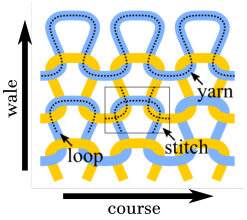
Supplementary Material for Helix-Free Stripes for Knit Graph Design

1 KNITTING PRIMER

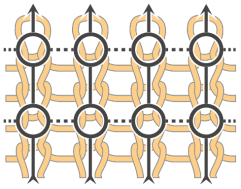
1.1 Knit Pattern



In (weft) knitting, a fabric is formed from rows of yarn loops that are interwoven to form grids of stitches. The simplest (single jersey) pattern is visualized in the figure to the right, from [5]. Rows of these basic stitches form *courses*, while columns form *wales*.

A regular grid of such stitches has an inherently flat geometry, so to introduce curvature, one must insert *stitch irregularities* in the form of *short rows* and *increases* and *decreases*. These are illustrated at the stitch level in Fig. 1. Short rows introduce additional course rows, while increases/decreases change the number of wale columns in the stitch pattern. By placing such stitch irregularities in an informed fashion, one can reproduce a target geometry.

1.2 Knit Graphs



In this work, we follow the paradigm established in AutoKnit [4] and aim to construct a *knit graph* to represent the stitch pattern. Each node in a knit graph represents a pair of stacked stitches, neighboring each other in the wale direction, as illustrated in the inset figure from [4]. Pairs of stitches are used instead of single stitches to accommodate the tracing of short rows, which actually consist of two rows of stitches.

For areas without stitch irregularities, each node is part of four directed edges. Two are course edges, one incoming and one outgoing, and describe the overall direction in which yarn is laid down by the knitting machine. The other two are wale edges, one incoming and one outgoing, and describe the direction in which course rows are being produced.

To represent stitch irregularities, one varies the number of incoming and outgoing course and wale edges. Short row ends are represented by nodes lacking either an incoming or outgoing course edge. Increases/decreases are represented by nodes having two outgoing/incoming wale edges, respectively. See Fig. 1, right.

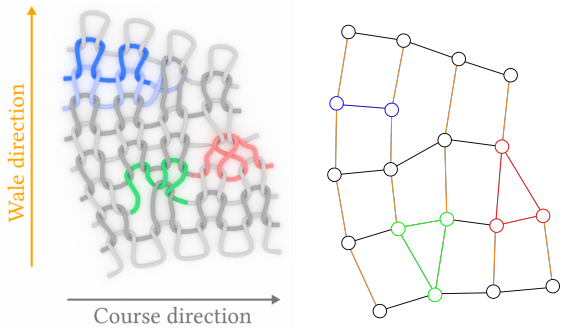


Fig. 1. A short row end, an increase, and a decrease, and the corresponding knit graph. Modified from [2].

•

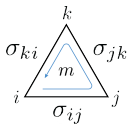
2 DISCRETE DIFFERENTIAL FORMS PRIMER

We present basic definitions from the theory of discrete exterior calculus, and give some intuitions on the objects and operators. Our discussion is brief, but more complete expositions may be found, e.g., [1].

On a triangle mesh M, discrete differential 0-forms, 1-forms, and 2-forms are assignments of real numbers to vertices V, oriented edges E, and oriented faces F, respectively. The orientations of edges, are chosen arbitrarily, and the orientations of faces are specified by the outward normal. Thus, a *0-form* $f : V \to \mathbb{R}$ is a discretized function over the mesh. Linear interpolation of the vertex values over triangles approximates the original function.

A *1-form* $\sigma : E \to \mathbb{R}$ gives a real number for each oriented edge, and may be thought of as a discretized vector field over the surface. The value assigned to each oriented edge is the path integral of the vector field along that edge. Following convention, we denote its value on an edge ij with σ_{ij}, noting that $\sigma_{ji} = -\sigma_{ij}$, to account for orientation. Lastly, a *2-form* $\omega : F \to \mathbb{R}$ assigns a number to each oriented triangular face, and may be thought of as an area measure over the mesh.

Acting on these k-forms are discrete exterior differential operators, d_0 and d_1, discrete gradient and curl operators, respectively. In particular, d_0 acts on a 0-form f to return a 1-form $d_0 f$ whose value on an oriented edge ij is given by $(d_0 f)_{ij} = f(v_j) - f(v_i)$. And d_1 acts on a 1-form σ to return a 2-form $d_1\sigma$ whose value on a face m is given by the oriented sum of σ around the boundary. The inset figure illustrates this, showing that $(d_1\sigma)_m = \sigma_{ij} + \sigma_{jk} + \sigma_{ki}$ (normal pointing out of the page).



3 STRIPE PATTERN TRACING

We first integrate both σ_c and σ_w along a spanning tree of the mesh edges, and store the values (mod P) as α_{mod} and β_{mod}, respectively, at each vertex.

3.1 On the interior of a single triangle:

σ_c is integrated locally over every triangle, $ijk \in F$, with $\alpha_i = (\alpha_{\text{mod}})_i$, $\alpha_j = \alpha_i + (\sigma_c)_{ij}$, and $\alpha_k = \alpha_j + (\sigma_c)_{jk}$. σ_w and β_i, β_j, and β_k are calculated analogously. On triangles where both σ_c and σ_w are non-singular, the stripe texture functions are linearly interpolated, so a linear system in barycentric coordinates b_i, b_j, b_k is used to find the locations of stripe intersections.

$$\alpha_i b_i + \alpha_j b_j + \alpha_k b_k = Z_c$$
$$\beta_i b_i + \beta_j b_j + \beta_k b_k = Z_w$$

where Z_c and Z_w represent the stripe level sets ($\{kP + P/4 \mid k \in \mathbb{Z}\}$) over $\triangle ijk$ for the course and wale directions, respectively. Knit graph vertices are placed at stripe intersections, and directed knit graph edges are designated based on the level set values. In particular, vertices with the same Z_w are connected with wale edges in the direction of increasing Z_c. Course edges are found analogously.

For faces where one of σ_c and σ_w are singular, we use the interpolant presented in [3]. Here P is the period and n is the triangle index. For ease of notation, we assume σ_c is singular and σ_w is not. Our strategy is symmetric if σ_w is singular and σ_c is not.

Barycentric region 1: $b_k \le b_i$ & $b_k \le b_j$

$$\alpha_i b_i + \left(\alpha_j - \frac{Pn}{3}\right) b_j + \left(\alpha_k - \frac{2pn}{3}\right) b_k + \frac{Pn}{6}\left(1 + \frac{b_j - b_i}{1 - 3b_k}\right) = Z_c$$

Barycentric region 2: $b_i \le b_j$ & $b_i \le b_k$

$$\alpha_i b_i + \left(\alpha_j - \frac{Pn}{3}\right) b_j + \left(\alpha_k - \frac{2Pn}{3}\right) b_k + \frac{Pn}{6}\left(3 + \frac{b_k - b_j}{1 - 3b_i}\right) = Z_c$$

Barycentric region 3: $b_j \leq b_i \;\&\; b_j \leq b_k$

$$\alpha_i b_i + \left(\alpha_j - \frac{Pn}{3}\right) b_j + \left(\alpha_k - \frac{2Pn}{3}\right) b_k + \frac{Pn}{6}\left(5 + \frac{b_i - b_k}{1 - 3b_j}\right) = Z_c$$

Orthogonal stripes are linear and given by: $\beta_i b_i + \beta_j b_j + \beta_k b_k = Z_w$

The above system is simply the intersection of a line with a quadratic in each barycentric region. We solve for b_i, b_j and b_k individually in each region and then use the barycentric coordinates to determine the position of the knit graph vertex. Within each region, edge connectivity is identical to the case of non-singular triangles as described above. Across barycentric regions, vertices are connected using the strategy described below.

3.2 Across adjacent triangles:

Our method implements the connection of graph vertices across adjacent triangles through the creation of "virtual" vertices on the triangle borders. In particular, in each triangle, we calculate the intersection of stripe level sets with the boundary and connect them to any interior knit graph vertices. On singular triangles, this is done separately on each barycentric region, as the underlying formulae are different on these domains. In a final step, we merge and eliminate these virtual vertices based on proximity and connect up their associated knit graph vertices.

4 EDGE LENGTH ERROR HISTOGRAMS

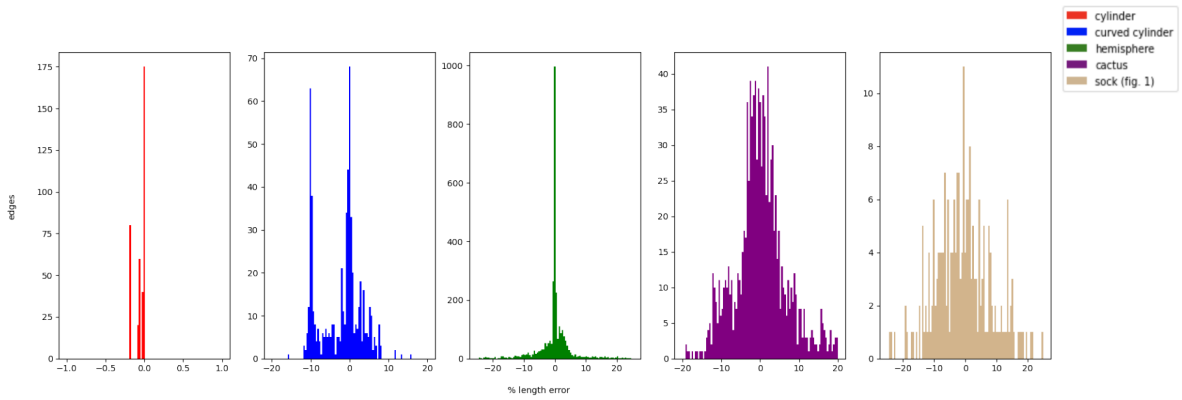


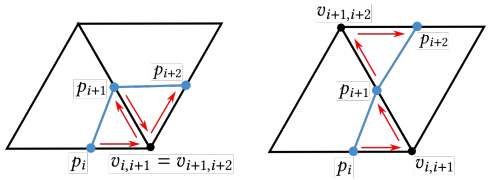
Fig. 2. We present histograms of the percent error in knit graph edges. Target edge length is defined by the period, P. As in [4], most edges have less than 10% error across models.

.

5 LEMMA 1 PROOF

5.1 Simplified form for level set constraints

Before we establish the lemma, let us note a simplified form for the level set constraint along a path γ. This is argued by observing the two cases that occur when considering adjacent segments of the polyline, illustrated in the inset figure. In both the left and right diagrams, the four red arrows denote the four terms below, respectively:



$$(-s_i^{i+1}\sigma_{e_i}r_i^{i+1} + s_{i+1}^{i}\sigma_{e_{i+1}}r_{i+1}^{i}) + (-s_{i+1}^{i+2}\sigma_{e_{i+1}}r_{i+1}^{i+2} + s_{i+2}^{i+1}\sigma_{e_{i+2}}r_{i+2}^{i+1}) \tag{1}$$

In the left case, where $v_{i,i+1} = v_{i+1,i+2}$, the middle two integrals cancel, as they are along the same segment, but in opposite directions. In the right case, when $v_{i,i+1} \neq v_{i+1,i+2}$, the middle two integrals combine to form an integral along the mesh edge from $v_{i,i+1}$ to $v_{i+1,i+2}$. In both cases, if we allow $\int_{[p,q]} \sigma$ to denote the path integral of σ along the line segment between points p and q, then Eq. (1) is equivalent to:

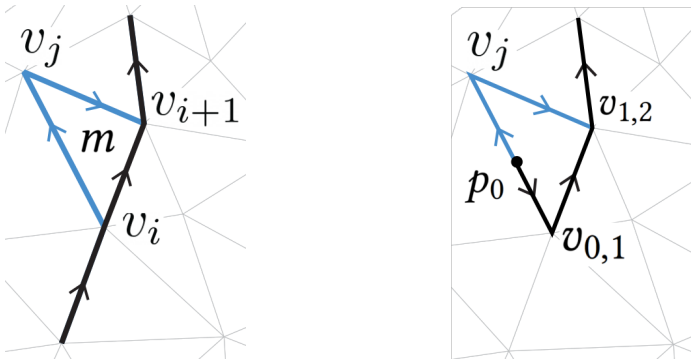
$$\int_{[p_i, v_{i,i+1}]} \sigma + \int_{[v_{i,i+1}, v_{i+1,i+2}]} \sigma + \int_{[v_{i+1,i+2}, p_{i+2}]} \sigma$$

In the left case, the middle integral vanishes, and in the right it is along a mesh edge. Applying this reasoning inductively, we see that an equivalent expression for our level set constraint is:

$$\int_\gamma \sigma = \int_{[p_0, v_{0,1}]} \sigma + \sum_{i=0}^{n-2} \left(\int_{[v_{i,i+1}, v_{i+1,i+2}]} \sigma \right) + \int_{[v_{n-1}, p_n]} \sigma$$

All of the path integral terms in the central sum are over mesh edges. Thus, we can assume WLOG that our polyline γ is of this form.

5.2 Proof of Lemma 1



(a) Midpath case (b) Endpoint case

Fig. 3

Consider a polyline γ and a homologous curve γ' obtained by the change illustrated in Fig. 3a. The curve γ' follows the blue path around triangle m visiting vertices v_i, v_j, v_{i+1}, while curve γ follows the black path around

triangle m visiting just v_i and v_{i+1}. Triangle m must be non-singular as we are on the mesh $M' = M \setminus K$, and so we have that:

$$0 = (d_1\sigma)_m = \int_{[v_i,v_{i+1}]} \sigma + \int_{[v_{i+1},v_j]} \sigma + \int_{[v_j,v_i]} \sigma$$
$$\Rightarrow \int_{[v_i,v_{i+1}]} \sigma = -\int_{[v_j,v_i]} \sigma - \int_{[v_{i+1},v_j]} \sigma = \int_{[v_i,v_j]} \sigma + \int_{[v_j,v_{i+1}]} \sigma$$

Thus, we have that $\int_\gamma \sigma = \int_{\gamma'} \sigma$. Analogous arguments can be applied to triangles near endpoints p_0 and p_n, illustrated in Fig. 3b. The integral equalities that result are:

$$\int_{[p_0,v_{0,1}]} \sigma + \int_{[v_{0,1},v_{1,2}]} \sigma = \int_{[p_0,v_j]} \sigma + \int_{[v_j,v_{1,2}]} \sigma$$

This implies that paths homologous under changes near the endpoints also have their path integral values preserved. As paths are homologous if they differ by a sequence of such changes, we see that the lemma is proven. Lastly, note that this result also holds for level set constraints applied to cycles, which we have referred to as helix elimination constraints.

6 REMARK 1 DISCUSSION

In this section, we sketch an argument that helices in a course stripe pattern may not be defined without a wale stripe pattern. Consider the instructive example below:

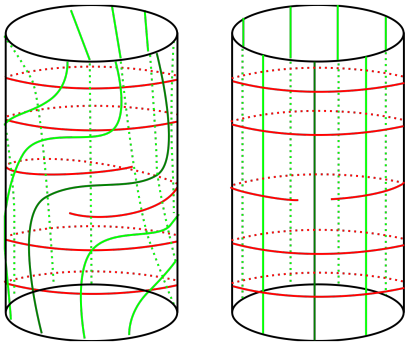


Fig. 4. Isotopic course (red) and wale (green) stripe patterns

As can be seen, the two stripe patterns are isotopic to each other, and what is seemingly a helix in the left cylinder can be avoided by a tortured wale stripe pattern. In a more general scenario, i.e., on an arbitrary surface, with an arbitrary number of potential helical stripes, we may consider disjoint topological neighborhoods of each potential helical stripe. Isotopies restricted to these neighborhoods may draw the ends of these stripes arbitrarily near each other, and any wale stripe pattern may then be imposed, which will avoid helices. The isotopies can then be reversed to obtain a distorted wale stripe pattern that will avoid the original potential helices. Thus, in finding and avoiding helices in our methods, we cannot restrict to imposing constraints or referencing information from just σ_c and the course stripe pattern.

7 REMARK 2 DISCUSSION

A helical stripe from a singular triangle a to a singular triangle b would need to follow a level set created at a and destroyed at b. We would like to argue that a helix elimination constraint along a cycle γ separating a from b locally makes it energetically very unfavorable to have such a level set. The result is a practical guarantee that our helix elimination constraints will remove helices.

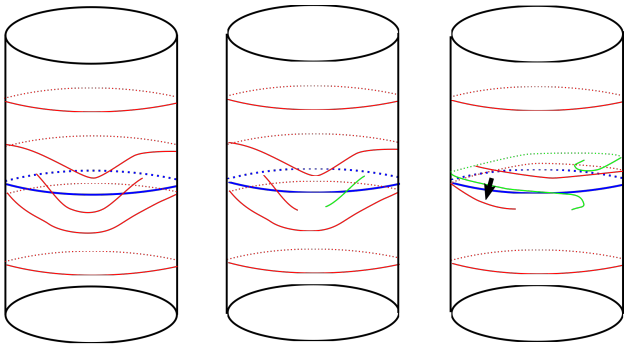


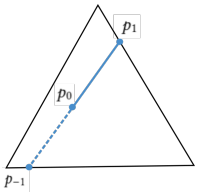
Fig. 5. Energetic unfavorability of helical course stripes (red and green) across a helix elimination constraint (blue). See text.

We refer to the figure above to guide our informal argument. When a helix constraint is imposed on a cycle γ that roughly follows the isocontours of the time function h, there are typically no level sets that begin on one side of γ and end on the other, as illustrated on the left. It is, however, possible for this to occur without violating the constraint on γ, if there is another level set going the other direction which cancels it out, as illustrated in the middle. The canceling level set is depicted in green. If there is a level set which is helical in shape, then this leaves little room for a canceling level set to go the other direction, as illustrated on the right (some red course stripes have been omitted to avoid crowding the diagram). Even if there is such a canceling level set (in green), the local direction of σ_c or the stripe texturing function (represented with black arrow) will be nearly anti-aligned with the underlying gradient of the time function, resulting in a large objective value.

As can be seen, it is difficult to make our argument watertight, because of similar ideas at play in §6. In particular, it should be impossible to make such stripe patterns infeasible with constraints imposed just on σ_c, as would be suggested by Remark 1.

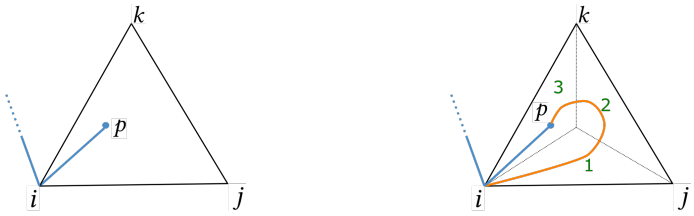
8 STRIPE ALIGNMENT CONSTRAINTS WITH INTERIOR ENDPOINTS

This case is particularly useful when applying the global optimization approach of S2, when one is motivated to work on relatively coarse meshes, for runtime purposes. Recall again, that the endpoints must be in non-singular triangles, as level sets are not linear in singular triangles. In the inset figure, we depict the scenario for the start point p_0. As the stripe texture function is linear, we can achieve a level set alignment by extending the polyline to the boundary, and applying the standard form of the constraint to the polyline with vertices $p_{-1}, p_1, p_2, \ldots, p_n$. An analogous argument may be made for the endpoint in the interior of a non-singular triangle.



9 STRIPE PLACEMENT CONSTRAINTS WITH INTERIOR ENDPOINTS

As with the stripe alignment constraints with interior endpoints, these are useful for coarse triangulations. For the subcases, the following diagram will be referenced, depicting the last segment of γ in blue, and the point in question p.



(a) Non-singular case (b) Singular case

9.1 Non-singular triangles

Case depicted in Fig. 6a. If p has barycentric coordinates b_i, b_j, b_k, then the linear nature of the function on triangle ijk allows us to see that:

$$\int_{[i,p]} \sigma = b_j \sigma_{ij} + b_k \sigma_{ik}$$

9.2 Singular triangles

When we are looking to place stripes, the particular path to the point in question does not matter, as we are just trying to specify the stripe texture function (mod P). As such, we use the orange curve, denoted $[i,p]'$, in place of the original line segment $[i,p]$, and the same effective constraint will result.

We work through the reasoning for one region, depicted in Fig. 6b, as the formulae for other regions is analogous. The regions are labelled in green, so the region 3 case is shown. The relevant expression is Eq. (**??**), which gives the behavior of the interpolant on singular triangles. Locally integrated values (starting at 0 at i) are used: $\alpha_i = 0$, $\alpha_j = \sigma_{ij}$, and $\alpha_k = \sigma_{ij} + \sigma jk$. This gives us the expression:

$$\int_{[i,p]'} \sigma = \left(\sigma_{ij} - \frac{Pn}{3}\right) b_j + \left(\sigma_{ij} + \sigma_{jk} - \frac{2Pn}{3}\right) b_k + \frac{Pn}{6}\left(5 + \frac{b_i - b_k}{1 - 3b_j}\right)$$

REFERENCES

[1] Keenan Crane, Fernando de Goes, Mathieu Desbrun, and Peter Schröder. 2013. Digital Geometry Processing with Discrete Exterior Calculus. In *ACM SIGGRAPH 2013 courses* (Anaheim, California) *(SIGGRAPH '13)*. ACM, New York, NY, USA, 126 pages.

[2] Alexandre Kaspar, Kui Wu, Yiyue Luo, Liane Makatura, and Wojciech Matusik. 2021. Knit Sketching: from Cut & Sew Patterns to Machine-Knit Garments. *ACM Transactions on Graphics (Proc. SIGGRAPH)* 40, 4 (2021).

[3] Felix Knöppel, Keenan Crane, Ulrich Pinkall, and Peter Schröder. 2015. Stripe patterns on surfaces. *ACM Transactions on Graphics* 34, 4 (July 2015), 39:1–39:11.

[4] Vidya Narayanan, Lea Albaugh, Jessica Hodgins, Stelian Coros, and James Mccann. 2018. Automatic Machine Knitting of 3D Meshes. *ACM Transactions on Graphics* 37, 3 (Aug. 2018), 35:1–35:15.

[5] Vidya Narayanan, Kui Wu, Cem Yuksel, and James McCann. 2019. Visual knitting machine programming. *ACM Transactions on Graphics* 38, 4 (July 2019), 63:1–63:13.