# Homogenized Yarn-Level Cloth

*Pseudocode*

GEORG SPERL, IST Austria
RAHUL NARAIN, Indian Institute of Technology Delhi
CHRIS WOJTAN, IST Austria

This document contains pseudocode for each step in the pipeline of our method: on the microscale, we prepare the reference configuration of patterns under tension (Algorithm C1) and find the energy minimum using constrained Newton optimization (Algorithm C2, Algorithm C3); we then compute many such minima for various sampled deformations (Algorithm C4); we fit this data with regularized splines (Algorithm C5, Algorithm C6, Algorithm C7); finally, we use the fitted model to compute forces in a cloth solver (Algorithm C8).

For notation, we reference main paper sections as "Section 1.2" and equations as "(10)", and supplementary with the prefix S as "Section S1.2" and equations as "(S10)".

**Algorithm C1** Compute reference yarn geometry under tension and at rest wrt. stretching (textual explanation in Section S2.3).

**Input:** periodic yarn geometry not at rest
periodic lengths $p_{\xi_1}, p_{\xi_2}$
material parameters
**Output:** reference-state geometry $\xi_1, \xi_2, h$
rest lengths, rest curvatures, rest twists, …

**procedure** ComputeReferenceState
  generate stress-free state:
    **repeat**
      reset rest lengths, rest curvatures and rest twists ▷ setting rest values to current values
      relax yarns ▷ Algorithm C3
    **until** $E < 10^{-10}$
  apply tension:
    **if** knitted **then**
      rest lengths ← 0.9 rest lengths
      rest curvatures ← 0.8 rest curvatures
    **else**
      rest curvatures ← 0.9 rest curvatures
    **end if**
  find in-plane minimum: $\min_{p_{\xi_1}, p_{\xi_2}} \int_\Omega \Psi$
    modify $p_{\xi_1}, p_{\xi_2}$ and compute total energy at equilibrium
  translate center to $\int_\Omega(\xi_1, \xi_2, h) = \mathbf{0}$
**end procedure**

**Algorithm C2** Initialize micro-scale Newton solver.

**Input:** reference yarn pattern ▷ Algorithm C1
strains $s_x$, $s_a$, $s_y$, $\overline{\mathbf{II}}$

**procedure** Initialize
  compute $\boldsymbol{\varphi}$: ▷ Section S1.1
    compute $\bar{\mathbf{I}}$ ▷ (30)
    compute $\bar{S}$, $\boldsymbol{n}_{,\alpha}$ ▷ (S11), (S12)
    solve for $\boldsymbol{\varphi}$ on a grid ▷ (16), (17)
  set initial guess $\tilde{\boldsymbol{u}} = \mathbf{0}$ by setting each vertex to $\boldsymbol{\varphi} + h\,\boldsymbol{n}$
  initialize static AABB-tree collision broadphase
  initialize constraints:
    prepare individual constraints (20), (25), (18), (27)
    concatenate $\boldsymbol{C}, \boldsymbol{d}$ ▷ (20), (25)
    concatenate $\boldsymbol{C}_L, \boldsymbol{d}_L$ ▷ (18), (27)
    compute elimination matrix $\tilde{\boldsymbol{C}}$ and $\tilde{\boldsymbol{d}}$ ▷ Appendix A
    precompute $\boldsymbol{C}_L\tilde{\boldsymbol{C}}$ and $(\boldsymbol{C}_L\tilde{\boldsymbol{C}}\tilde{\boldsymbol{C}}^\top\boldsymbol{C}_L^\top)^{-1}$
  compute unit scaling matrix $\boldsymbol{M}$ ▷ (S22)
  compute initial $\boldsymbol{y}$ ▷ Appendix A
  project reference frames ▷ (24)
**end procedure**

**Algorithm C3** Step micro-scale Newton solver.

**procedure** NewtonStep
  compute forces $\nabla E$
  assemble rhs. of (S22)
  **if** stopping criterion (S25) **then**
    return energy area density $\overline{\Psi} = E/A$
  **end if**
  compute Hessian $\boldsymbol{H}$
  assemble lhs. of (S22)
  solve for $\delta\boldsymbol{y}$ in (S22)
  limit maximum displacement:
    $d_{\max} \leftarrow r\,\lambda_{\min}(\bar{\mathbf{I}})$ ▷ Section S2.2
    $\delta\boldsymbol{q} \leftarrow \tilde{\boldsymbol{C}}\,\delta\boldsymbol{y}$
    $d \leftarrow$ maximum vertex displacement $(\delta\boldsymbol{q})$
    **if** $d > d_{\max}$ **then**
      $\delta\boldsymbol{y} \leftarrow d_{\max}/d\;\delta\boldsymbol{y}$
    **end if**
  backtracking linesearch:
    store parallel transport quantities ▷ $\underline{\boldsymbol{d}}_\alpha$ etc.
    $\boldsymbol{y}_0 \leftarrow \boldsymbol{y}$
    **for** $i \leftarrow 0 \ldots 10$ **do**
      restore parallel transport quantities
      $\boldsymbol{y} \leftarrow \boldsymbol{y}_0 + 0.1^i\;\delta\boldsymbol{y}$
      $\boldsymbol{q} \leftarrow \tilde{\boldsymbol{C}}\boldsymbol{y} + \tilde{\boldsymbol{d}}$
      update parallel transport quantities
      update collision broadphase
      compute total energy $E$
      **if** $E$ improved **then**
        continue to next NewtonStep
      **end if**
    **end for**
**end procedure**

**Algorithm C4** Sample deformation ranges for fitting. All individual simulations can be run in parallel.

**Input:** 1D-sampling ranges $[z_{\min}, z_{\max}]$ for each strain $z$
2D-sampling ranges $[z_{i\min}, z_{i\max}], [z_{j\min}, z_{j\max}]$ for $(z_i, z_j)$

**function** SampleRange($z$, $z_{\min}$, $z_{\max}$, $N$)
  **if** $z = s_x$ **or** $z = s_y$ **then**
    $Z \leftarrow \text{linspace}\big((z_{\min}+1)^{\frac{1}{10}}, (z_{\max}+1)^{\frac{1}{10}}, N\big)^{10} - 1$
  **else**
    $Z \leftarrow \text{linspace}\big(-\sqrt{\text{abs}(z_{\min})}, \sqrt{\text{abs}(z_{\max})}, N\big)$
    $Z \leftarrow \text{sgn}(Z)\, Z^2$
  **end if**
  **return** $Z$
**end function**

**procedure** Sample
  **for** $z$ **in** $\{s_x, s_a, s_y, \text{II}_{00}, \text{II}_{11}\}$ **do**
    $Z \leftarrow$ SampleRange($z$, $z_{\min}$, $z_{\max}$, 150)
    simulate each sample $\boldsymbol{z} = (0 \ldots, z_i = Z, \ldots 0)$
  **end for**
  **for all** $\{z_i, z_j\}$ **do**
    $Z_i \leftarrow$ SampleRange($z_i$, $z_{i\min}$, $z_{i\max}$, 50)
    $Z_j \leftarrow$ SampleRange($z_j$, $z_{j\min}$, $z_{j\max}$, 50)
    simulate each sample $\boldsymbol{z} = (0 \ldots, z_i = Z_i, z_j = Z_j, \ldots 0)$
  **end for**
**end procedure**

**Algorithm C5** Fit material model splines from data (code supplied with paper).

**Input:** data $(s_x, s_a, s_y, \text{II}_{00}, \text{II}_{11}; \overline{\Psi})$ for 1D and 2D ranges
**Output:** $f_0, f_i, f_x, f_y, f_{ij}, f_{ix}, f_{iy}$

**procedure** FitSplines
  $X \leftarrow$ normalize(strains)
    ▷ divide each coord. by max. abs. value in data
  $f_0 \leftarrow \Psi(\mathbf{0})$
  **for** $i \in \{1, 2, 3, x, y\}$ **do**
    gather $X^i$, $\overline{\Psi}^i$ ▷ strain and energy for data range
    $f_i \leftarrow$ Fit1D($X^i$,$\overline{\Psi}^i$) ▷ Algorithm C6
  **end for**
  **for** $ij \in \{12, 13, 23, 1x, 2x, 3x, 1y, 2y, 3y\}$ **do**
    gather $X^{ij}$, $Y^{ij}$, $\overline{\Psi}^{ij}$ ▷ strains and energy for data range
    $f_{ij} \leftarrow$ Fit2D($X^{ij}$,$Y^{ij}$,$\overline{\Psi}^{ij}$) ▷ Algorithm C7
  **end for**
**end procedure**

**Algorithm C6** Fit 1D-splines (code supplied with paper).

**Input:** strains $X$, energy densities $\overline{\Psi}$
**Output:** cubic hermite spline $f$ with coefficients
  $\boldsymbol{x}$ …control point locations
  $\boldsymbol{p}$ …control point values
  $\boldsymbol{p}^x$ …control point derivatives

**function** Fit1D($X$, $\overline{\Psi}$)
  MLS:
    def. $g(u) = \text{symexp}(\text{MLS}(u \mid X, \text{symlog}(\overline{\Psi})))$
      ▷ for $s_a$ replace $\text{MLS}(u) = 0.5\,(\text{MLS}(u) + \text{MLS}(-u))$
  INITIAL FIT:
    $\boldsymbol{x} \leftarrow$ linspace within $[0.95 \min X, 0.95 \max X]$
      ▷ include $x = 0$
    $\boldsymbol{p} \leftarrow g(\boldsymbol{x})$
    $\boldsymbol{p}^x \leftarrow$ finite-difference $g(u)$
  QUASICONVEXIFY:
    $p^x_{\min} \leftarrow 0.001$ ▷ 0.01 if stockinette
    **if** in-plane strain **then**
      $i_{\min} \leftarrow x_{i_{\min}} = 0$
    **else**
      $i_{\min} \leftarrow \text{argmin}_i\,(p_i)$ ▷ find bending min.
    **end if**
    march outward from $i_{\min}$ and enforce:
    $p_{i+1} \geq p_i + (x_{i+1} - x_i)\, p^x_{\min}$ ▷ $i-1$ if $i < i_{\min}$
    $p^x_i \geq p^x_{\min}$ ▷ $p^x_i \leq -p^x_{\min}$ if $i < i_{\min}$
  apply monotone cubic algorithm ▷ [Fritsch1980]
  ensure increasing extrapolation ▷ clamp $p^x$ on boundary
  $p_i \leftarrow p_i - p_{x=0}$ ▷ convert to residual
  **return** $f \leftarrow \{\boldsymbol{x}, \boldsymbol{p}, \boldsymbol{p}^x\}$
**end function**

**Algorithm C7** Fit 2D-splines (code supplied with paper).

**Input:** strains $X$ and $Y$, energy densities $\overline{\Psi}$
**Output:** bicubic hermite spline $f$ with coefficients
$\boldsymbol{x}$ …control point locations
$\boldsymbol{p}$ …control point values
$\boldsymbol{p}^x, \boldsymbol{p}^y, \boldsymbol{p}^{xy}$ …control point derivatives

**function** FIT2D($X$, $Y$, $\overline{\Psi}$)
  MLS:
    def. $g(u,v) = \text{symexp}(\text{MLS}(u, v \mid X, Y, \text{symlog}(\overline{\Psi})))$
    ▷ for $s_a$ replace $\text{MLS}(u,v) = 0.5\ (\text{MLS}(u,v) + \text{MLS}(-u,v))$
  INITIAL FIT:
    $\boldsymbol{x} \leftarrow$ linspace [0.9*min $X$, 0] and [0, 0.9*max $X$]
    $\boldsymbol{y} \leftarrow$ linspace [0.9*min $Y$, 0] and [0, 0.9*max $Y$]
    ▷ for stockinette replace 0.9 with 0.3
    $\boldsymbol{p} \leftarrow g(\boldsymbol{x}, \boldsymbol{y})$
    $\boldsymbol{p}^x, \boldsymbol{p}^y, \boldsymbol{p}^{xy} \leftarrow$ finite-difference $g(u,v)$
  QUASICONVEXIFY: ▷ if not ($s_x$, $s_y$)
    $p^x_{\min} \leftarrow 0.001$ ▷ 0.01 if stockinette
    $i_{\min} \leftarrow x_{i_{\min}} = 0$ ▷ $i$ always in-plane
    **if** $Y$ is in-plane strain **then**
      $j_{\min} \leftarrow y_{j_{\min}} = 0$
    **else**
      $j_{\min} \leftarrow \text{argmin}_j\,(p(x_{i_{\min}}, y_j))$ ▷ find bending min.
    **end if**
    march outward from $i_{\min}, j_{\min}$ and enforce:
    $p_{i+1,j} \geq p_{i,j} + (y_{i+1} - y_i)\, p^x_{\min}$ ▷ $i-1$ if $i < i_{\min}$
    $p_{i,j+1} \geq p_{i,j} + (x_{i+1} - x_i)\, p^x_{\min}$ ▷ $j-1$ if $j < j_{\min}$
    $p^y_{ij} \geq p^x_{\min}$ ▷ $p^y_{ij} \leq -p^x_{\min}$ if $i < i_{\min}$
    $p^x_{ij} \geq p^x_{\min}$ ▷ $p^x_{ij} \leq -p^x_{\min}$ if $j < j_{\min}$
  ENSURE INCREASING EXTRAPOLATION:
    clamp $p^x$ and $p^y$ on boundary
    $p^{xy} \leftarrow 0$ on boundary
  apply monotone bicubic algorithm ▷ Section S5
  CONVERT TO RESIDUAL: ▷ r(x,y) = f(x,y) - f(x,0) - f(0,y) + f(0,0)
    $p_{i,j} \leftarrow p_{i,j} - p_{y=0,j} - p_{i,x=0} + p_{y=0,x=0}$
    $p^x_{i,j} \leftarrow p^x_{i,j} - p_{y=0,j}$
    $p^y_{i,j} \leftarrow p^x_{i,j} - p_{i,x=0}$
  0-COMPRESSION RESIDUAL: ▷ if not ($s_x$, $s_y$)
    **for** $\{i,j\}$ compressed **do**
      $p_{i,j} \leftarrow p^x_{i,j} \leftarrow p^y_{i,j} \leftarrow p^{xy}_{i,j} \leftarrow 0$
    **end for**
  **return** $f \leftarrow \{\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{p}, \boldsymbol{p}^x, \boldsymbol{p}^y, \boldsymbol{p}^{xy}\}$
**end function**

**Algorithm C8** On the macro-scale, our method replaces only the computation of energy and its derivatives.

**procedure** COMPUTEDERIVATIVES
  **for all** triangles $\triangle$ **do** ▷ parallel
    compute $\boldsymbol{z}$, $\frac{\partial \boldsymbol{z}}{\partial \boldsymbol{q}_\triangle}$, $\frac{\partial^2 \boldsymbol{z}}{\partial \boldsymbol{q}_\triangle \partial \boldsymbol{q}_\triangle}$ ▷ (35), (36), (30), (S32)
    compute $\frac{\partial \overline{\Psi}_\triangle}{\partial \boldsymbol{z}}$, $\frac{\partial^2 \overline{\Psi}_\triangle}{\partial \boldsymbol{z} \partial \boldsymbol{z}}$ ▷ (S26)
    compute $\frac{\partial E_\triangle}{\partial \boldsymbol{q}_\triangle}$, $\frac{\partial^2 E_\triangle}{\partial \boldsymbol{q}_\triangle \partial \boldsymbol{q}_\triangle}$ ▷ (S43), (S44)
  **end for**
  assemble global force $\frac{\partial E}{\partial \boldsymbol{q}}$ and Hessian $\frac{\partial^2 E}{\partial \boldsymbol{q} \partial \boldsymbol{q}}$
**end procedure**

## REFERENCES

Frederick N Fritsch and Ralph E Carlson. 1980. Monotone piecewise cubic interpolation. *SIAM J. Numer. Anal.* 17, 2 (1980), 238–246.