

Crocheting Knots with Curved Surfaces

Charles Zinn

Jersey City, New Jersey, USA; charles.zinn@gmail.com

Abstract

I created a program that generates crochet patterns from a 3D computer model of a knotted torus. The patterns use increases and decreases to induce curves at the appropriate points so that the crocheted surface of the knot does not bunch up, even when tightly curving around another strand of the knot. Using the program, I generated patterns for, and subsequently crocheted, the first 14 prime knots and the unknot, which form an aesthetically pleasing collection.



Figure 1: *The first 14 prime knots along with the unknot. From top to bottom, left to right, the knots are ordered first by crossing number, then by increasing hyperbolic volume (with torus knots treated as having a hyperbolic volume of zero).*

Introduction

I had the idea for this project while doing some recreational exploration of knot theory. After staring at so many knots on a screen, I wanted to hold one in my hands, and I decided that yarn would be a good medium for creating some physical knots.

For most knitting and crochet patterns for knots and knotted pillows, a tube is created and then tied into a knot once it reaches the required length. As an example, see [6]. This is certainly an easier way to create a knot than the method presented here. However, where such a knot bends, the material bunches up on the inside of the bend (or is stretched on the outside), and this is especially apparent when the bends are tight. I wanted to create knots where the surface itself had the appropriate curvature to match the bends of the knot, the same way that socks have a heel to match the shape of feet.

To achieve this, I switched the direction of the rows in the pattern: instead of having each row go around a meridian of the knot (the short way around the tube), I made each row follow a longitude, going around the whole knot lengthwise. This works because the length of each meridian of the torus (looped tube) is a constant (corresponding to a constant number of rows), whereas the length of the longitudes vary as the torus bends. By having rows follow longitudes, I was able to use increases and decreases to change the relative length of rows, inducing the curvature.

In this paper I explain how I generated the crochet patterns using this insight and how I proceeded to crochet the knots to produce the collection shown in Figure 1, which contains all prime knots with seven or fewer crossings.

The patterns use only basic crochet stitches: chain, single crochet, increases, and decreases. I use American crochet terminology.

Generating patterns

I generated 3D models of the knots I wished to crochet with the program KnotPlot [5], which performs a relaxation simulation to find a more compact model starting from an initial mostly-2D embedding of a given knot. I imported the model into Blender [1] and wrote a script to export a grid of 3D coordinates from the surface of the knotted torus, where rows of the grid correspond to longitudes of the torus and columns correspond to meridians.

The grid serves as a projection of the surface of the torus into a flat plane. Because I wanted to work in the round, I drew a spiral onto the plane to represent the path I wanted the rows of stitches to take, so the end of each row joins to the start of the next one. (More details in Figure 4.) Then, I evenly placed stitches along the rows, measuring distance by projecting points from the plane back into points on the surface of the torus by interpolating from the nearest grid coordinates. Finally, I visualized the results by importing the stitch positions back into Blender (Figure 2).

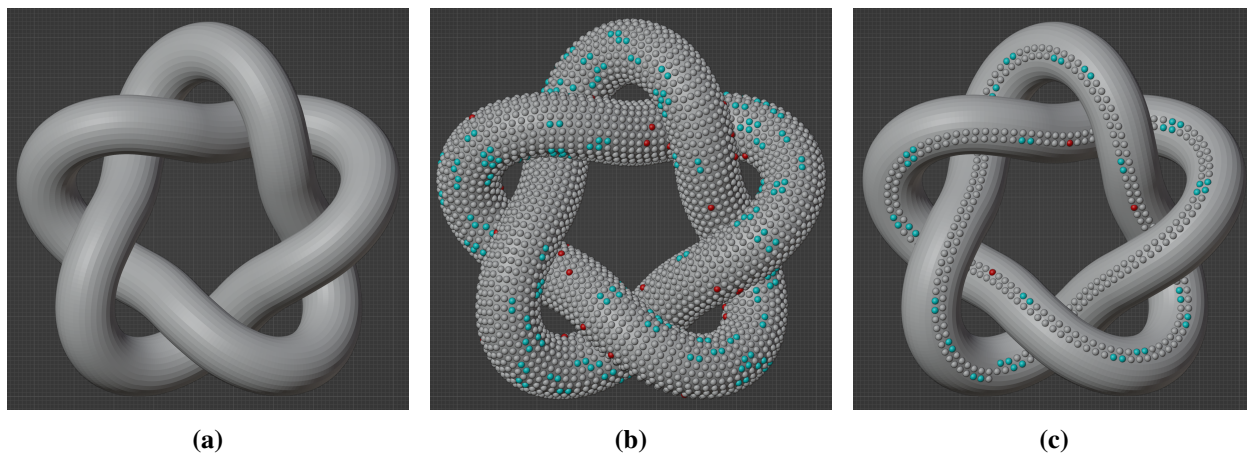


Figure 2: Model of the $(5, 2)$ -torus knot in Blender, (a) with no stitches, (b) all stitches laid out on the surface, and (c) just two rows. Stitches in cyan are increases, and the stitches in red are decreases.

To determine how far apart each stitch should be, I measured a meridian of the torus, divided it by the number of rows I wanted (which determines the overall size of the model), and multiplied it by a gauge factor which I empirically measured to be about 1.25; this is the ratio of the width of a single crochet stitch to its height.

At a point where the torus bends, rows on the outside of the curve will have more stitches than rows on

the inside, and so increases and decreases must be used to transition from shorter rows to longer ones and longer rows to shorter ones respectively. To determine where the increases and decreases should occur in each row, I used the rule that each stitch should connect to its nearest neighbour (measuring along the surface of the torus) in the previous row and the following row. Although a stitch always has only a single nearest neighbour in an adjacent row, sometimes it is the nearest neighbour of multiple stitches in that adjacent row, which corresponds to an increase or decrease. Figure 3 shows how this works for stitches laid out on a 2D ribbon. I believe this procedure results in the same connection graph as the optimization procedure mentioned in [3].

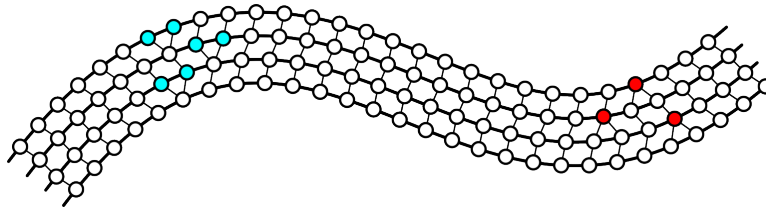


Figure 3: Stitches laid out evenly along a curving ribbon, showing how each stitch is joined to the nearest stitch in the adjacent rows. Increases are colored cyan and decreases are red, as in Figure 2.

Another way to think about this procedure is as follows: at each stitch in a given row, compute the scalar quantity “how much longer is this row than the previous row at this point along the row”, round it to the nearest stitch, and add an increase or decrease when this rounds up or down to a new whole number.

Sometimes this rounding results in very short distances between an increase and the next decrease or vice versa, where the unrounded scalar value just barely crosses the rounding threshold and then falls back below the threshold as the torus curves slightly differently. To simplify the pattern, I eliminated “noisy” increases and decreases that occurred too closely together. I also simplified some longer sequences. For example, where my algorithm recommended `[inc, 2 sc, dec, 2 sc, inc]` to turn 8 stitches from the previous row into 9 stitches in the current row, I changed it to `[3 sc, inc, 4 sc]`. This preserves the number of stitches and the net number of increases while removing the noise, and places the single remaining increase at the center of the prior cluster.

Dealing with twist

In the original model exported by KnotPlot, if you follow a longitude around the whole knot, you will end up back at the same longitude. This is a necessary condition for the generated rows of crochet to line up smoothly (though since I was working in the round, each row actually connects to the next row). If you were to cut along a meridian, twist one end 180 degrees, and reattach it, this would no longer be the case. However, it is possible to add any integer number of twists to the longitudes and still have them align correctly. These are referred to as different framings of the same knot, and the topic is discussed further in the context of knitting knots in [4]. I implemented the choice of framing in my program by changing how the spiral is drawn based on an optional `twist` parameter (Figure 4).

When generating my patterns, I usually chose the `twist` value that gave the fewest number of total stitches. However, in some cases it improved the pattern to add or remove an additional twist. Since it is possible to start the pattern at any row, I always started it on the shortest row to minimize the length of the seam. Adding or removing a twist can make that seam shorter, which also generally served to hide the seam on the inside of the model. As a concrete example, consider the (5, 2)-torus knot. With no twist modification from the base model exported by KnotPlot, each row is the same length and travels from the inside of the torus to the outside and back as it goes around the knot. With a twist applied, a row on the outside stays on the outside and a row on the inside stays on the inside, and the seam is much less visible (Figure 5). Importing the stitches into Blender with different values of twist helped me evaluate which was best for a given knot.

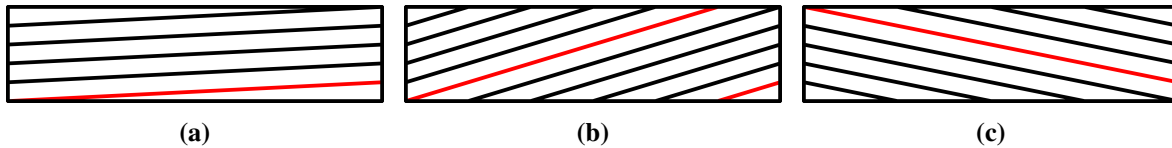


Figure 4: *Spiral rows drawn on the flattened projection of a torus. In all cases, following one row around the torus will end up at the start of the subsequent row, like a spiral going around a cylinder. (a) shows the default with no extra twist, (b) has an additional positive twist, and (c) has a negative twist. The angle of the rows looks exaggerated here, but in the actual knots, the longitudes are many times longer than the meridians, so the twist is not locally noticeable.*

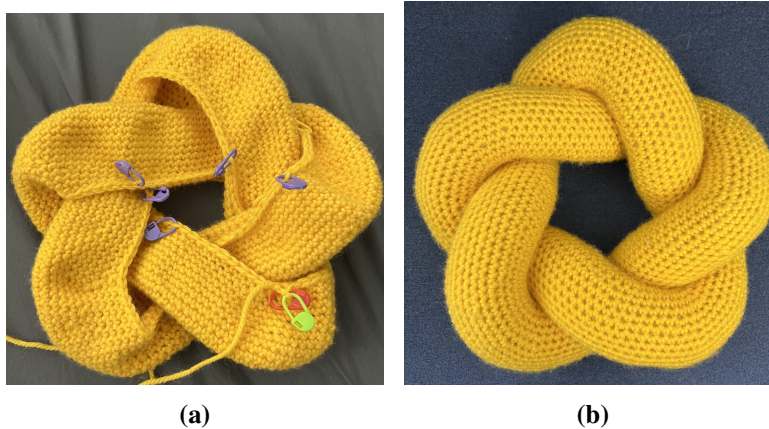


Figure 5: *(5, 2)-torus knot, (a) shown before stuffing, with the open seam running along the inside of the knot, and (b) completed, with the seam invisible because it always runs along the inside instead of precessing around the knot.*

Crochet stitches are not stacked perfectly vertically

Unlike in knitting where each stitch lies directly above the corresponding stitch in the previous row, crochet stitches lie slightly between the stitch in the previous row which they are stitched into and the preceding stitch. I empirically measured this effect to be about 0.25: each stitch lies a quarter of stitch behind the corresponding stitch in the previous row. I adjusted for this by adding an offset to the “closest stitch” calculations when determining which stitch to connect to in the next and previous rows.

Crocheting the knots

The generated patterns for these knots are fairly simple overall, in that they use only basic crochet stitches. However, there are several tricky bits involved in actually making them.

Joining the first row

When the first row is complete, the last stitch must be joined to the first to form a loop so that crocheting in the round can begin. For the knots, it is necessary that this first row be tied into the appropriate knot at this step, since it will be impossible to do later.

Additionally, the first row must have the correct number of twists. This is true for any knitting or crochet pattern; if you have a twist in your first row, you will be making a doubly-twisted Möbius strip, and that will be difficult to wear as a sweater. However, with the knots, especially with the more complicated ones, it is hard to tell at a glance whether the number of twists is correct. Unlike a sweater, there is not one correct framing

for a given knot, since as discussed in the section on dealing with twist, multiple framings are possible. The important thing is to match the framing selected in the pattern generation.

To make joining the first row easier, I wrote a script to project a pair of rows from the model to a plane, printed it out, and use a blocking mat to pin the first row to the pattern (see Figure 6a). This prevented most mistakes, though I still had to frog¹ an entire knot because I had added one extra twist by mistake.

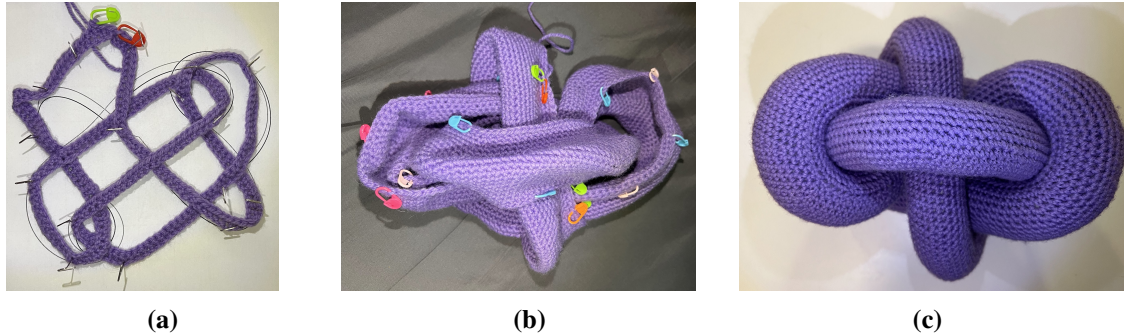


Figure 6: Various stages of the final 7-crossing knot (7ah_6 in Regina knot database [2]): (a) the first row tied into a knot based on the printed projection of the model, (b) with stitching finished and with the correct alignment but unstuffed, and (c) complete.

Following the pattern

In most knitting and crochet patterns, each row is fairly similar to the previous row. If there are repeating sections they will line up, and the rows themselves usually repeat after a small number of variations. With these knot patterns, this is not the case: every row is different, the number of stitches between each increase and decrease often feels like a random number, and there is no easy reference in the previous row for where an increase or decrease in the current row should go. This makes it very easy to lose count in the middle of a row. To help with this, I used stitch markers to mark where the increases and decreases would go before starting each row (visible in Figures 5a and 6b). This had the added effect of double checking that the previous row had the correct number of stitches, making it (relatively) easy to fix mistakes before continuing.

The other difficulty stems from the fact that the work is tied in a knot. Twice per row (sometimes three or more times as I neared the end of the patterns), I needed to pull the knot through itself so that I had an unknotted section to continue to work on. Sometimes the free end of yarn got tangled during such operations, but for the most part this was straightforward.

Aligning the knot to the model

Once all the stitching was done, I completed each knot by sewing the last row to the first, stuffing it as I went. However, before I could begin sewing and stuffing, I had to make sure that the physical knot was lined up to the model, so that the curvature of the knot's surface matched up with the way the knot was actually bending. This cannot be done later, as the knot becomes more difficult and eventually impossible to manipulate into a different shape as it is stuffed and the slack is taken up.

For smaller knots, aligning them was fairly trivial, as was the case for knots with a high degree of symmetry such as the (7, 2)-torus knot. For some of the 7-crossing knots, I had to sketch a 2D drawing of the crocheted knot and compare it with the projection of the model I used for the first row and look for similarities, because it was harder to spot the correspondences between the 3D knots. For the trickiest knot (the dark blue one in the bottom row in Figure 1, which is 7ah_5 in the Regina knot database), I wrote a

¹Pulling (ripping) out yarn to undo stitches is called “frogging” because you “Rip it. Rip it.”

script that used Regina to perform a breadth-first search between the initial diagram and the final diagram. The shortest sequence it found consisted of 13 Reidemeister moves, which I was able to carefully execute on the physical knot. Who knew that knots could get so tangled?

When I did manage to successfully align the knot with the model, there were tangible signs that I had done it correctly: the surface was no longer twisted, and there was much more slack in the knot because the curvature was working with the bends instead of against them. Even before sewing and stuffing, the knot holds its shape fairly well (Figure 6b), though it doesn't take on its final shape until it is stuffed (Figure 6c).

Summary and Conclusions

In this paper, I presented a novel method of crocheting knots such that the curvature of their surface matches the bends in the knot, and used this method to create a collection of the first 14 prime knots. This project took me almost two years to complete, and I'm immensely satisfied with the results. The code I wrote is published on GitHub [7], and I look forward to see what others make of it.

I believe there is much future work that can be done in the space of generating knitting and crochet patterns from computer models. The technique I used to determine where increases and decreases should go on a curved surface could be adapted to create patterns for other shapes, as long as a continuous path can be drawn along the surface of the model so that the rows are of approximately equal heights. It may also be possible to compensate for paths where the distance between rows varies by using crochet stitches of different heights (such half double or double crochet stitches). In the context of knots, using stitches of different heights would make it possible to have rows go along meridians around the tube (instead of lengthwise along the longitudes as in this paper) and still achieve the desired curvature.

Acknowledgements

I'm grateful to my mother, Laura Weeks, for reviewing early drafts of this paper and helping me learn to crochet in the first place, and Dr. Laura Taalman for encouraging me to write and submit this paper.

References

- [1] Blender Development Team. *Blender (Version 3.4.1)*. 2022. <https://www.blender.org>
- [2] Benjamin A. Burton, Ryan Budney, William Pettersson, et al. *Regina: Software for low-dimensional topology*. 1999–2023. <http://regina-normal.github.io/>
- [3] Mirela Ben Chen and Michal Edelstein. “Amigurumi Crochet Patterns from Geodesic Distances.” *Bridges Conference Proceedings*, Richmond, Virginia, Aug. 1–5, 2024, pp. 369–372. <https://archive.bridgesmathart.org/2024/bridges2024-369.html>
- [4] Nadav Drukker, Elise Paznokas, and Dominik Schrimpel. “Knitting Knots & the Framing Anomaly.” *Bridges Conference Proceedings*, Helsinki and Espoo, Finland, Aug. 1–5, 2022, pp. 245–252. <https://archive.bridgesmathart.org/2022/bridges2022-245.html>
- [5] Robert Sharein. *KnotPlot (Version 1.0)*. 2022. <https://knotplot.com/>
- [6] Laura Taalman. “Knit Knots: Large-Scale Soft Conformations of Minimum-Ropelength Knots and Links.” *Bridges Conference Proceedings*, Halifax, Canada, Jul. 27–31, 2023, pp. 369–372. <https://archive.bridgesmathart.org/2023/bridges2023-369.html>
- [7] Charles Zinn. *Scripts for generating crochet patterns for knots*. 2025. <https://github.com/czinn/knot-crochet>