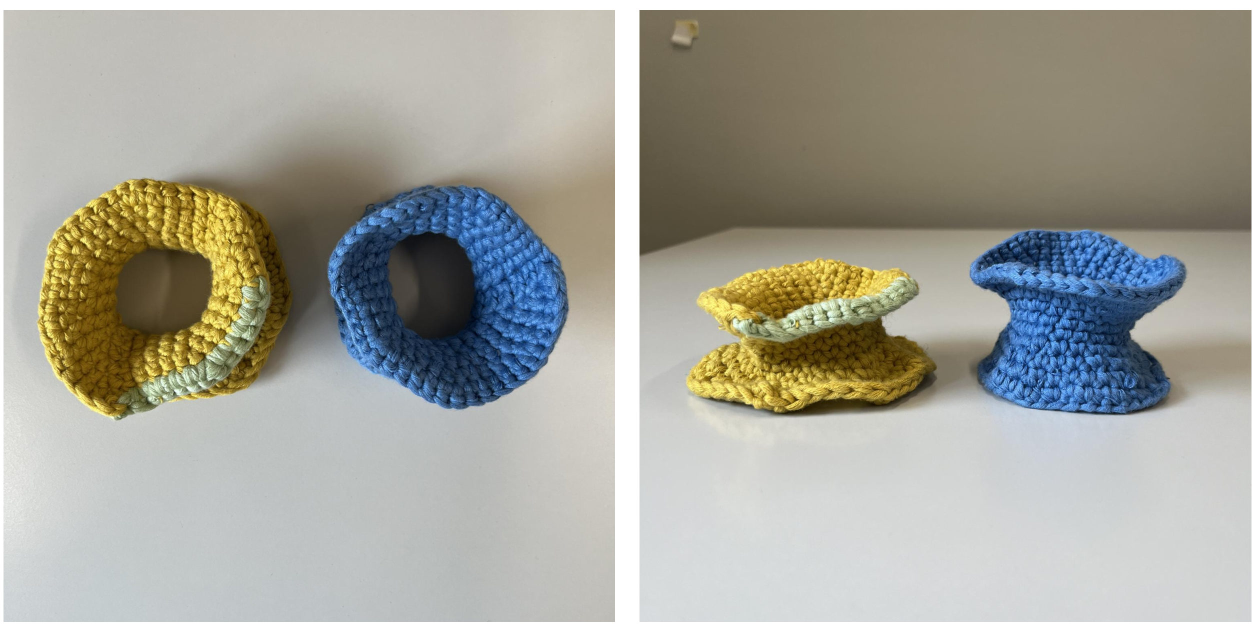
First, we agreed that I should explore the variation of the hook width, by further tightening the gauge, for completeness.

Second, based on the results of the first experiment, we wanted to explore a different idea for promoting the stability of crocheted objects. Namely, we wanted to explore a metamaterial approach. The results of the first set of crochet experiments demonstrated to us that stability is possible, but it didn't illustrate a straightforward relationship between the crochet parameters, shape parameters, and resulting crocheted accuracy and stability. At smaller scales, saddles A, B, and C demonstrate that high Gaussian curvature does in fact promote rigidity and stability. At a much larger scale, catenoid B illustrates that locally, far from the boundaries, some level of accuracy and stability is attainable. However, the attained stability qualitatively appears less than the smaller objects with higher Gaussian curvature.

Hence, we hypothesized that a metamaterial approach that tiles a surface with a highly-curved surface patch might be able to promote stability at larger scales.

2.1 Hook Width Variation

For the hook width variation, to produce a tighter gauge, I decreased the hook width. I kept the pattern (catenoid_a.txt) and yarn ("katia easy knit cotton") constant, and used a 4.5mm and 4mm tapered crochet hook. Below you can see the catenoid crocheted with the 4.5mm hook on the left (in yellow and green) and the catenoid crocheted with the 4mm hook on the right (in blue).



(a) Top view

(b) Side View

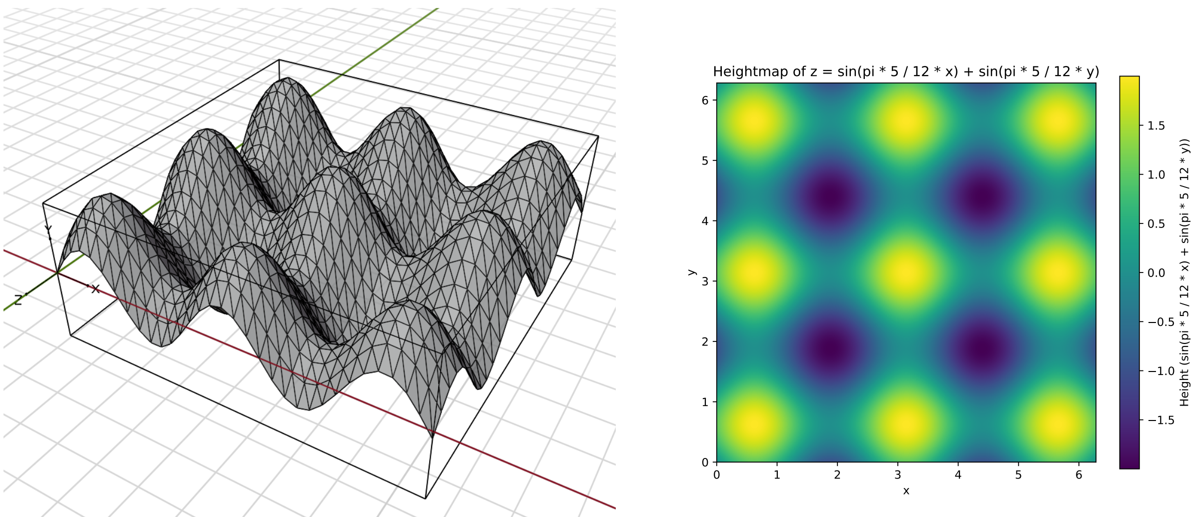
Figure 8: Catenoid Hook Width Variation

Interestingly, while the catenoid crocheted with the 4.5mm hook seemed just as floppy as that crocheted with the 5mm hook, the catenoid crocheted with the 4mm hook held itself up better and seemed a lot more stable. So, varying the hook width for completeness was fruitful, since it demonstrated to us, that decreasing from the recommended 5.5-6mm to 4mm made a qualitatively substantial difference in the resulting crocheted stability.

For this reason, I crocheted the rest of the objects in the second set of crochet experiments using a 4mm hook.

2.2 First Metamaterial Attempts

I started to test out the hypothesis that a metamaterial approach that tiles a surface with a highly-curved surface patch might promote stability at larger scales by designing a crochet metamaterial inspired by a sine curve. My design goal was to locally increase the Gaussian curvature of a target surface while globally approximating a similar shape. It is important to note, that at this point in the semester project, I deprioritized the goal of accurately crocheting shapes, and instead singly focused on stably crocheting shapes.



(a) Sine-surface in Blender

(b) Sine-surface heightmap

Figure 9: Sine-Surface Metamaterial Design

My intuition as a crochet artist told me that placing concave and convex "bumps" on an input surface is one way to locally inject Gaussian curvature into a target surface. For the case of a simpler shape, like a plane, I knew I could do something like this with a variation of the "sine-surface" $z = \sin(x) + \cos(x)$. Figure 9 illustrates a perspective rendering and a heightmap of a scaled variation of the sine surface. Furthermore, if a shape has a UV-parameterization, we could use the Sin-surface height values in UV space to displace the embedded parametric surface, effectively "tiling" it with concave and convex "bumps."

My first attempt at crocheting a Sine-surface metamaterial involved trying to adapt my catenoid pattern generation script to generate a pattern for a "bumpy" catenoid. I quickly realized that even if I could apply the Sin-surface displacement in UV space, the resulting shape was no longer a surface of revolution, and my algorithm would no longer work. I next spent some time trying to solve the problem for the Sin-surface by trying to reduce the problem to that of computing the length of each "crocheting row" on the Sine-surface. Starting with one edge of the Sine-surface's square boundary, I brute-force computed a series of constant Euclidean-distance contours (very similar to the wrapping step in Igarashi et al.'s algorithm [3]). Then, I used my greedy algorithm to determine the location of the increases and decreases between each row. However, this resulted in a pattern identical to that of a flat square. I realized that the greedy approach to connect two crochet rows of a given length only works in the case of a surface of rotation because of the rotational symmetry. Due to the rotational symmetry, equally spacing the increases / decreases worked perfectly. However, with the lack of such symmetry, equally spacing the increases / decreases completely ignores the embedding of a surface. It is not sufficient to use the row lengths and greedily connect them, one must take into account the embedding of each row. So, I reprioritized, and I decided that instead of spending time implementing an algorithm like Igarashi et al.'s to solve the inverse design problem for crochet, it would be more useful to use my crochet resources and intuition to try to manually design a Sine-surface-like crochet metamaterial.

My first attempt at manually designing a Sine-surface-like crochet metamaterial relied on my crochet intuition that I can locally inject Gaussian curvature by discretely inserting "too many stitches." One of my key takeaways from learning about and doing hyperbolic crochet is that the reason the crocheted hyperbolic planes and pseudospheres curl out of the plane is due to the fact that locally, there is "more area" than in the flat case. I transferred this insight to model a bump as an augmentation of a flat rectangular single crochet patch. Suppose we have a rectangular crochet patch with n rows each with m single crochet stitches. A bump can be made, by starting from the bottom row, progressively increasing the number of stitches in each row until the middle row, followed by progressively decreasing the number of stitches in each row until the last row. Since each bump is obtained by augmenting a rectangular single crochet patches, the bumps fit together as rectangles to tile the plane.

I crocheted a flat rectangular single crochet control swatch, as well as two variations of the "bump" tilings, as a first attempt at designing and constructing a crochet metamaterial.

2.2.1 Control

The control swatch is a flat rectangular 15 x 16 single crotchet swatch. I used a 4mm tapered hook. The control swatch's main purpose is to serve as a comparison to Metamaterial A and Metamaterial B. Namely, the control swatch is flat (even though its edges do curl up a little).

(a) Top view

(b) Side view

Figure 10: Control

2.2.2 Metamaterial A

Metamaterial A consists of 3 rows and 3 columns of bumps consisting of rectangular regions of 5 x 4 stitches. The pattern for the bump motif is given as follows. Starting with 5 contiguous stitches,

1. sc, inc, sc, inc, sc (7).
2. 2 sc, 3 inc, 2 sc (10).

3. 2 sc, 3 dec, 2 sc (7).
4. sc, dec, sc, dec, sc (5).

(a) Top view

(b) Side view

Figure 11: Metamaterial A

Qualitatively, the crocheted result feels a little more rigid and stable than the control swatch. It is notable that one cannot see the well-defined 3 x 3 tiling of the bumps.

2.2.3 Metamaterial B

Metamaterial B is a variation of Metamaterial A with more pronounced bumps. It also consists of 3 rows and 3 columns of bumps consisting of rectangular regions of 5 x 4 stitches. The pattern for the bump motif is given as follows. Starting with 5 contiguous stitches,

1. sc, 3 inc, sc (8).
2. sc, 6 inc, sc (14).
3. sc, 6 dec, sc (8).
4. sc, 3 d (5).

Qualitatively, the crocheted result feels even more rigid and stable than the control swatch, and a little more rigid and stable than Metamaterial A. One can see how it curves and lifts itself off the table. Furthermore, one can see the well-defined 3 x 3 tiling of the bumps. I would say that Metamaterial B supports the hypothesis that, in the simpler flat case, dramatizing the bump motif (by locally injecting more area via more aggressive increasing) results in a more rigid and stable crocheted result.

3 Inverse Design Problem for Crochet Implementation

I met with Professor Ren to discuss the results of the second set of crochet experiments, and we further refined the exploration direction of the semester project. We discussed

1. The idea for an algorithm that greedily places bumps to promote stability where they are needed most, and

(a) Top view (b) Side view

Figure 12: Metamaterial B

2. algorithmic approaches for the inverse design problem for crochet (since I was not able to adapt my scripts that generated patterns for surfaces of revolution to generate a pattern for the Sine-surface).

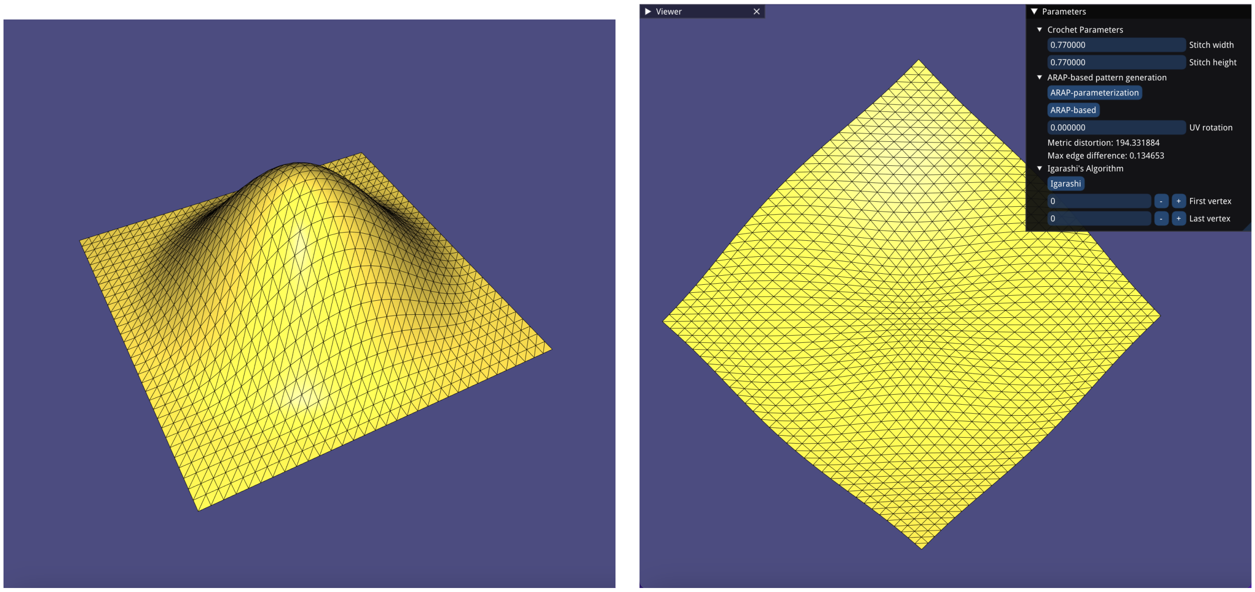
After crocheting Metamaterial A and B, my hand started to hurt while crocheting, I believe due to the extra amount of force necessitated by using a dramatically smaller hook paired with many increases and decreases. Hence, I prioritized implementing an algorithm that Professor Ren outlined, as well as a simplified version of Igarashi et al.'s algorithm for the inverse design problem for knitting.

3.1 ARAP-based Pattern Generation

Professor Ren proposed the following algorithm to automatically generate crochet patterns for topological disks. The key idea behind the approach is that one can use an ARAP parameterization to obtain a 2D parameterization of the input mesh that minimizes the metric distortion. Since the metric distortion is minimized, the hope is that a regularly sampled grid in the 2D UV domain would have similar edge lengths to the corresponding embedded grid. The algorithm proceeds as follows:

1. It first computes an ARAP parameterization of the mesh.
2. Then, in the UV domain, a row / column orientation is specified by the user, and a regular grid graph is sampled. The horizontal spacing is specified by the single crochet stitch width, and the vertical spacing is specified by the single crochet stitch height.
3. Lastly, the method uses the parameterization to push forward the regularly sampled grid graph from the UV space onto the 3D mesh. The resulting embedded graph represents the stitch graph corresponding to the crochet pattern.

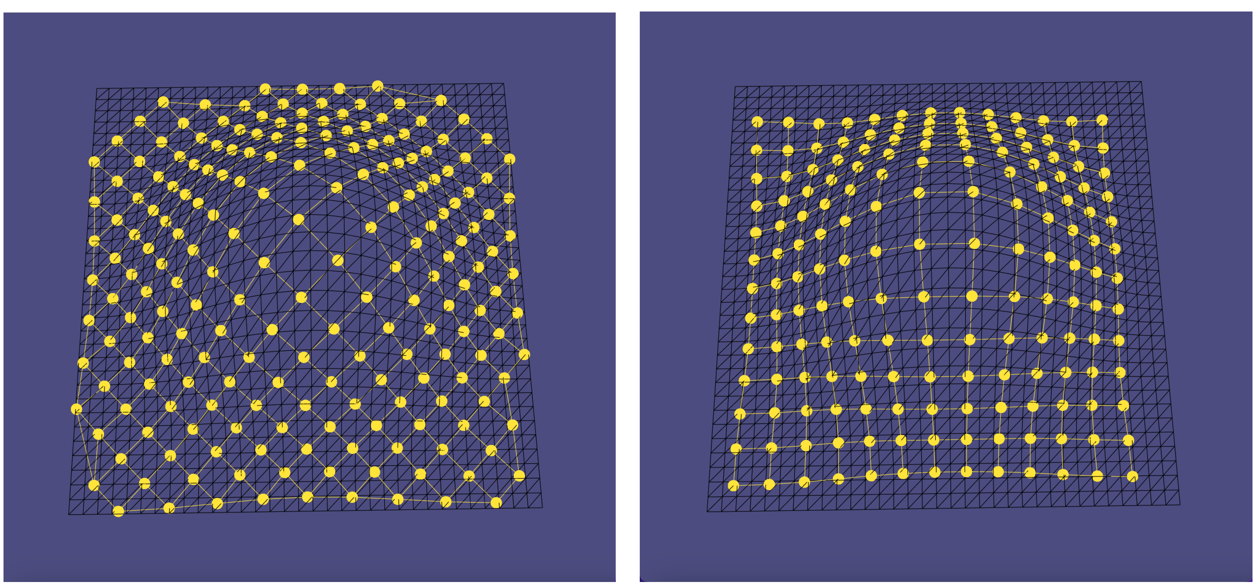
I implemented the ARAP-based pattern generation algorithm in C++ using libigl. I followed libigl's tutorial to perform the ARAP parameterization, using a harmonic parameterization as the initial solution. My implementation first allows the user to specify a stitch width and a stitch height. Then, they can click the "ARAP-parameterization button" to obtain the ARAP parameterization. After obtaining the ARAP parameterization, the user can click the "ARAP-based" button to visualize the generated stitch graph. The implementation uses the covariance matrix of the parameterization to choose an initial "optimal alignment" for the row / column orientation, and the user can further specify a rotation of the UV domain, given as



(a) bump.obj

(b) Its ARAP parameterization.

Figure 13: bump.obj and its ARAP parameterization.



(a) ARAP-based stitch graph with a UV rotation of 0.

(b) ARAP-based stitch graph with a UV rotation of $2\pi/8$.

Figure 14: ARAP-based stitch mesh generation of bump.obj

a float in the range 0 to 1, linearly mapped to 0 to 2π. The implementation further displays the metric distortion (the sum of the differences of the edge lengths) as well as the maximum difference between edge lengths.

Figure 13a is an input triangle mesh that I used to test my implementation, and figure 13b is the corresponding ARAP parameterization for the mesh. Note that the ARAP parameterization has a nontrivial metric distortion of about 194.33 and a maximum edge difference of about 0.13. Figure 14 displays two different stitch graphs produced by the ARAP-based pattern generation algorithm. Notably, figure 14a has a UV rotation of 0, whereas figure 14b has a UV rotation of $2\pi/8$.

I believe that my experimental results demonstrate that the metric minimizing quality of the ARAP parameterization is not sufficient to reduce the problem of stitch graph generation to that of regularly sampling a 2D UV domain. Firstly, the nontrivial metric distortion of about 194.33 is very high for this use case. Secondly, in Figure 14b, one can see that the number of stitches in each row is constant; there are no increases or decreases. To crochet a bump, one needs to use both increases and decreases for the shaping. The generated pattern corresponds to a flat rectangular single crochet patch.

3.2 Igarashi et al.'s Algorithm for Pattern Generation

Hence, I proceeded to implement a state-of-the-art algorithm. I chose to implement the algorithm that Igarashi et al. outline in their paper "Knitting a 3D Model" [3]. Doing the literature review, I learned that the algorithm outlined in their paper is the "simplest" of the inverse design for knitting / crochet algorithms, and that the majority of the modern algorithms build on top of its key concepts. Hence, I decided that spending some time implementing that algorithm would be a good first step of digging into the computational aspect of the inverse design problem for crochet. I will first give a big picture overview of the algorithm and then delineate the specifics of my implementation.

For the input of a triangle mesh representing a topological disk, Igarashi et al.'s algorithm consists of three major steps:

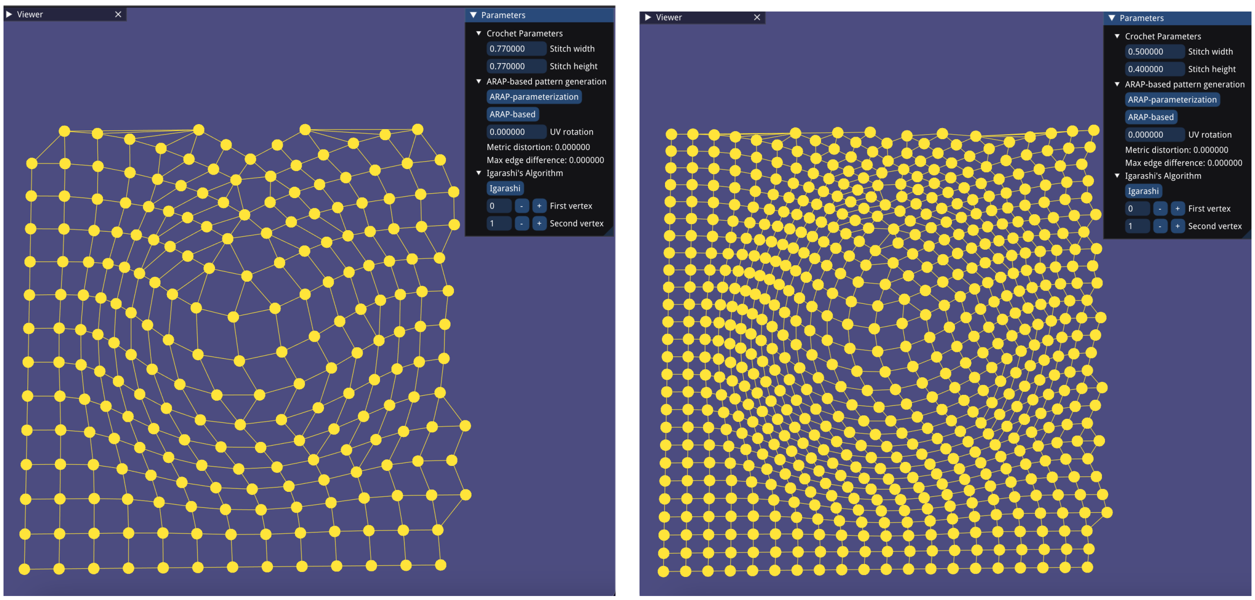
1. **Wrapping**: The user specifies the first row. The algorithm specifically takes the boundary curve of the topological disk as the first crochet row (but their method generalizes to any arbitrary starting curve on the input mesh). The method iteratively computes the next stitch-height-Euclidean-distance level set from the current set of crochet rows, which is a contour that represents the next crochet row. Hence, the wrapping step takes as input a curve representing the first crochet row, and produces the contours representing all the crochet rows.
2. **Resampling**: Each contour in the set of crochet rows is regularly resampled so that each edge has a length equal to the user specified stitch width. Hence, the resampling step takes the contours representing all the crochet rows and resamples them so that each edge of each contour represents the width of a single crochet stitch.
3. **Meshing**: The meshing step takes the set of resampled contours representing crochet rows and determines the inter-row connectivity to add edges which represent the height of a single crochet stitch. To do this, for each pair of subsequent contours, the algorithm computes the closest vertex from the first row to the second (producing a forward edge), as well as from the second row to the first (producing a backward edge). The union of the set of forward and backward edges are taken as the crochet stitch graph's column edges. Hence, the meshing step takes the set of resampled contours and determines where to add column edges to produce a valid crochet stitch mesh.

I implemented the a simplified version of Igarashi's algorithm in the same C++ / libigl project as I implemented the ARAP-based method. I chose to make a handful of simplifications to limit the time I spent on the implementation for the scope of the semester thesis. These include the following:

- Firstly, I simplified the specification of the first contour. The user can provide an index of a first and second vertex, and those two vertices are connected to form an edge, which represents the first contour. In this fashion, the user can visualize the vertex indices, and choose an initial pair to represent the first contour.

- I simplified the wrapping step so that it only works for contours that are topological segments and not closed loops. Furthermore, the wrapping terminates the moment it detects a contour with more than one connected component.
- I could not find a "quick" way to analytically perform the regular resampling of the crochet row contours, so I performed a brute-force approximation.

Just like before, the user first specifies a stitch width and a stitch height. Then, the user further specifies the first and second vertex indices, which are used to initialize the first crochet row / contour. Lastly, the user can click the "Igarashi" button to run the algorithm and visualize the generated stitch graph.



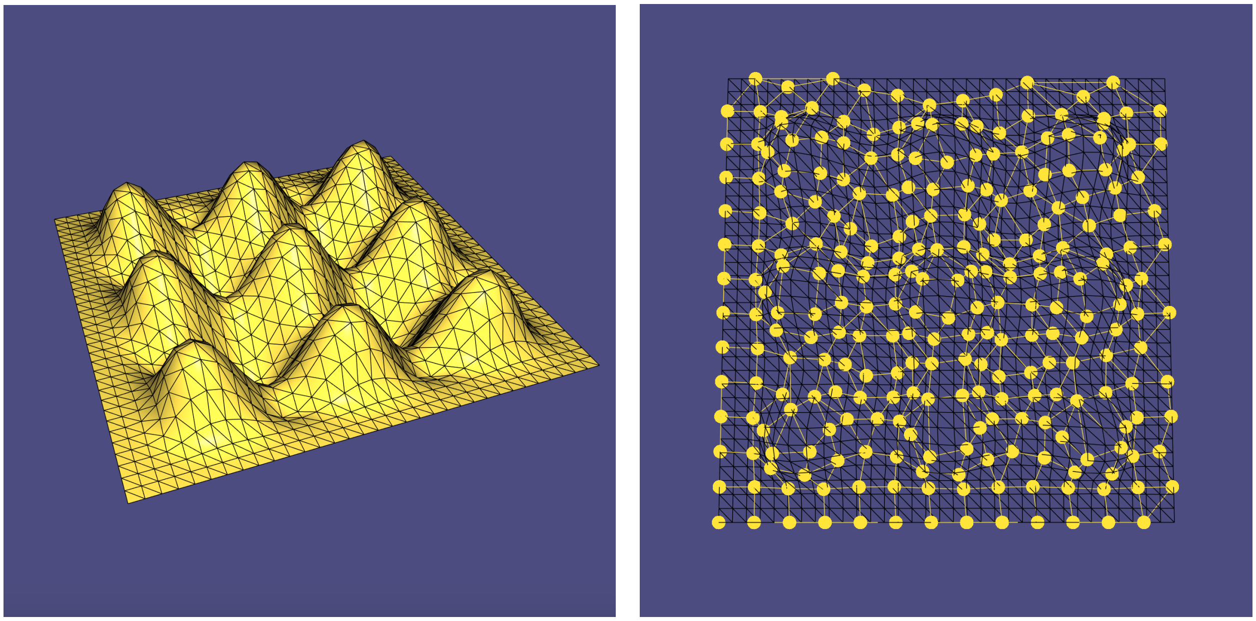
(a) bump.obj's stitch graph with a single crochet stitch width and height of 0.77cm.

(b) bump.obj's stitch graph with a single crochet stitch width of 0.5cm and height of 0.4cm.

Figure 15: Two stitch graphs for bump.obj produced with Igarashi et al.'s method.

Here are some key results demonstrating my implementation. Figure 15 illustrates two stitch graphs produced with Igarashi et al.'s algorithm; Figure 15a uses a stitch width and height of 0.77cm, and figure 15b uses a stitch width of 0.5cm and height of 0.4cm. Compared to the ARAP-based method, one can see how the rows increase and decrease in order to model the bump. By construction, we have that the row edges have the exact same length as the specified stitch width, and the hope is that the column edges are close to the specified stitch height.

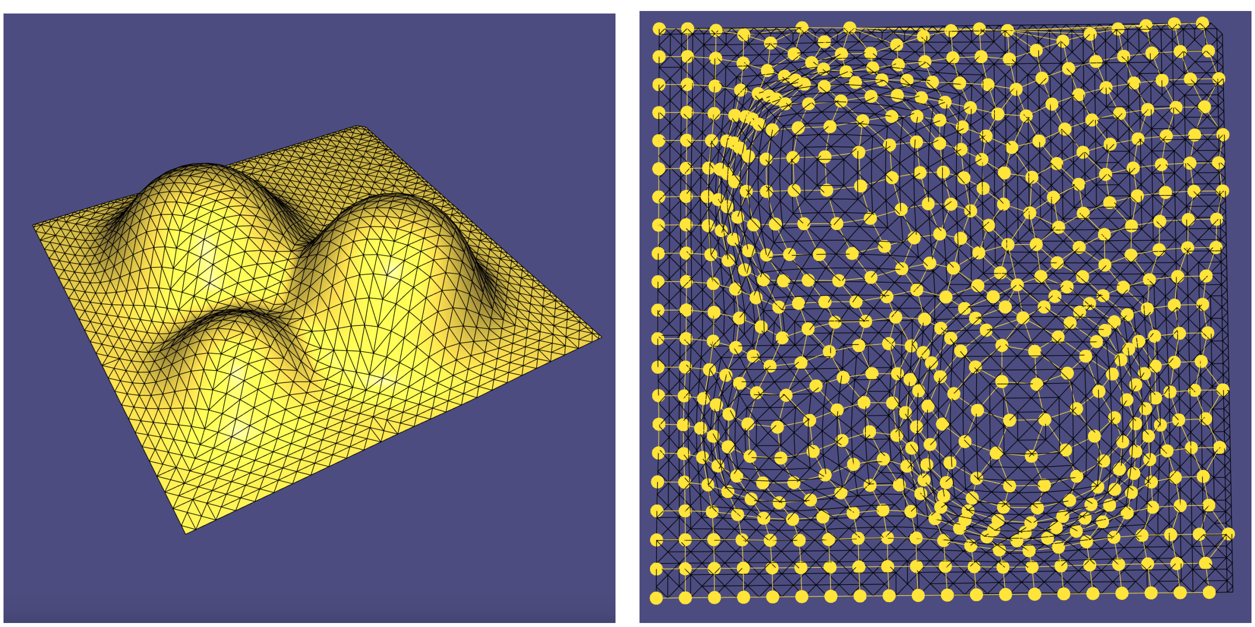
Figures 16 and 17 illustrate the stitch graphs produced by Igarashi's algorithm using bumpy_plane_low.obj and three_bumps.obj as input, respectively. Notably, in Figure 17b, one can see that there are non-boundary rows which contain decreases of three stitches at a time. While higher order decreases are possible, they are less common in the pattern literature, and can be physically hard to execute. When I was showing Professor Ren some of the algorithm's results, she pointed out that there are case surfaces where this algorithm fails, in the sense that, an undesirable quantity of high order decreases might be produced. More common in knitting than in crochet, short rows can be used to avoid this exact problem. The method presented in "Automatic Machine Knitting of 3D Meshes" [4] explicitly modifies their time function to produce short rows, potentially circumventing this issue. On the other hand, the method presented in "AmiGo: Computational Design of Amigurumi Crochet Patterns" [2], which also uses a modified time function, explicitly avoids short rows so that the crochet object can be constructed in a join-as-you-go manner. I am curious how both of these methods would handle a more dramatized version of the three_bumps.obj input, and whether there is further work to understand if and how surfaces can be crocheted.



(a) bumpy_plane_low.obj

(b) Corresponding stitch graph with a single crochet stitch width and height of 0.77cm.

Figure 16: bumpy_plane_low.obj and its corresponding Igarashi's stitch graph.



(a) three_bumps.obj

(b) Corresponding stitch graph with a single crochet stitch width and height of 1cm.

Figure 17: three_bumps.obj and its corresponding Igarashi's stitch graph.

4 Third Crochet Experiments

In the last set of crochet experiments, I

1. crocheted an example illustrating Professor Ren's idea for an algorithm that greedily places bumps to promote stability where they are needed most, and
2. crocheted the bump motif on a cylinder, to further investigate whether the bump-based metamaterial is more stable than its flat counterpart.

4.1 Catenoid Metamaterial Variation

Professor Ren suggested that instead of covering an entire surface with bumps to promote stability, we could instead come up with criteria to "greedily place them." We were not sure how to define the criteria, so we discussed how previous papers predicted the shape of the knitted / crocheted result. Notably, the AmiGo paper uses ShapeUp [1] to predict the resulting crocheted shape; however, the paper relies on the resulting crocheted object being stuffed, and only uses "purely geometric conditions." These geometric conditions are: constraining the crochet graph embedded edge lengths, constraining the position of the seed point (the first row), and smoothness. Because these conditions don't take into account gravity or other dynamics, the shape prediction would not be informative about "where we need to introduce stability." We wondered if we could incorporate additional constraints, such as gravity, for example, in order to be able to define a measure of "stability."

To explore this idea, we thought it would be useful to add a row of bumps to the catenoid that I had previously crocheted. I generated a pattern for a catenoid slightly bigger than the small catenoids, and I identified two not-too-far-away interior rows that have the same number of stitches. Then, I replaced the instructions for the sequence of interior rows with that of a row of bumps.

When we had first designed the crochet experiment to explore this idea, we thought it would be useful to have a bump variation where the first and last stitch of each row of the bump are "pinched together" with the working yarn. After starting to crochet the generated pattern, I could not work out how to do this consistently, and I proceeded with the regular bump motif.

(a) Top view

(b) Side view

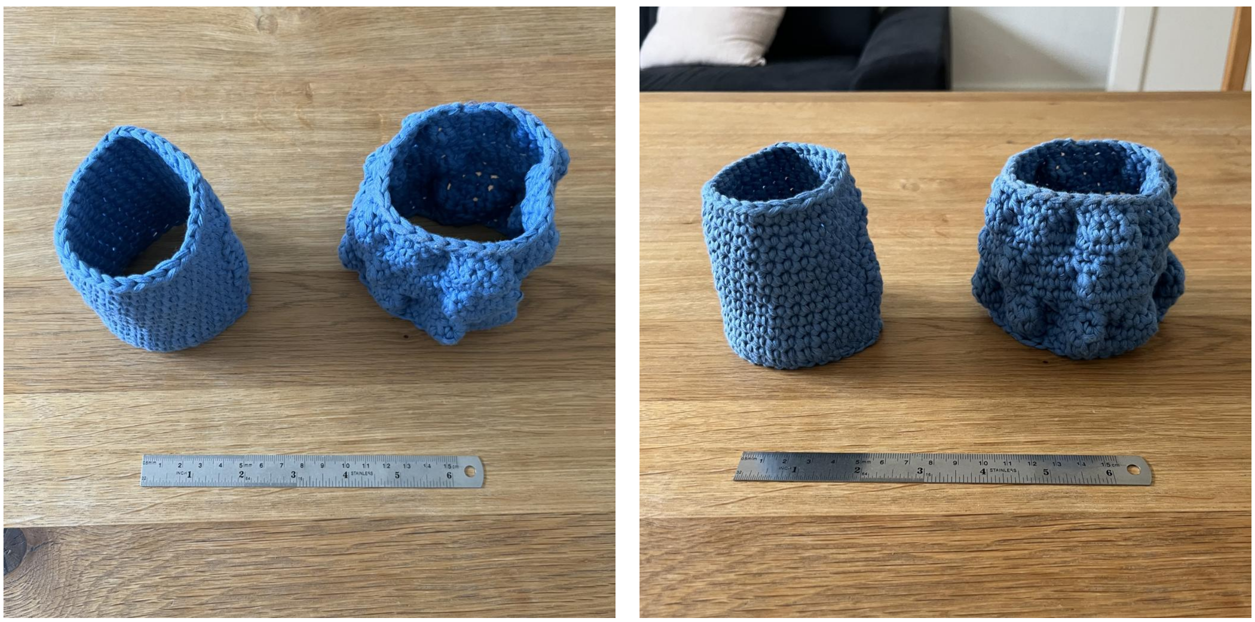
Figure 18: Catenoid Metamaterial Variation

You can see that the crocheted result looks like a slightly larger catenoid where the middle section is a row of bumps. Qualitatively, the middle section of the bumps feels a bit more stable than the rest of the catenoid. However, the first couple of first and last rows are the least stable, and perhaps would have benifited more from the bumps.

4.2 Metamaterial C

I lastly wanted to further investigate whether the bump-based metamaterial is more stable than its flat counterpart by crocheting the bump motif on a cylinder. My hope was that by crocheting a cylinder, it would be easier to measure the resistance to deformation, compared to the rectangular patches I crocheted for Metamaterial A and B. So I crocheted both a control cylinder and a cylinder comprised of 3 rows each with 8 bumps. The pattern for the bump motif is given as follows. Starting with 4 contiguous stitches,

1. sc, 2 inc, sc (6).
2. sc, 4 inc, sc (10).
3. sc, 4 dec, sc (6).
4. sc, 2 dec, sc (4).

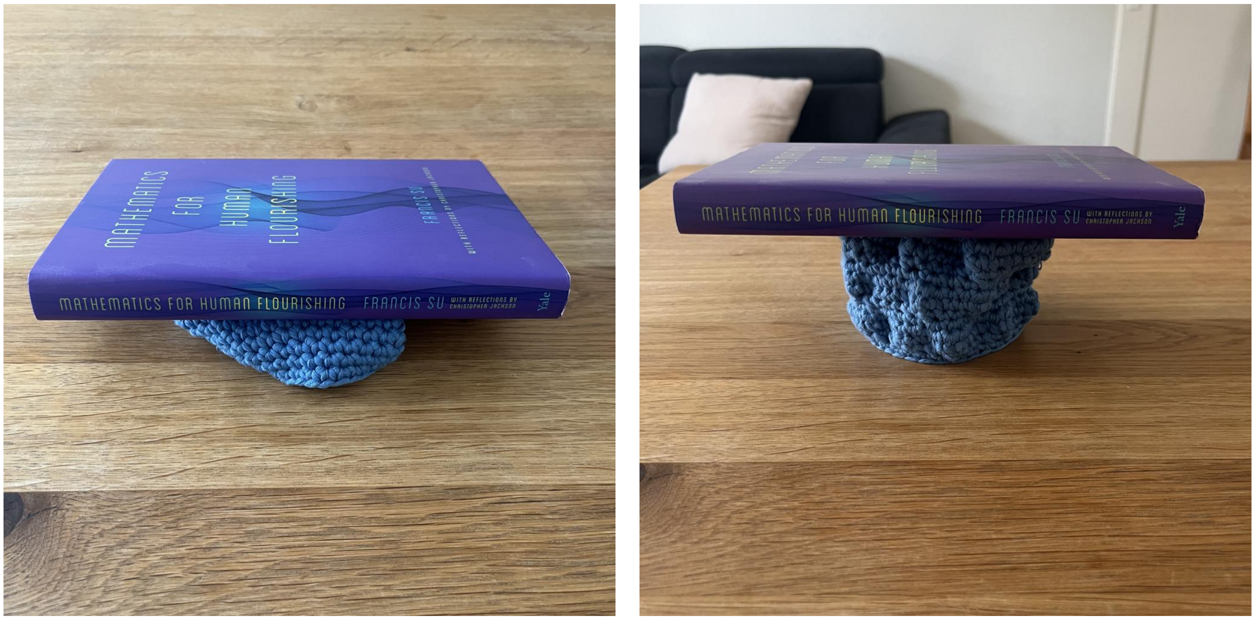


(a) Top view (b) Side view

Figure 19: Control (left) and Metamaterial C (right)

Figure 19 shows the control cylinder on the left and Metamaterial C on the right. I experimented with placing different flat objects on top of the two crocheted objects in order to measure their resistance to deformation. I found that for certain objects, both crocheted items were either stable, both items collapsed, or metamaterial c was stable whereas the control collapsed. Figure 20 illustrates that for my copy of the book "Mathematics of Human Flourishing", the control cylinder collapses, whereas metamaterial c buckles, but ultimately stably supports the added weight. I argue that Metamaterial C supports the hypothesis that in the more complicated curved surface case, using a bump motif to tile a surface, does in fact result in a more rigid and stable crocheted result.

Professor Ren's idea to find a way to greedily place the bumps to promote stability where they are needed most is relevant, particularly because tiling a surface with bumps significantly increases the number of stitches. For example, the cylinder control pattern consists of 448 stitches, whereas the Metamaterial C consists of 656 stitches, which is just about 1.5 times more stitches than the control pattern. I would argue that a pattern's expected crochet time is linear in its number of stitches, and so tiling the entirety of a surface with bumps can significantly increase the construction time.



(a) Control compliance test.

(b) Metamaterial C compliance test.

Figure 20: Control and Metamaterial C compliance tests.

5 Conclusion

Over the course of this semester project, I explored the topic of the computational design of stable crochet. I did a literature review and learned about how the inverse design problem for knitting and crochet, and related methods in fabrication, are addressed. I performed a series of experiments to explore how crochet parameters and shape parameters relate to a resulting crocheted object's stability, and learned that at smaller scales higher Gaussian curvature seems to have an impact on stability, but at larger scales, the crocheted surfaces become less stable. I then pivoted to explore the idea of a metamaterial approach that would attempt to get the best of both worlds of high stability at a local scale and loss of stability at a larger scale. Experimentally, I found that tiling a surface with a bump motif does promote the stability of the resulting crocheted object. At the same time, introducing the bumps can be expensive for the crocheter, and there are many further parameters of bump-motif crochet metamaterials to explore. I lastly explored the computational aspect of the inverse design problem for crochet by implementing and visualizing two methods that produce stitch graphs for topological disks.

References

[1] Sofien Bouaziz, Mario Deuss, Yuliy Schwartzburg, Thibaut Weise, and Mark Pauly. Shape-up: Shaping discrete geometry with projections. *Computer Graphics Forum*, 31(5):1657–1667, 2012.

[2] Michal Edelstein, Hila Peleg, Shachar Itzhaky, and Mirela Ben-Chen. Amigo: Computational design of amigurumi crochet patterns. In *Proceedings of the 7th Annual ACM Symposium on Computational Fabrication*, SCF '22, New York, NY, USA, 2022. Association for Computing Machinery.

[3] Yuki Igarashi, Takeo Igarashi, and Hiromasa Suzuki. Knitting a 3d model. *Computer Graphics Forum*, 27(7):1737–1743, 2008.

[4] Vidya Narayanan, Lea Albaugh, Jessica Hodgins, Stelian Coros, and James Mccann. Automatic machine knitting of 3d meshes. *ACM Trans. Graph.*, 37(3), aug 2018.

[5] Madlaina Signer, Alexandra Ion, and Olga Sorkine-Hornung. Developable metamaterials: Mass-fabricable metamaterials by laser-cutting elastic structures. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, CHI '21, New York, NY, USA, 2021. Association for Computing Machinery.

Figure 21: Semester project oveview.