Supplementary Materials for "Geometric quantification for nonlinear deformation in knitted fabrics"

Jiani Fang, Xiaoxiao Ding, Gary P. T. Choi

Contents

S1 Curve quantification

Here we present the mathematical details of our curve quantification methods. Using the B-spline structure, smooth fabric curves could be reconstructed, and subsequent curve

curvature and torsion quantification steps could be performed based on the resulting model.

S1.1 Smooth curve reconstruction by B-spline

We are initially given x, y, and z coordinates of 16 nodes in each unit of fabric, from which we aim to construct a smooth curve depicting the yarn itself. Since the coordinates in three dimensions are stored separately in three block matrices of size 16×1, we perform least-squares B-spline approximation to calculate the curve in each of the three dimensions independently.

Take data in x-coordinates as an example: Consider the B-spline function parametrized by $u \in [0, 1]$. We first generate 16 uniformly distributed parameter values u, ranging from 0 to 1, and parameterize the control points to the $[0, 1]$ interval. Meanwhile, fourth-order B-spline basis functions are created, and a B-spline curve that minimizes the total squared deviation from all control points is obtained using least-squares fitting.

Mathematically, the curve is expressed as:

$$C(u) = \sum_{i=0}^{m} P_i N_{i,p}(u),$$

where P_i are the control points, $N_{i,p}(u)$ are the p-degree B-spline basis functions ($p = 4$ in our case) with

$$N_{i,0}(u) = \begin{cases} 1 & \text{if } u_i \leq u < u_{i+1}, \\ 0 & \text{otherwise,} \end{cases}$$

$$N_{i,p}(u) = \frac{u - u_i}{u_{i+p} - u_i} N_{i,p-1}(u) + \frac{u_{i+p+1} - u}{u_{i+p+1} - u_{i+1}} N_{i+1,p-1}(u),$$

and u is the parameter defined on the interval $[0, 1]$.

By minimizing the error between the curve and the data points, the approximation B-spline could be obtained. Specifically, we need to solve for control points P_i that minimize the following objective function:

$$\min_{P_0, P_1, \ldots, P_m} \sum_{k=0}^{n} \|C(u_k) - D_k\|^2,$$

where u_k are the parameterized parameter values and D_k is the k-th data point.

The above problem can be transformed into a linear system of equations, represented as:

$$\boldsymbol{AP} = \boldsymbol{D},$$

where $\boldsymbol{A}$ is the basis function matrix with elements $A_{k,i} = N_{i,p}(u_k)$, $\boldsymbol{P}$ is the control point vector $\boldsymbol{P} = [P_0, P_1, \ldots, P_m]^T$, and D is the data point vector $\boldsymbol{D} = [D_0, D_1, \ldots, D_n]^T$. The solution is obtained through the least squares method:

$$\boldsymbol{P} = (\boldsymbol{A}^T \boldsymbol{A})^{-1} \boldsymbol{A}^T \boldsymbol{D}.$$

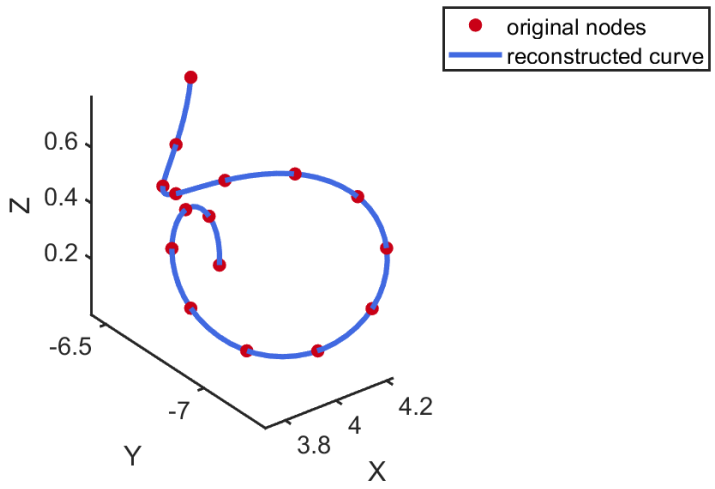

Figure S1: **Curve reconstruction using B-spline approximation.** Here, the red points indicate the original vertices, and the blue curve represents the reconstructed smooth curve.

After obtaining the B-spline structure, n equally spaced new parameter values u are generated ($u \in [0, 1]$), after which n x-coordinate values corresponding to the n sampling parameter points can be calculated using the spline structure. By adjusting n, the sampling density of the output curve could be controlled to balance speed and accuracy. The resulting data are stored in an $n \times 1$ matrix and output as the x-coordinates of the n new sampling points. The y and z coordinates are processed in the same way, and the three-dimensional coordinates of the n new sampling points could then be obtained.

Through the above process, we obtain a B-spline curve consisting of n sampling points that approximates the control points while maintaining smoothness and continuity properties (see Fig. S1 for an example).

S1.2 Curve curvature and torsion quantification

Having obtained the reconstructed boundary curve, we can analyze the local geometric structure of the unit fabric by computing its curvature and torsion, which are essential components of a complete quantification framework. Specifically, curvature measures the degree of bending of a curve at a given point, while torsion describes the extent to which a curve deviates from being planar, reflecting the "twisting" characteristic of the curve.

Mathematically, the curvature of a parametric space curve $\mathbf{r} : I \to \mathbb{R}^3$ with $\mathbf{r}(u) = (x(u), y(u), z(u))$, where I is a parameter interval, is given by:

$$\kappa(u) = \frac{\|\mathbf{r}'(u) \times \mathbf{r}''(u)\|}{\|\mathbf{r}'(u)\|^3},$$

where $\mathbf{r}'(u)$ and $\mathbf{r}''(u)$ are the first and second derivatives of the curve, respectively. The

torsion of the curve is given by

$$\tau(u) = \frac{(\mathbf{r}'(u) \times \mathbf{r}''(u)) \cdot \mathbf{r}'''(u)}{\|\mathbf{r}'(u) \times \mathbf{r}''(u)\|^2}.$$

If we further consider the arc length parameterization, i.e., a re-parameterization of the curve by its length: $\mathbf{r} : [0, L] \to \mathbb{R}^3$ where L is the total length of the curve, then we have $\|\mathbf{r}'(s)\| = 1$ for all $s \in [0, L]$ and hence

$$\kappa(s) = \|\mathbf{r}'(s) \times \mathbf{r}''(s)\| = \|\mathbf{T}'(s)\|,$$

where $\mathbf{T}(s) = \frac{\mathbf{r}'(s)}{\|\mathbf{r}'(s)\|} = \mathbf{r}'(s)$ is the tangent vector. Analogously, the torsion can be simplified as

$$\tau(s) = -\mathbf{B}'(s) \cdot \mathbf{N}(s),$$

where $\mathbf{N}(s) = \frac{\mathbf{T}'(s)}{\|\mathbf{T}'(s)\|}$ is the normal vector and $\mathbf{B}(s) = \mathbf{T}(s) \times \mathbf{N}(s)$ is the binormal vector.

In Fig. S2, we further explain the geometric meaning of the torsion. Note that the osculating plane of a curve is the plane spanned by the tangent vector $\mathbf{T}(s)$ and the normal vector $\mathbf{N}(s)$, while the binormal vector $\mathbf{B}(s)$ is a unit vector perpendicular to the osculating plane, whose direction directly determines the orientation of the osculating plane. Hence, the variation rate $\mathbf{B}'(s)$ of the binormal vector reflects the rotation of the osculating plane. Note that the formula for torsion quantifies the projection of the variation rate of the binormal vector onto the normal vector. As such, the magnitude of torsion directly reflects the rotational rate of the osculating plane about the tangent vector, serving as a metric for describing the "twisting" degree of the curve. The greater the torsion, the more pronounced the curve's deviation from planarity, and the faster the osculating plane rotates.

In practice, the curvature and torsion can be computed by substituting the vertex coordinates of the reconstructed B-spline curve into the `frenet_robust` function in MATLAB [1]. This function performs local geometric analysis of a 3D space curve using a sliding-window approach, computing the curvature and torsion at each sampling point. It estimates the tangent vector through linear regression, fits an optimal fusion plane via singular value decomposition, and applies the Taubin circle fitting algorithm to determine local curvature. Moreover, the torsion is computed as the rate of change of the binormal vector, with Bayesian prior optimization ensuring continuity between adjacent windows.

S2 Surface quantification

To capture the surface-based features of the fabric structures, we considered different approaches for surface reconstruction and the computation of the area and curvature quantities.

S2.1 Triangulation

As mentioned in the main text, a coarse triangulation in each unit stitch is performed to get a reference value for comparing with the result obtained in subsequent area computation

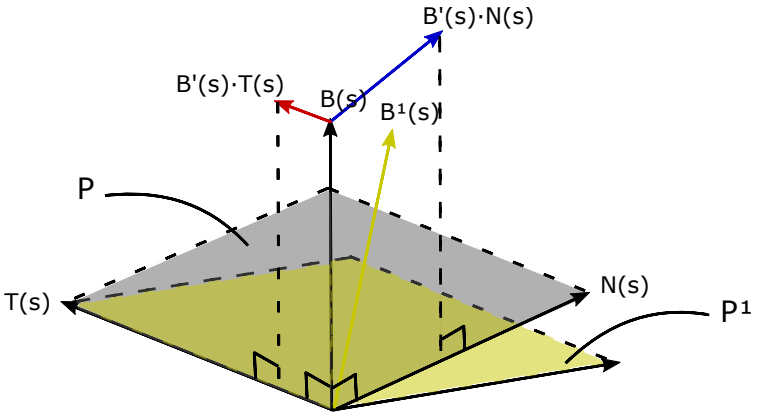

Figure S2: **Geometric meaning of the torsion of a curve.** In the figure, $\mathbf{T}(s)$, $\mathbf{N}(s)$, $\mathbf{B}(s)$ represent the tangent vector, normal vector, and binormal vector, respectively. $\mathbf{P}$ represents the osculating plane spanned by the normal vector and the tangent vector, which is perpendicular to the binormal vector. $\mathbf{B}'(s) \cdot \mathbf{N}(s)$ represents the projection of the variation rate of the binormal vector onto the normal vector. $\mathbf{B^1}(s)$ represents the variational tendency of the binormal vector in the normal direction, and $\mathbf{P^1}$ represents the rotational tendency of the osculating plane about the tangent vector. Here, we only consider the variation of the binormal vector in the direction of the normal vector, and the corresponding rotation of the osculating plane about the tangent vector, to visualize the quantity described by torsion. There is also possibly a variation of the binormal vector in the direction of the tangent vector (as shown in the figure, denoted by $\mathbf{B}'(s) \cdot \mathbf{T}(s)$), and hence the rotation of the osculating plane about the normal vector.

methods.

The specific steps of triangulation using 16 nodes are as follows. First, label the 16 nodes in order as $1, 2, 3, \ldots, 16$. Then, divide them into two groups of seven triangular patches each, with the triangular patch in the first group has vertex labels: $\{i, i+1, 17-i\}$ for $i = 1, 2, \ldots, 7$, and those in the second group has vertex labels: $\{i, i+1, 17-i\}$, for $i = 9, 10, \ldots, 15$. Using the known coordinates of the triangle vertices in each group, calculate the area of each triangular patch using the vector dot product, and summing them provides a coarse approximation of the unit area. Fig. S3 demonstrates the triangulation described above.

S2.2 Bézier surface

As mentioned in the main text, another approach for generating the unit surface is to consider the Bézier surface. First, we define four sets of three-dimensional control points $\{P_{00}, P_{01}, P_{02}, P_{03}\}$, $\{P_{10}, P_{11}, P_{12}, P_{13}\}$, $\{P_{20}, P_{21}, P_{22}, P_{23}\}$, $\{P_{30}, P_{31}, P_{32}, P_{33}\}$, each with four points. Then, a Bézier curve from each control point set is generated. Specifically, using the input coordinates of four points, a Bézier curve of degree $n = 3$ can be constructed by computing the Bernstein coefficients as basis function coefficients and creating a parameter

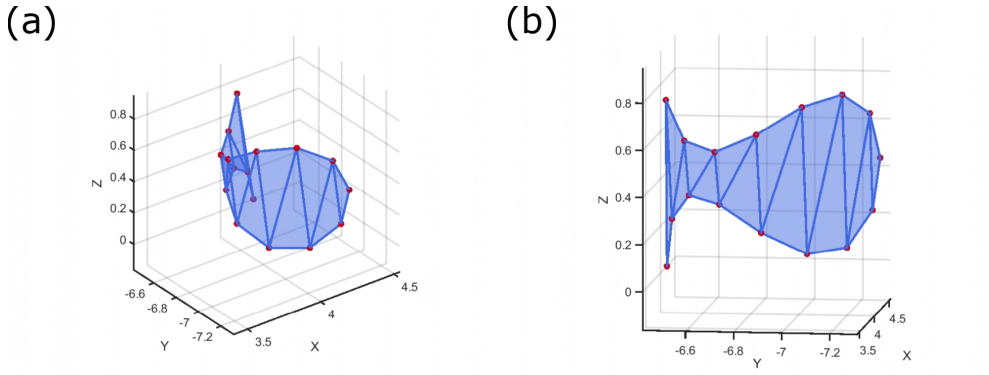

Figure S3: **Direct triangulation of the given nodes in the unit fabric shape.** Two views are provided in (a) and (b).

u ranging from 0 to 1 with the step size Δu (we use $\Delta u = 0.002$ in our computation). Then, by computing the corresponding Bernstein basis function value for each u and concatenating the results into a matrix, the resulting matrix can represent the desired curve, which is composed of a series of discrete points.

Here, the Bézier curve $P(u)$ is defined by control points P_i and Bernstein basis functions $B_i^n(u)$:

$$P(u) = \sum_{i=0}^{n} B_i^n(u) P_i \quad (u \in [0,1]), \tag{S1}$$

where the Bernstein basis functions are given by:

$$B_i^n(u) = \binom{n}{i} u^i (1-u)^{n-i}. \tag{S2}$$

Using the resulting four curves, a surface control mesh could be constructed. Specifically, one can traverse four Bézier curves simultaneously by selecting one point from each curve at a time, generating a new set of four points as control points, from which new Bézier curves can be constructed. Iteratively, all points on the original Bézier curve will be used once to construct the surface mesh intersecting the original ones. Hence, the resulting Bézier surface is composed of a doubly parameterized curve mesh, expressed as:

$$S(u,v) = \sum_{i=0}^{m} \sum_{j=0}^{n} B_i^m(u) B_j^n(v) P_{ij}, \tag{S3}$$

where $u, v \in [0,1]$. In practice, we can use the `bezret` and `bez3d` functions in MATLAB [2] to compute the Bézier surface based on the above-mentioned procedure.

Fig. S4(a)–(b) show the Bézier Surface constructed using seven groups of vertices: $\{i, i+1, 16-i, 17-i\}$, for $i = 1, 2, \ldots, 7$. Fig. S4(c)–(d) show the Bézier Surface constructed using three groups of vertices: $\{1,2,3,4,13,14,15,16\}$, $\{4,5,6,7,10,11,12,13\}$, and $\{7,8,9,10\}$.

Although the method above is intuitive and computationally simple, it is not precise enough. If too many groups are formed, the Bézier surface will be coarse (Fig. S4(a)–(b)),

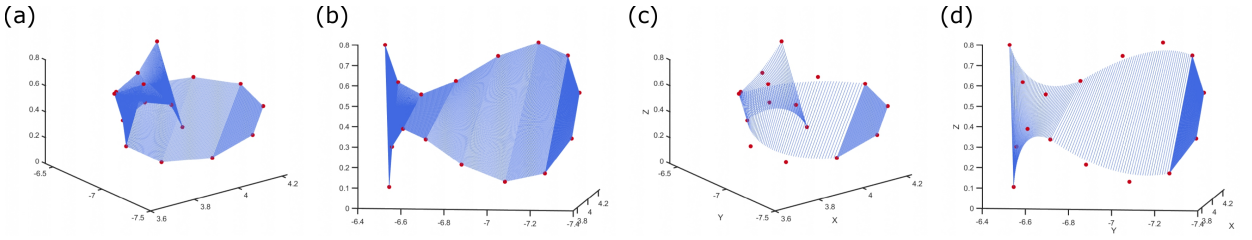

Figure S4: **Surface construction using the Bézier surface.** (a)–(b) Bézier surface constructed using seven groups of four points. (c)–(d) Bézier surface constructed using two groups of eight points and one group of four points.

while if a group contains too many original nodes, the surfaces generated exhibit deviation from the nodes (Fig. S4(c)–(d)). This motivates us to consider an improved Bézier surface construction approach, which will be explained in the next section.

S2.3 Improved Bézier Surface

We first parametrize the input (u, v) coordinates by linearly mapping them to $[0, 1] \times [0, 1]$ interval:

$$u' = \frac{u - u_{\min}}{u_{\max} - u_{\min}}, \quad v' = \frac{v - v_{\min}}{v_{\max} - v_{\min}}. \tag{S4}$$

Next, we construct the basis function matrix by computing 16 bicubic Bernstein basis functions, defined as:

$$B_{i,n}(u) = \binom{n}{i} u^i (1-u)^{n-i}, \tag{S5}$$

where $\binom{n}{i}$ is the binomial coefficient with $n = 3$ for cubic Bézier curves, $i, j \in [0, 3]$ for surface control points, and $u, v \in [0, 1]$ parameterize the unit domain.

The bicubic form combines two basis functions:

$$B_{i,j}(u, v) = B_{i,3}(u) B_{j,3}(v). \tag{S6}$$

Subsequently, we can determine the new 16 control points from the original ones by constructing an overdetermined system of equations and solving for the least-squares solution using the pseudo-inverse matrix. Specifically, with the original set of nodal points, which has been parametrized as (u_i, v_i) with corresponding surface points $\boldsymbol{P}_i$, we construct the overdetermined system:

$$\boldsymbol{B}\boldsymbol{K} = \boldsymbol{P}, \tag{S7}$$

where $\boldsymbol{B}$ is the $m \times 16$ basis function matrix ($m \geq 16$), $\boldsymbol{K}$ is the 16×3 control point matrix, and $\boldsymbol{P}$ is the $m \times 3$ point matrix. The least-squares solution for the above linear system is obtained using the Moore-Penrose pseudoinverse:

$$\boldsymbol{K} = \boldsymbol{B}^{+}\boldsymbol{P}, \tag{S8}$$

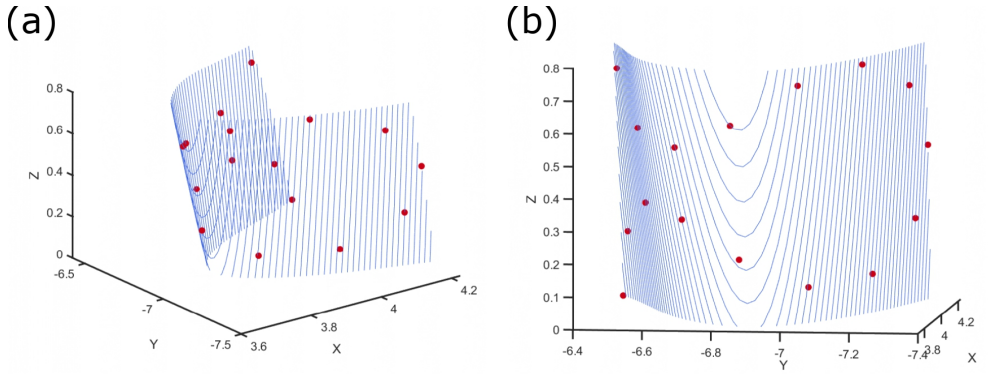

Figure S5: **Surface construction using improved Bézier surface.** Two views are provided in (a) and (b).

where the pseudoinverse is defined as:

$$\boldsymbol{B}^{+} = (\boldsymbol{B}^{T}\boldsymbol{B})^{-1}\boldsymbol{B}^{T}. \tag{S9}$$

Note that the above procedure is performed independently in three dimensions:

$$\boldsymbol{K}_x = \boldsymbol{B}^{+}\boldsymbol{P}_x \ , \quad \boldsymbol{K}_y = \boldsymbol{B}^{+}\boldsymbol{P}_y \ , \quad \boldsymbol{K}_z = \boldsymbol{B}^{+}\boldsymbol{P}_z. \tag{S10}$$

In practice, we use the MATLAB implementation as described in [3] to construct and solve the above-mentioned system.

Fig. S5 shows that the issue of excessive deviation between the surface and control points has been resolved. However, it also demonstrates the problem that the point cloud generated by the above code may extend beyond the unit surface area.

To compute the Delaunay triangulation, the points in the point cloud are first projected onto a certain plane (in our example, the x-y plane). Following this, the Delaunay triangulation will give a triangulation of the projected points, whose boundary is formed by the points on the convex hull of the projected points. Meanwhile, the triangulation will make the smallest interior angle of all triangles as large as possible to avoid elongated triangles. Then, we compute the area of the triangular patches with the original node positions, and summing them gives the surface area of the convex hull containing the point cloud.

Fig. S6 illustrates a simple example of a Delaunay triangulation, which fails to produce the correct surface we desire. Using the 16 nodal points as an example, the Delaunay triangulation method will return a surface that includes the side faces we do not need. However, if we attempt to manually remove this side face, Delaunay triangulation will then return a surface containing triangular faces that we do not intend to remove.

S2.4 Ruled surface

To construct a ruled surface from the reconstructed boundary curve, we select n_b points on the boundary curve.

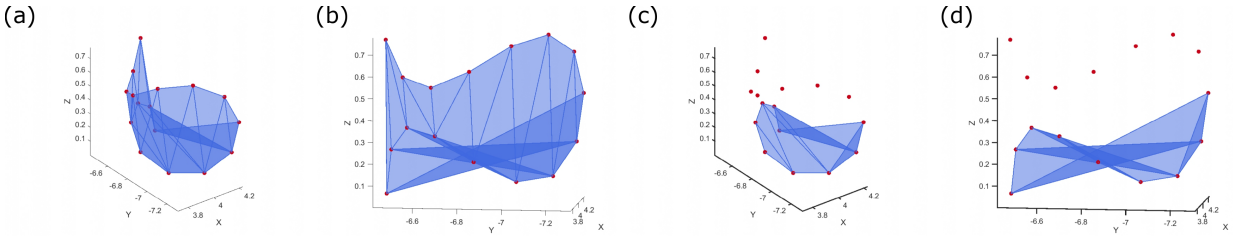

Figure S6: **Surface obtained using Delaunay triangulation.** (a)–(b) Surface obtained using Delaunay triangulation of projected 16 nodes. (c)–(d) Surface obtained using Delaunay triangulation of projected boundary points of the unwanted bottom surface.

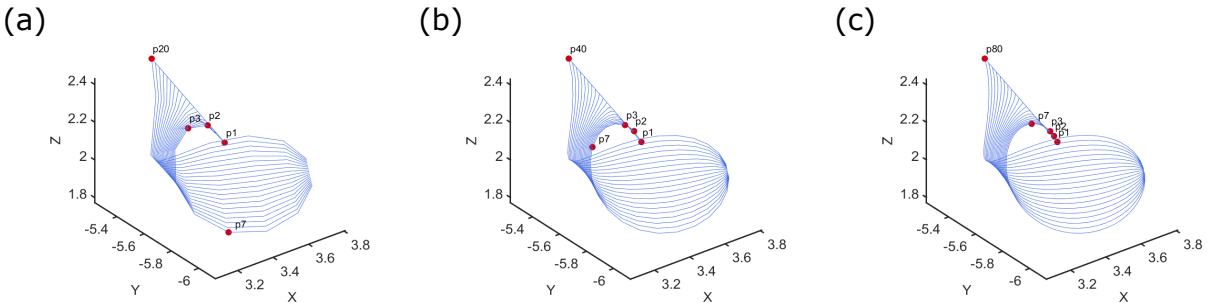

Figure S7: **Surface reconstruction obtained by combining the B-spline boundary curve and the ruled surface.** (a) Ruled surface using 20 boundary points. (b) Ruled surface using 40 boundary points. (c) Ruled surface using 80 boundary points.

First, we label the n_b points on the obtained boundary curve in order as $1, 2, \ldots, i, \ldots, n_b$, and denote their coordinates by $\vec{p}(i)$. We then parametrize the ruled surface S by (m, n) for $m \in \{1, 2, \ldots, \frac{n_b}{2}\}$, $n \in \{1, 2, \ldots, n_l\}$, and compute $S(m, n)$ by:

$$S(m,n) = \vec{\alpha}(m) + \frac{n-1}{n_l - 1}\vec{\beta}(m), \tag{S11}$$

where $\vec{\alpha}(m) = \vec{p}(m)$ and $\vec{\beta}(m) = \vec{p}(n_b - m + 1) - \vec{p}(m)$. In this way, we obtain a mesh of $n_b \times n_l$ sampling points on the ruled surface with the boundary curve being the given B-spline curve.

With larger n_b and n_l, a smoother surface could be plotted. For example, the surface in Fig. S7(a) uses 20 boundary points to reconstruct the boundary curve, and each generator thereafter uses 20 control points. Whereas the surface in Fig. S7(b) increased the number of boundary points to 40, and that in Fig. S7(c) increased to 80, maintaining the number of control points on each generator unchanged. Comparing the three surfaces, it is clear that the surface with more control points is smoother.

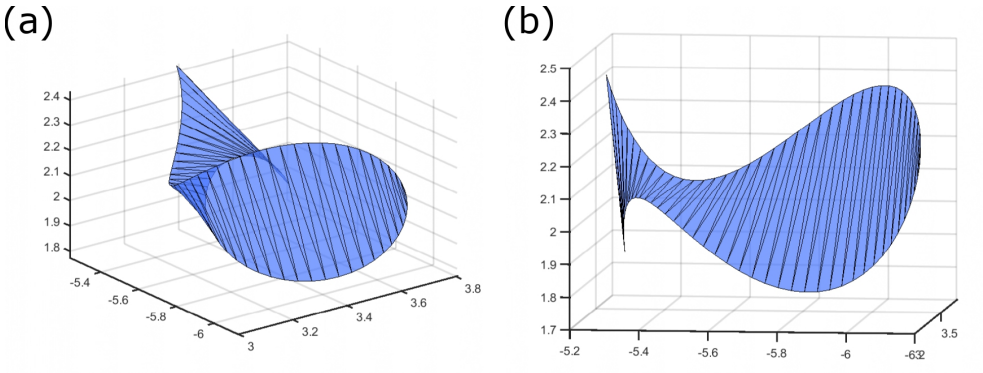

Figure S8: **Triangulation constructed using the B-spline boundary curve for unit area quantification.** Two views are provided in (a) and (b).

S2.5 Unit area quantification

Using the above-mentioned surface reconstruction methods, we can easily quantify the area of each unit cell.

For the triangulation approach, we can compute the area by directly summing up the areas of all triangle elements. As for computing the surface area using the B-spline boundary curve, we can start from the n_b sampling points on the boundary curve and perform a triangulation using each pair of adjacent points on the same side of the boundary and the point on the opposite side. For example, connecting points of index i, $i+1$, and the symmetrical point of index n_b+1-i can form a triangle located on the upper part of the surface; connecting points of index i, $i-1$, and the symmetrical point of index n_b+2-i can form a triangle located on the lower part of the surface. Fig. S8 demonstrates the triangulation result using the above methods. After forming the triangulation, we can compute the total area by summing the areas of the individual triangles.

S2.6 Surface curvature quantification

Through the ruled surface reconstruction, we can obtain $n_b \times n_l$ discrete points on the surface. The X, Y, Z coordinates of these points can be stored in three two-dimensional arrays X, Y, Z of size n_b by n_l (for example, $X(i,j)$ corresponds to the x-coordinate of the (i,j)-th point). Values such as X_u, X_v, X_{uu}, X_{uv} can be calculated using the gradient function in MATLAB. With partial derivatives of X, Y and Z along the u and v directions, coefficients of the first and second fundamental form E, F, G, e, f, g of the surface at discrete points can be calculated accordingly, and K (Gaussian curvature) and H (Mean curvature) follow from the formulas:

$$K = \frac{eg - f^2}{EG - F^2}, \tag{S12}$$

and

$$H = \frac{eG - 2fF + gE}{2(EG - F^2)}. \tag{S13}$$

Method	Number of Points	Unit Area	Percentage Difference	Remark
Monte-Carlo method	1300000	0.5876	-	Large computational cost
Ruled surface	1000	0.5718	2.6889%	More efficient with decent accuracy
Direct triangulation	16	0.5558	5.4118%	Unsatisfying accuracy

Table S1: Comparison between different surface reconstruction methods for surface area quantification.

S2.7 Comparison between different surface quantification methods

To assess the suitability of the above-mentioned surface reconstruction methods for quantifying the surface-based quantities of the fabric structures, we performed a detailed comparison between them in terms of the area and surface curvature obtained from them.

First, we compared the resulting unit surface area in Table S1. Among all the methods, the hybrid technique combining B-spline and ruled surface is the most satisfactory in balancing the accuracy and efficiency.

Next, we compare the surface curvature calculated by different surface mesh construction methods (Table S2). When using the Bézier surface with seven groups of four points, the range of each surface will be small. Therefore, the resulting surfaces will be more twisted. If two sets of eight points and a set of four points are used to construct the Bézier surface, the resulting surface will demonstrate severe deviation from the original nodes. Since a Bézier curve generally does not pass through the intermediate points, but instead forms a smoother curve among them, the generated surface is less curved than in reality. Hence, the curvature is smaller than reality. For the case when new control points of the Bézier surface are chosen, the resulting surface will not fit well with the boundary curve, thereby including redundant area for curvature calculation. Observing the surface constructed by the improved Bézier surface algorithm, the most curved part (around $y \in [-6.6, -7.1]$) has the most redundant area, which explains the reason for the larger curvature result. In comparison, when the B-spline curve and ruled surface are used to form a surface mesh, the reconstructed surface fits well with the original nodes, with no obvious sharp turnings or twisting, and no redundant area is included.

S3 Volume quantification

In this section, we present three approaches we considered for constructing and quantifying the 3D structure of the unit fabric.

Method	Number of Points	Gaussian Curvature	Mean Curvature	Remark
Bézier surface (7 partitions)	18207	12.1347	6.1706	Sharp turnings (large curvatures) occur at edges
Bézier surface (3 partitions)	7803	0.7784	0.4216	large deviation from original nodes
Bézier surface (new control points)	7803	6.2852	1.1821	Do not fit the boundary curve (redundant area)
Ruled surface	400	1.7884	0.6571	Use fewer points with an accurate range and a smooth surface

Table S2: Comparison between different surface reconstruction methods for surface curvature quantification.

S3.1 Axis-aligned bounding box

First, we considered using axis-aligned bounding boxes for the volume quantification. In this approach, we directly consider approximating the 3D shape using a bounding box aligned with the x, y, and z axes. We can then obtain the length, width, and height of the bounding box:

$$\Delta x = x_{\max} - x_{\min} \quad , \quad \Delta y = y_{\max} - y_{\min} \quad , \quad \Delta z = z_{\max} - z_{\min}, \tag{S14}$$

from which we can easily obtain various 3D measurements:

$$\text{Volume} = \Delta x \cdot \Delta y \cdot \Delta z, \tag{S15}$$

$$\text{Aspect ratio} = \frac{\Delta y}{\Delta x}, \tag{S16}$$

$$\text{Centroid} = \left[\frac{x_{\min} + x_{\max}}{2}, \frac{y_{\min} + y_{\max}}{2}, \frac{z_{\min} + z_{\max}}{2}\right]. \tag{S17}$$

S3.2 Minimal bounding box

Another approach for the volume quantification is to consider a minimal bounding box, which computes a bounding box with the minimum volume for a set of points without any assumption on the alignment with the coordinate axes. In particular, one can first compute the convex hull of the point cloud to rule out those points inside the convex hull, so that the problem size can be reduced. Following this, a local coordinate system can be established on each convex hull face using the Schmidt orthogonalization method. Subsequently, the point cloud is rotated into each local coordinate system, the corresponding bounding box dimensions

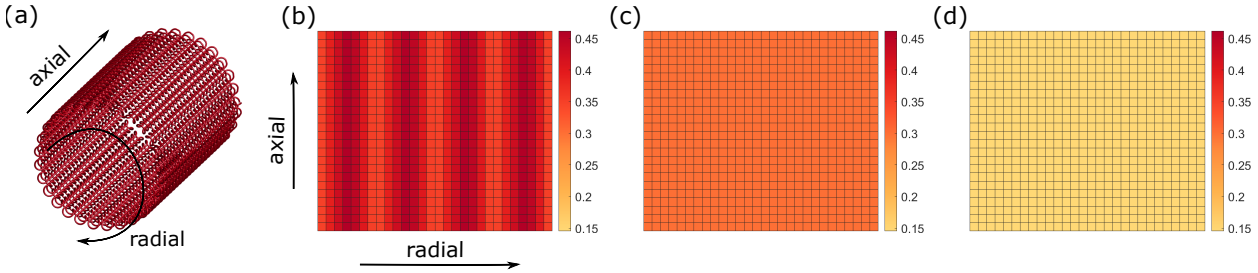

Figure S9: **Comparison between different volume quantification methods.** (a) A cylindrical sample used for the comparison. (b)–(d) We calculated the unit volume using three methods: (b) axis-aligned bounding box, (c) minimal bounding box, (d) convex hull.

are calculated, and then compared with the current optimal solution for updating. After comparing all the local coordinates, the best solution can be obtained. In practice, we obtain the minimum bounding box based on the above procedure using the `minboundbox` function in MATLAB [4]. After obtaining the minimum bounding box, we can further calculate the bounding box volume v and record the coordinates of its 8 corner points. Using the centroid of the minimal bounding box (i.e., the average of the coordinates of the eight corner points), the position of the unit fabric is determined. Finally, we calculate the aspect ratio as the ratio of the maximum distance of points in the point cloud along the x and y directions.

S3.3 Convex hull

We further consider computing the convex hull for volume quantification. Note that the convex hull of a given set of points is the smallest convex polyhedron that contains all the points. Finding the convex hull typically involves the following steps: First, one has to sort the set of points by polar angle (i.e., the angle with a reference point) to form an ordered sequence. Then, one can select the extreme points that constitute the convex hull boundary among the ordered points. Finally, the point set is divided into multiple subsets, and each subset is processed to approximate the convex hull boundary. In practice, we can utilize the built-in `convhulln` function in MATLAB to obtain the convex hull together with its volume. The position of each unit fabric is defined as the average of the extreme values of the coordinates of points in the point set: Centroid $= \left[\frac{x_{\min}+x_{\max}}{2}, \frac{y_{\min}+y_{\max}}{2}, \frac{z_{\min}+z_{\max}}{2}\right]$. The aspect ratio is defined the same as above: Aspect ratio $= \frac{\Delta y}{\Delta x}$.

S3.4 Comparison between different volume quantification methods

We tested three volume quantification methods on a cylindrical sample (Fig. S9(a)) to evaluate their capability of handling 3D structures.

We calculated the unit volume at a static state, and the resulting data demonstrated obviously different patterns (Fig. S9(b)–(d)). The data obtained using the axis-aligned bounding box illustrates a periodic pattern along the radial direction, which could be explained by the curling structure of the cylinder, which leads to the continuous rotation of

Method	Results	Orientation Invariance	Ratio to Minimum Value
Axis-aligned bounding box	0.3486-0.4613	No	239.92%-317.48%
Minimal bounding box	0.2999	Yes	206.40%
Convex hull	0.1453	Yes	100%

Table S3: Comparison of distances between surfaces generated by different surface construction methods and nodes.

the unit fabric. When a unit fabric is oriented parallel to a coordinate plane (e.g., the x-y plane), the three dimensions of the bounding box closely conform to the actual extent of the stitch, minimizing redundant space and occupying volume that closely approximates the true material occupancy. In contrast, when the unit stitch is aligned along a diagonal or oblique plane of the axis-aligned bounding box, the axis-aligned bounding box must encompass its maximum projections across all coordinate axes, resulting in significant void regions. This leads to overestimation of volume, which can compromise modeling accuracy and adversely affect subsequent mechanical analyses. When using the unit volume computed by the minimal bounding box (Fig. S9(c)) and convex hull (Fig. S9(d)), results illustrated a consistent pattern throughout the sampling fabric, confirming the previous discussion that these two methods are well-suited for spatial rotation. Moreover, the maximal unit volume computed by the axis-aligned bounding box, the minimal bounding box, and the convex hull are 0.35, 0.3, and 0.15, respectively, which is also consistent with previous assertions that the convex hull is the most effective in adapting to complex spatial structures and reducing redundant space.

Table S3 summarizes the characteristics of different volume calculation methods.

S4 Additional results

S4.1 Anisotropic mechanical responses

As discussed in the main text, after developing our geometric quantification framework, we can analyze the anisotropic mechanical responses for different fabric structures. In the main text, we presented the results for the jersey pattern.

In Fig. S10–S12, we further show the quantification of the anisotropic mechanical responses in the garter, rib, and seed patterns.

S4.2 Heterogeneous mechanical responses

In Fig. S13–S16, we show hot spots of local geometric quantities in four patterns. This is formed by overlaying heatmaps of temporal changes in the data with a certain level of transparency and by smoothing out boundaries between units to simulate realistic transitions between regions. We can therefore observe the uniformity of certain changes in quantity and detect regions undergoing the most intensive change.

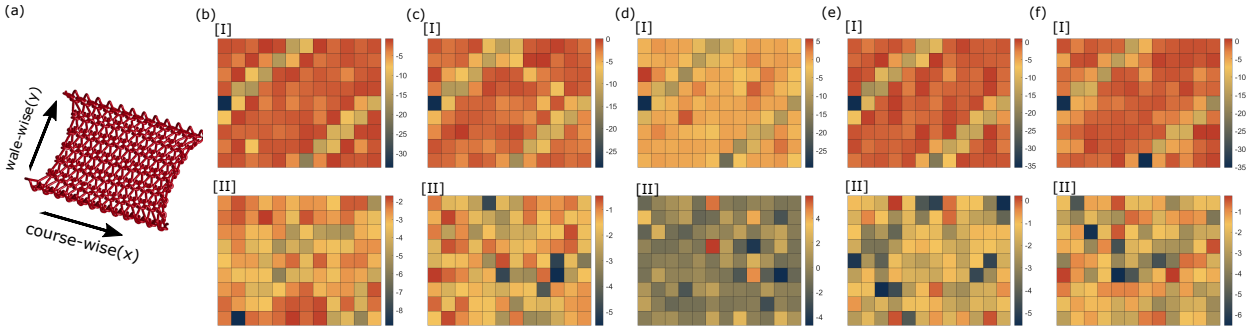

Figure S10: **Quantification of the anisotropic mechanical responses in the garter pattern.** (a) The garter pattern. (b)–(f) Spatial distribution of temporal changes in five representative quantities under tensile strain from 0% to 120%, where the color of each cell denotes the total change in the geometric quantity for the corresponding stitch: (b) Curve curvature, (c) Curve torsion, (d) Gaussian curvature, (e) Area, (f) Volume. Loading directions: [I] Course-wise tension. [II] Wale-wise Tension.

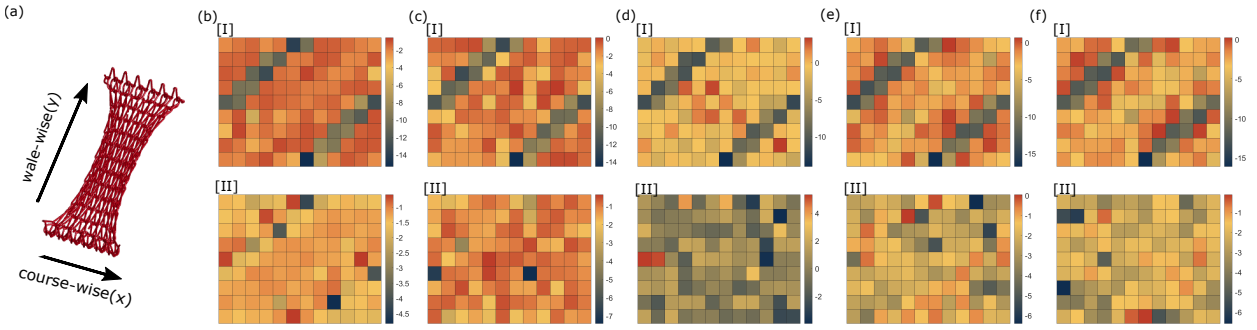

Figure S11: **Quantification of the anisotropic mechanical responses in the rib pattern.** (a) The rib pattern. (b)–(f) Spatial distribution of temporal changes in five representative quantities under tensile strain from 0% to 120%, where the color of each cell denotes the total change in the geometric quantity for the corresponding stitch: (b) Curve curvature, (c) Curve torsion, (d) Gaussian curvature, (e) Area, (f) Volume. Loading directions: [I] Course-wise Tension. [II] Wale-wise Tension.

S4.3 Mechanical responses from mixed-pattern fabrics

As for the mixed-pattern fabrics, in the main text we presented the analysis of their mechanical responses in terms of several geometric quantities. In particular, we considered two mixed-pattern fabrics:

- Pattern A (main text Fig. 4(a)): A mixed pattern with the top half being jersey and the bottom half being garter.
- Pattern B (main text Fig. 4(b)): A mixed pattern with the top and bottom quarter being jersey and the middle half being garter.

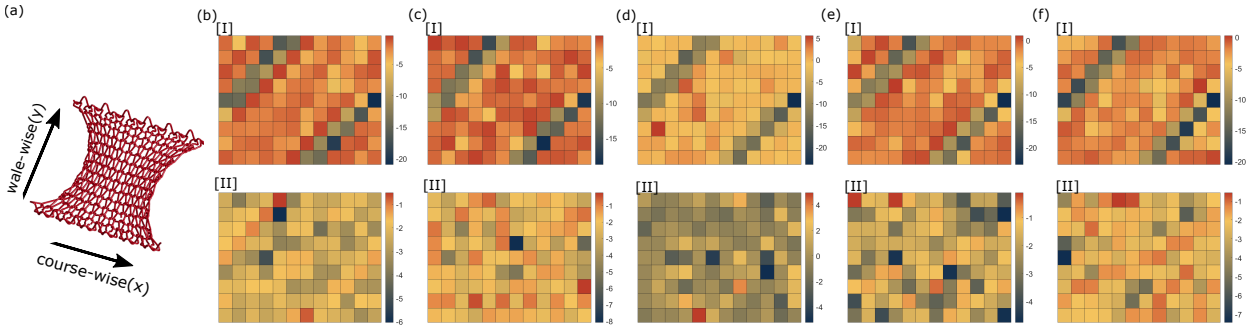

Figure S12: **Quantification of the anisotropic mechanical responses in the seed pattern.** (a) The seed pattern. (b)–(f) Spatial distribution of temporal changes in five representative quantities under tensile strain from 0% to 120%, where the color of each cell denotes the total change in the geometric quantity for the corresponding stitch: (b) Curve curvature, (c) Curve torsion, (d) Gaussian curvature, (e) Area, (f) Volume. Loading directions: [I] Course-wise tension. [II] Wale-wise tension.

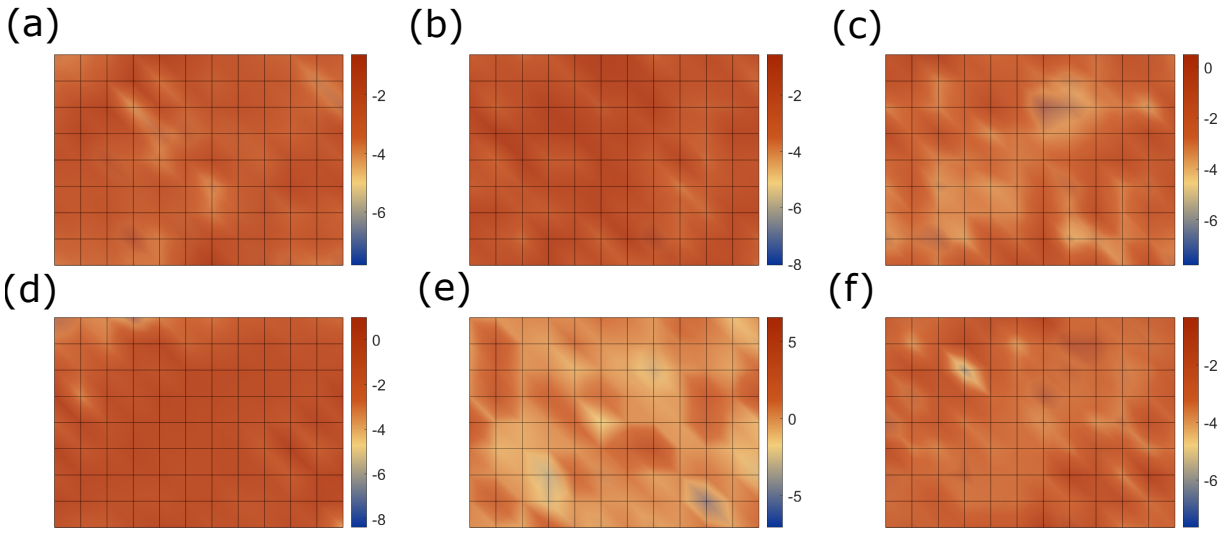

Figure S13: **Heterogeneous mechanical responses and hot spots of representative quantities in jersey fabrics under wale-wise tension.** Time-aggregated maps of changes in six key geometric quantities, generated by overlaying normalized heat maps acquired during successive tensile loading at 0%, 20%, 40%, 60%, 80%, 100%, and 120% strain with applied spatial smoothing: (a) Curve curvature. (b) Curve torsion. (c) Area. (d) Gaussian curvature. (e) Mean curvature. (f) Aspect ratio. (g) Volume.

In Fig. S17, we further present additional geometric quantities for both Pattern A and Pattern B, including the curve torsion variation, area variation, Gaussian curvature variation, and mean curvature variation.

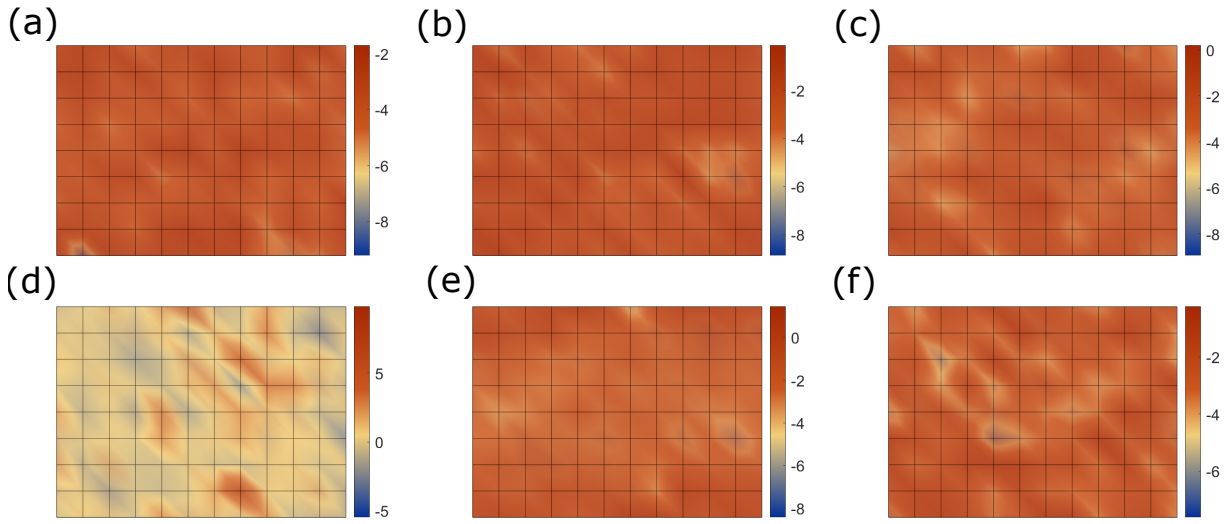

Figure S14: **Heterogeneous mechanical responses and hot spots of representative quantities in garter fabrics under wale-wise tension.** Time-aggregated maps of changes in six key geometric quantities, generated by overlaying normalized heat maps acquired during successive tensile loading at 0%, 20%, 40%, 60%, 80%, 100%, and 120% strain with applied spatial smoothing: (a) Curve curvature. (b) Curve torsion. (c) Area. (d) Gaussian curvature. (e) Mean curvature. (f) Aspect ratio. (g) Volume.

S4.4 Temporal change of the spatial variation of 2.5D fabrics: Onset and evolution of hot spots

As for the temporal change of the spatial variation of 2.5D fabrics, besides the jersey results shown in the main text, here we further analyze the onset and evolution of hot spots in the garter, rib, and seed patterns (Fig. S18–S20).

References

[1] B. Friedrich, "frenet_robust, MATLAB Central File Exchange," 2014. https://www.mathworks.com/matlabcentral/fileexchange/47885-frenet_robust-zip, accessed on November 18, 2025.

[2] S. Mohanty, "Bezier Surface, MATLAB Central File Exchange," 2015. https://www.mathworks.com/matlabcentral/fileexchange/66678-bezier-surface, accessed on November 18, 2025.

[3] G. P. T. Choi, L. H. Dudte, and L. Mahadevan, "Programming shape using kirigami tessellations," *Nat. Mater.*, vol. 18, no. 9, pp. 999–1004, 2019.

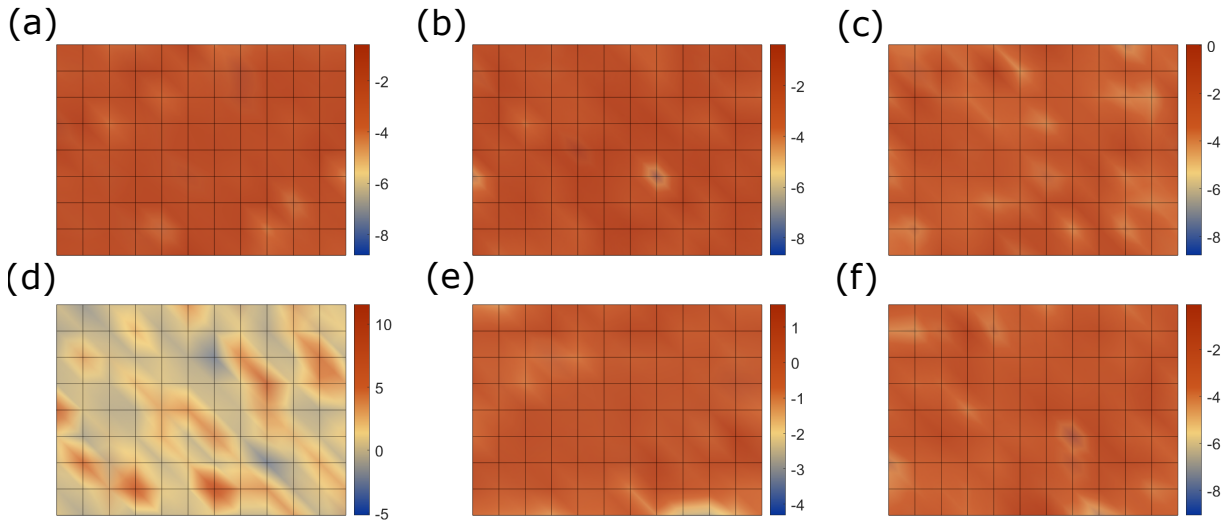

Figure S15: **Heterogeneous mechanical responses and hot spots of representative quantities in rib fabrics under wale-wise tension.** Time-aggregated maps of changes in six key geometric quantities, generated by overlaying normalized heat maps acquired during successive tensile loading at 0%, 20%, 40%, 60%, 80%, 100%, and 120% strain with applied spatial smoothing: (a) Curve curvature. (b) Curve torsion. (c) Area. (d) Gaussian curvature. (e) Mean curvature. (f) Aspect ratio. (g) Volume.

[4] J. Korsawe, "Minimal Bounding Box, MATLAB Central File Exchange," 2015. https://www.mathworks.com/matlabcentral/fileexchange/18264-minimal-bounding-box, accessed on November 18, 2025.

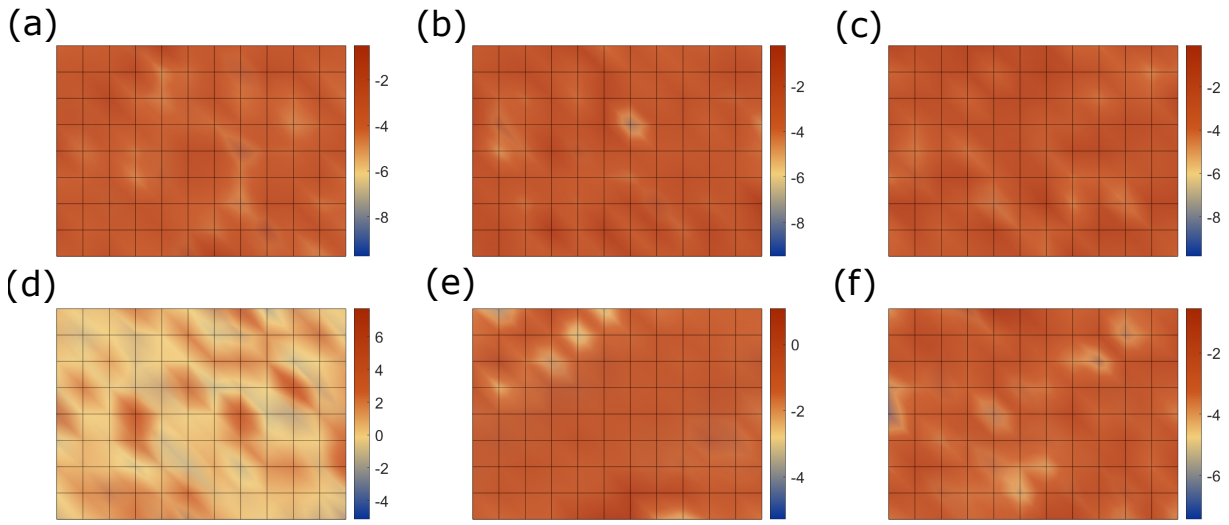

Figure S16: **Heterogeneous mechanical responses and hot spots of representative quantities in seed fabrics under wale-wise tension.** Time-aggregated maps of changes in six key geometric quantities, generated by overlaying normalized heat maps acquired during successive tensile loading at 0%, 20%, 40%, 60%, 80%, 100%, and 120% strain with applied spatial smoothing: (a) Curve curvature. (b) Curve torsion. (c) Area. (d) Gaussian curvature. (e) Mean curvature. (f) Aspect ratio. (g) Volume.

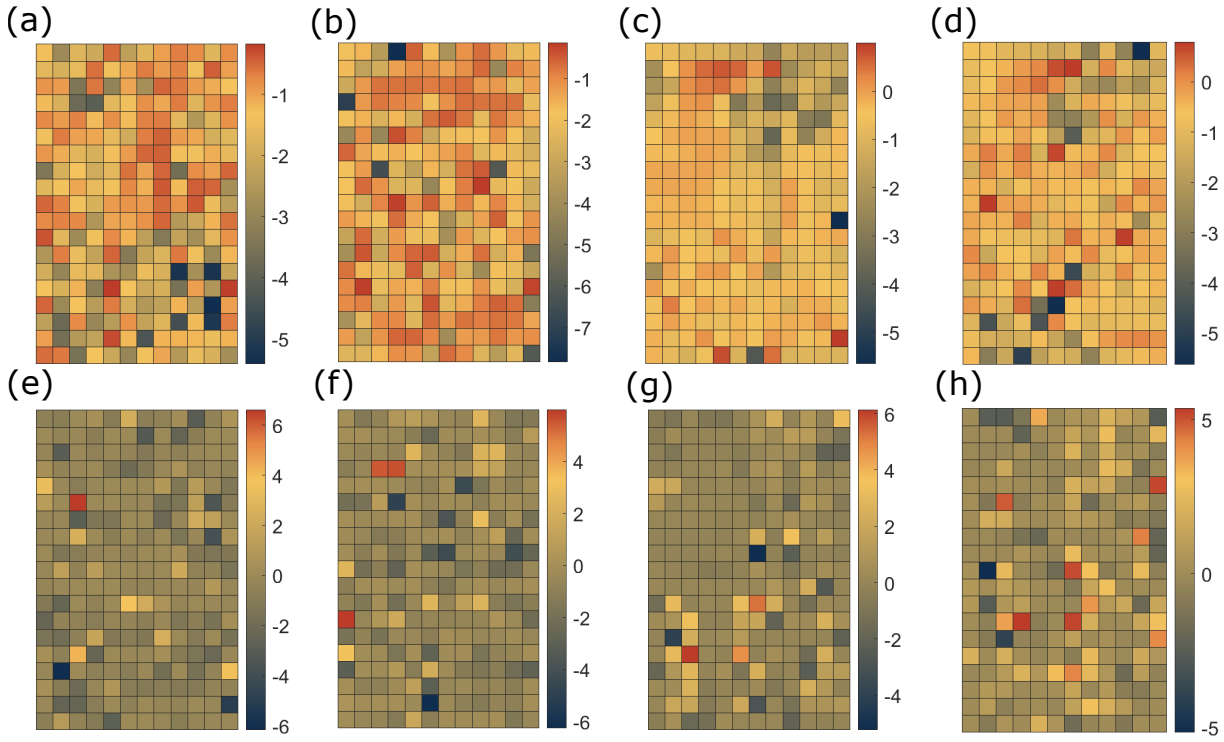

Figure S17: **Quantifying the mechanical responses from mixed-pattern fabrics.** Spatial distribution of geometric quantity changes in two mixed-pattern fabrics under wale-wise uniaxial tensile strain from 0% to 180%: (a) Curve torsion change for pattern A. (b) Curve torsion change for pattern B. (c) Area change for pattern A. (d) Area change for pattern B. (e) Gaussian curvature change for pattern A. (f) Gaussian curvature change for pattern B. (g) Mean curvature change for pattern A. (h) Mean curvature change for pattern B.

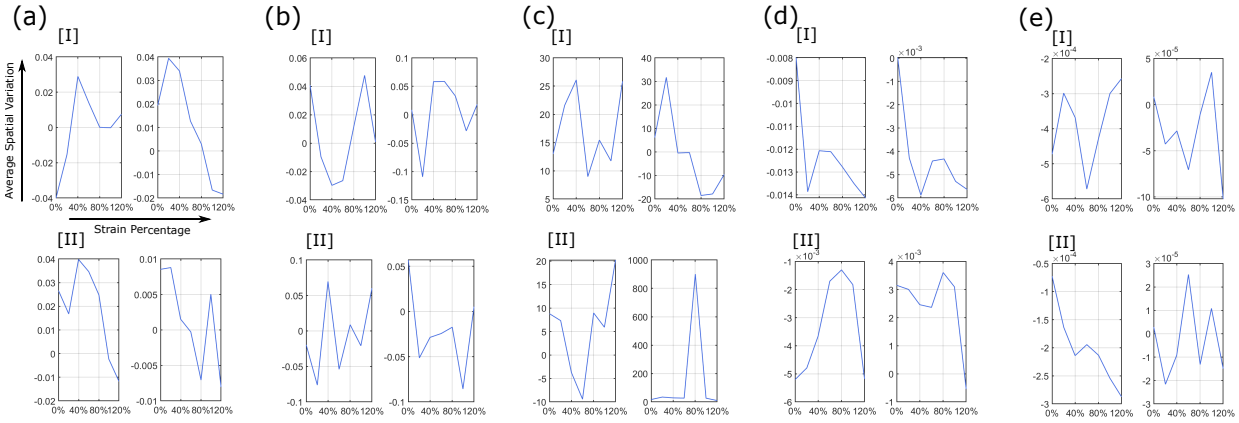

Figure S18: **Temporal changes in spatial variations of four representative quantities for garter fabrics, recorded across a tensile strain range of 0% to 120%.** (a) Curve curvature. (b) Curve torsion. (c) Gaussian curvature. (d) Area. (e) Volume. Loading directions: [I] Course-wise. [II] Wale-wise. In each of (a)–(e), the left plot corresponds to course-wise variation and the right plot corresponds to wale-wise variation.

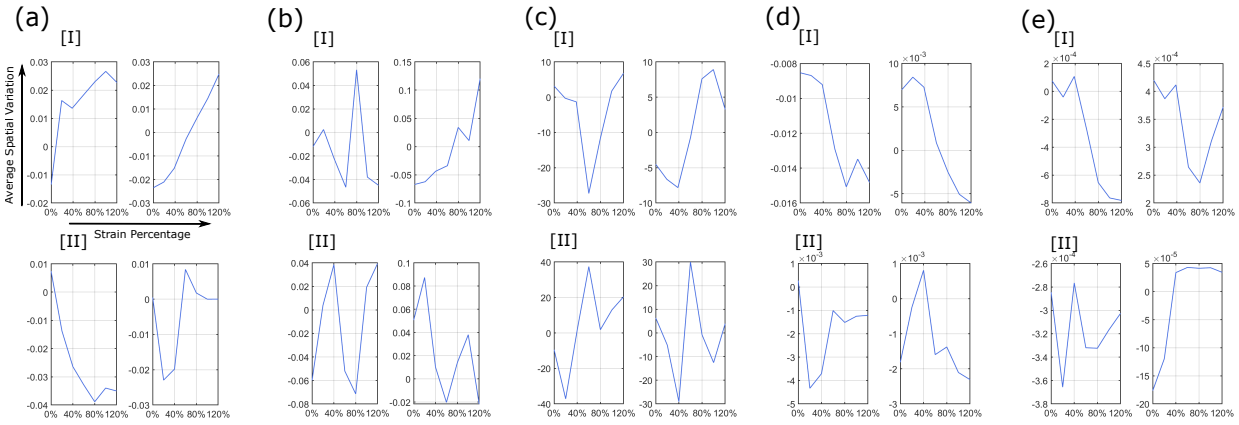

Figure S19: **Temporal changes in spatial variations of four representative quantities for rib fabrics, recorded across a tensile strain range of 0% to 120%.** (a) Curve curvature. (b) Curve torsion. (c) Gaussian curvature. (d) Area. (e) Volume. Loading directions: [I] Course-wise. [II] Wale-wise. In each of (a)–(e), the left plot corresponds to course-wise variation and the right plot corresponds to wale-wise variation.

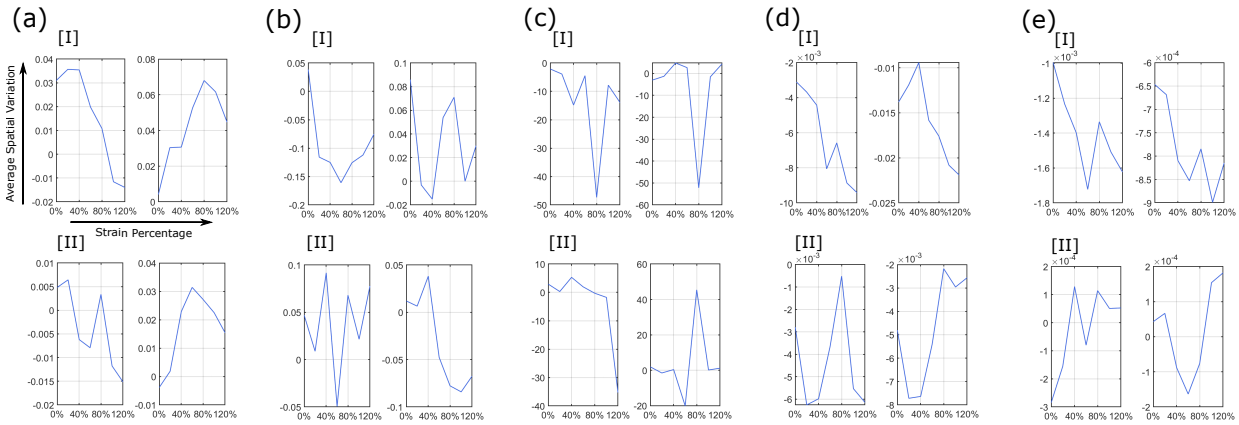

Figure S20: **Temporal changes in spatial variations of four representative quantities for seed fabrics, recorded across a tensile strain range of 0% to 120%.** (a) Curve curvature. (b) Curve torsion. (c) Gaussian curvature. (d) Area. (e) Volume. Loading directions: [I] Course-wise. [II] Wale-wise. In each of (a)–(e), the left plot corresponds to course-wise variation and the right plot corresponds to wale-wise variation.