Supplement to Lifted Curls: A Model for Tightly Coiled Hair Simulation

Alvin Shi* alvin.shi@yale.edu Yale University New Haven, CT, USA

Haomiao Wu* haomiao.wu@yale.edu Yale University New Haven, CT, USA

Jarred Parr jarred.parr@yale.edu Yale University New Haven, CT, USA

A.M. Darke darke@ucsc.edu UC Santa Cruz Santa Cruz, CA, USA

Theodore Kim theodore.kim@yale.edu Yale University USA

1 RANK-ONE UPDATE OF $\frac{\partial^2\theta}{\partial\mathbf{T}^2}$

We will apply the Bunch-Nielsen-Sorensen (BNS) algorithm [Bunch et al. 1978] to the analytic eigensystem of $\frac{\partial^2\theta}{\partial\mathbf{T}^2}$ in order to obtain the analytic eigensystem of:

$$\frac{\partial^2\Psi_b}{\partial\mathbf{T}^2} = \frac{\partial^2\Psi_b}{\partial\theta^2}\frac{\partial\theta}{\partial\mathbf{T}}\frac{\partial\theta}{\partial\mathbf{T}}^\top + \frac{\partial\Psi_b}{\partial\theta}\frac{\partial^2\theta}{\partial\mathbf{T}^2}. \tag{1}$$

The analytic eigensystem of $\frac{\partial^2\theta}{\partial\mathbf{T}^2}$ is already known:

$$\lambda_0^\theta = \frac{\cos\theta - 1}{\|\mathbf{b}\|} \qquad \mathbf{q}_0^\theta = \frac{1}{\sqrt{2}\|\mathbf{b}\|}\begin{bmatrix}\mathbf{b}\\ \mathbf{b}\end{bmatrix} \tag{2}$$

$$\lambda_1^\theta = \frac{\cos\theta + 1}{\|\mathbf{b}\|} \qquad \mathbf{q}_1^\theta = \frac{1}{\sqrt{2}\|\mathbf{b}\|}\begin{bmatrix}\mathbf{b}\\ -\mathbf{b}\end{bmatrix} \tag{3}$$

$$\lambda_2^\theta = -1 \qquad \mathbf{q}_2^\theta = \frac{1}{\sqrt{2}}\begin{bmatrix}\mathbf{t}_0 + \mathbf{b}_0\\ \mathbf{0}_3\end{bmatrix} \tag{4}$$

$$\lambda_3^\theta = 1 \qquad \mathbf{q}_3^\theta = \frac{1}{\sqrt{2}}\begin{bmatrix}\mathbf{t}_0 - \mathbf{b}_0\\ \mathbf{0}_3\end{bmatrix} \tag{5}$$

$$\lambda_4^\theta = -1 \qquad \mathbf{q}_4^\theta = \frac{1}{\sqrt{2}}\begin{bmatrix}\mathbf{0}_3\\ \mathbf{t}_1 + \mathbf{b}_1\end{bmatrix} \tag{6}$$

$$\lambda_5^\theta = 1 \qquad \mathbf{q}_5^\theta = \frac{1}{\sqrt{2}}\begin{bmatrix}\mathbf{0}_3\\ \mathbf{t}_1 - \mathbf{b}_1\end{bmatrix} \tag{7}$$

To review from the main document, we are examining three vertices that form a triangle, $\mathbf{x}_{0,1,2}$, denote the two strand edges as $\mathbf{e}_0 = \mathbf{x}_0 - \mathbf{x}_1$, $\mathbf{e}_1 = \mathbf{x}_2 - \mathbf{x}_1$, and normalized versions of these edges are the tangents $\mathbf{t}_0 = \frac{\mathbf{e}_0}{\|\mathbf{e}_0\|}$ and $\mathbf{t}_1 = \frac{\mathbf{e}_1}{\|\mathbf{e}_1\|}$. The angle between the two edges is $\theta = \text{acos}\left(\mathbf{t}_0^\top\mathbf{t}_1\right)$. The binormal and its surrounding quantities are denoted $\mathbf{b} = \mathbf{t}_0 \times \mathbf{t}_1$, $\mathbf{b}_0 = \mathbf{t}_0 \times \frac{\mathbf{b}}{\|\mathbf{b}\|}$ and $\mathbf{b}_1 = \mathbf{t}_1 \times \frac{\mathbf{b}}{\|\mathbf{b}\|}$.

We will now determine how the addition of the $\frac{\partial\theta}{\partial\mathbf{T}}$ update vector shifts these expressions. The stages of the BNS are as follows:

- Build a normalized version of the update vector, $\widehat{\frac{\partial\theta}{\partial\mathbf{T}}}$
- Project $\widehat{\frac{\partial\theta}{\partial\mathbf{T}}}$ onto the existing eigenvectors, $\mathbf{q}_{0\ldots5}^\theta$.
- Form the *secular equation* and solve for its roots to determine the new eigenvalues.
- Use the new eigenvalues to solve for the new eigenvectors.

Traditionally, these stages are all performed numerically, but we will see that the specific system we are examining contains enough structure that the algorithm can be applied analytically.

We will first solve for the simpler rank-one updated eigensystem

$$\mathbf{A} = \frac{\partial\theta}{\partial\mathbf{T}}\frac{\partial\theta}{\partial\mathbf{T}}^\top + \frac{\partial^2\theta}{\partial\mathbf{T}^2}. \tag{8}$$

Once this eigensystem is known, re-introducing the $\frac{\partial^2\Psi_b}{\partial\theta^2}$ and $\frac{\partial\Psi_b}{\partial\theta}$ scalars is straightforward.

1.1 Normalizing the Update Vector

The update vector is:

$$\frac{\partial\theta}{\partial\mathbf{T}} = \frac{-1}{\sin\theta}\left[\begin{array}{c|c}\mathbf{t}_1 - \mathbf{t}_0\cos\theta & \mathbf{t}_0 - \mathbf{t}_1\cos\theta\end{array}\right]. \tag{9}$$

Each column already has unit magnitude, e.g. the first column reduces as follows:

$$\frac{1}{\sin^2\theta}\left(\mathbf{t}_1 - \mathbf{t}_0\cos\theta\right)^\top\left(\mathbf{t}_1 - \mathbf{t}_0\cos\theta\right) =$$

$$\frac{1}{\sin^2\theta}\left(\mathbf{t}_1^\top\mathbf{t}_1 - \mathbf{t}_0^\top\mathbf{t}_1\cos\theta - \mathbf{t}_1^\top\mathbf{t}_0\cos\theta + \mathbf{t}_0^\top\mathbf{t}_0\cos^2\theta\right) =$$

$$\frac{1}{\sin^2\theta}\left(1 - 2\cos^2\theta + \cos^2\theta\right) = \frac{\sin^2\theta}{\sin^2\theta} = 1.$$

The normalized update vector is then straightforward:

$$\widehat{\frac{\partial\theta}{\partial\mathbf{T}}} = \frac{-1}{\sqrt{2}\sin\theta}\left[\begin{array}{c|c}\mathbf{t}_1 - \mathbf{t}_0\cos\theta & \mathbf{t}_0 - \mathbf{t}_1\cos\theta\end{array}\right]. \tag{10}$$

1.2 Projection Onto $\mathbf{q}_{0\ldots5}^\theta$

The update vector $\widehat{\frac{\partial\theta}{\partial\mathbf{T}}}$ is orthogonal to $\mathbf{q}_0^\theta$ and $\mathbf{q}_1^\theta$. Both eigenvectors are constructed entirely from $\mathbf{b} = \mathbf{t}_0 \times \mathbf{t}_1$, which is by definition orthogonal to $\mathbf{t}_0$ and $\mathbf{t}_1$. However, $\widehat{\frac{\partial\theta}{\partial\mathbf{T}}}$ is composed entirely of linear combinations of $\mathbf{t}_0$ and $\mathbf{t}_1$, so it will have zero projections onto $\mathbf{b}$, and therefore $\mathbf{q}_0^\theta$ and $\mathbf{q}_1^\theta$.

We next flatten $\widehat{\frac{\partial\theta}{\partial\mathbf{T}}}$ into a vector, $\text{vec}\left(\widehat{\frac{\partial\theta}{\partial\mathbf{T}}}\right) = \widehat{\frac{\partial\theta}{\partial\mathbf{t}}}$, and perform a projection onto $\mathbf{q}_2^\theta$:

$$\left(\mathbf{q}_2^\theta\right)^\top\widehat{\frac{\partial\theta}{\partial\mathbf{t}}} = \frac{-1}{\sqrt{2^2}\sin\theta}\begin{bmatrix}\mathbf{t}_0 + \mathbf{t}_0\times\mathbf{b}\\ \mathbf{0}_3\end{bmatrix}^\top\begin{bmatrix}\mathbf{t}_1 - \mathbf{t}_0\cos\theta\\ \mathbf{t}_0 - \mathbf{t}_1\cos\theta\end{bmatrix}$$

$$= \frac{-1}{2\sin\theta}\left(\mathbf{t}_0^\top\mathbf{t}_1 + (\mathbf{t}_0\times\mathbf{b})^\top\mathbf{t}_1 - \mathbf{t}_0^\top\mathbf{t}_0\cos\theta - (\mathbf{t}_0\times\mathbf{b})^\top\mathbf{t}_0\cos\theta\right)$$

$$= \frac{-1}{2\sin\theta}\left(\cos\theta - \sin^2\theta - \cos\theta\right)$$

*Joint first authors

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

$$= -\frac{\sin\theta}{2}$$

where we used the identities $(\mathbf{t}_0 \times \mathbf{b})^\top \mathbf{t}_1 = -\sin^2\theta$ and $(\mathbf{t}_0 \times \mathbf{b})^\top \mathbf{t}_0 = 0$. The rest of the projections follow analogously:

$$\left(\mathbf{q}_3^\theta\right)^\top \frac{\widehat{\partial\theta}}{\partial \mathbf{t}} = \frac{\sin\theta}{2} \tag{11}$$

$$\left(\mathbf{q}_4^\theta\right)^\top \frac{\widehat{\partial\theta}}{\partial \mathbf{t}} = -\frac{\sin\theta}{2} \tag{12}$$

$$\left(\mathbf{q}_5^\theta\right)^\top \frac{\widehat{\partial\theta}}{\partial \mathbf{t}} = \frac{\sin\theta}{2} \tag{13}$$

1.3 Building the Secular Equation

We can now use the projections to build the secular equation and solve for the updated eigenvalues:

$$1 + \left\|\frac{\partial\theta}{\partial \mathbf{T}}\right\|^2 \sum_{i=0}^{5} \left(\frac{\left(\left(\mathbf{q}_3^\theta\right)^\top \frac{\widehat{\partial\theta}}{\partial \mathbf{t}}\right)^2}{\lambda_i - \lambda} \right) = 0 \tag{14}$$

As the projection onto the first two eigenvectors was zero, the summation is effectively over $i = 2 \ldots 5$. Keeping in mind that the other eigenvalues were

$$\lambda_2^\theta = -1 \qquad \lambda_3^\theta = 1 \qquad \lambda_4^\theta = -1 \qquad \lambda_5^\theta = 1, \tag{15}$$

the secular equation becomes

$$1 + \left\|\frac{\partial\theta}{\partial \mathbf{T}}\right\|^2 \left(\frac{\sin^2\theta}{4(-1-\lambda)} + \frac{\sin^2\theta}{4\,(1-\lambda)} + \frac{\sin^2\theta}{4(-1-\lambda)} + \frac{\sin^2\theta}{4\,(1-\lambda)} \right) =$$

$$1 + \sin^2\theta \left\|\frac{\partial\theta}{\partial \mathbf{T}}\right\|^2 \left(\frac{1}{2\,(1-\lambda)} - \frac{1}{2(1+\lambda)} \right) = 0$$

We have now reduced what first appeared to be a 6th order polynomial to a quadratic whose roots can be written in closed form:

$$\lambda_{\text{new}} = \frac{1}{2}\left(\alpha \pm \sqrt{\alpha^2 + 4}\right)$$

$$\alpha = \sin^2\theta \left\|\frac{\partial\theta}{\partial \mathbf{T}}\right\|^2 = 2\sin^2\theta.$$

In the end, only two eigenvalues are shifted by the rank-one update.

1.4 Updating the Eigenvectors

The updated eigenvectors can now be obtained using

$$\mathbf{q}_{\text{new}} = \mathbf{Q}^\theta \left(\Lambda - \lambda_{\text{new}}\right)^{-1} \left(\mathbf{Q}^\theta\right)^\top \frac{\widehat{\partial\theta}}{\partial \mathbf{T}} \tag{16}$$

where $\mathbf{Q}^\theta$ denotes the matrix of eigenvectors $\mathbf{q}_{0\ldots5}^\theta$. We already computed the rightmost product in §1.2:

$$\left(\mathbf{q}_0^\theta\right)^\top \frac{\widehat{\partial\theta}}{\partial \mathbf{t}} = 0 \qquad \left(\mathbf{q}_1^\theta\right)^\top \frac{\widehat{\partial\theta}}{\partial \mathbf{t}} = 0 \qquad \left(\mathbf{q}_2^\theta\right)^\top \frac{\widehat{\partial\theta}}{\partial \mathbf{t}} = -\frac{\sin\theta}{2}$$

$$\left(\mathbf{q}_3^\theta\right)^\top \frac{\widehat{\partial\theta}}{\partial \mathbf{t}} = \frac{\sin\theta}{2} \qquad \left(\mathbf{q}_4^\theta\right)^\top \frac{\widehat{\partial\theta}}{\partial \mathbf{t}} = -\frac{\sin\theta}{2} \qquad \left(\mathbf{q}_5^\theta\right)^\top \frac{\widehat{\partial\theta}}{\partial \mathbf{t}} = \frac{\sin\theta}{2}$$

so we can then build:

$$\left(\mathbf{Q}^\theta\right)^\top \frac{\widehat{\partial\theta}}{\partial \mathbf{t}} = \frac{\sin\theta}{2} \begin{bmatrix} 0 \\ 0 \\ -1 \\ 1 \\ -1 \\ 1 \end{bmatrix} \tag{17}$$

and

$$\left(\Lambda - \lambda_{\text{new}}\right)^{-1} = \begin{bmatrix} 0 & & & & & \\ & 0 & & & & \\ & & \frac{-1}{1+\lambda_{\text{new}}} & & & \\ & & & \frac{1}{1-\lambda_{\text{new}}} & & \\ & & & & \frac{-1}{1+\lambda_{\text{new}}} & \\ & & & & & \frac{1}{1-\lambda_{\text{new}}} \end{bmatrix} \tag{18}$$

where zeros appear if the eigenvalues did not participate in the secular equation. We then arrive at

$$\left(\Lambda - \lambda_{\text{new}}\right)^{-1} \left(\mathbf{Q}^\theta\right)^\top \frac{\widehat{\partial\theta}}{\partial \mathbf{t}} = \frac{\sin\theta}{2} \begin{bmatrix} 0 \\ 0 \\ \frac{1}{1+\lambda_{\text{new}}} \\ \frac{1}{1-\lambda_{\text{new}}} \\ \frac{1}{1+\lambda_{\text{new}}} \\ \frac{1}{1-\lambda_{\text{new}}} \end{bmatrix} \tag{19}$$

Left multiplying by $\mathbf{Q}^\theta$ gives us the new eigenvectors:

$$\mathbf{q}_{\text{new}} = \frac{1}{1+\lambda_{\text{new}}}\left(\mathbf{q}_2^\theta + \mathbf{q}_4^\theta\right) + \frac{1}{1-\lambda_{\text{new}}}\left(\mathbf{q}_3^\theta + \mathbf{q}_5^\theta\right) \tag{20}$$

We drop the $\frac{\sin\theta}{2}$ factor since it disappears after normalization. We can further simplify by multiplying through by $(1+\lambda_{\text{new}})(1-\lambda_{\text{new}})$:

$$\mathbf{q}_{\text{new}} = (1-\lambda_{\text{new}})\left(\mathbf{q}_2^\theta + \mathbf{q}_4^\theta\right) + (1+\lambda_{\text{new}})\left(\mathbf{q}_3^\theta + \mathbf{q}_5^\theta\right) \tag{21}$$

Plugging in the expressions from $\mathbf{q}_i^\theta$ yields the final expression:

$$\mathbf{q}_{\text{new}} = \begin{bmatrix} \mathbf{t}_0 - \lambda_{\text{new}}\mathbf{b}_0 \\ \mathbf{t}_1 + \lambda_{\text{new}}\mathbf{b}_1 \end{bmatrix} \tag{22}$$

More concretely, if we assign the new eigenpairs the indices 2 and 3, they take the form:

$$\lambda_2 = \frac{1}{2}\left(\alpha + \sqrt{\alpha^2+4}\right) \qquad \mathbf{q}_2 = \begin{bmatrix} \mathbf{t}_0 - \lambda_2\mathbf{b}_0 \\ \mathbf{t}_1 + \lambda_2\mathbf{b}_1 \end{bmatrix} \tag{23}$$

$$\lambda_3 = \frac{1}{2}\left(\alpha - \sqrt{\alpha^2+4}\right) \qquad \mathbf{q}_3 = \begin{bmatrix} \mathbf{t}_0 - \lambda_3\mathbf{b}_0 \\ \mathbf{t}_1 + \lambda_3\mathbf{b}_1 \end{bmatrix} \tag{24}$$

The last two updated eigenpairs then span the remaining subspace:

$$\lambda_4 = 1 \qquad \mathbf{q}_4 = \frac{1}{2}\left(\mathbf{q}_2^\theta - \mathbf{q}_4^\theta\right) \tag{25}$$

$$\lambda_5 = -1 \qquad \mathbf{q}_5 = \frac{1}{2}\left(\mathbf{q}_3^\theta - \mathbf{q}_5^\theta\right) \tag{26}$$

Plugging in the $\mathbf{q}_i^\theta$ expressions then yields:

$$\lambda_4 = 1 \qquad \mathbf{q}_4 = \begin{bmatrix} \mathbf{t}_0 - \mathbf{b}_0 \\ -\mathbf{t}_1 + \mathbf{b}_1 \end{bmatrix} \tag{27}$$

$$\lambda_5 = -1 \qquad \mathbf{q}_5 = \begin{bmatrix} \mathbf{t}_0 + \mathbf{b}_0 \\ -\mathbf{t}_1 - \mathbf{b}_1 \end{bmatrix} \tag{28}$$

Bringing everything together, the simplified system

$$\mathbf{A} = \frac{\partial\theta}{\partial\mathbf{T}}\frac{\partial\theta}{\partial\mathbf{T}}^{\top} + \frac{\partial^2\theta}{\partial\mathbf{T}^2}. \tag{29}$$

has the analytic eigendecomposition

$$\lambda_0 = \lambda_0^\theta \qquad \mathbf{q}_0 = \mathbf{q}_0^\theta \tag{30}$$

$$\lambda_1 = \lambda_1^\theta \qquad \mathbf{q}_1 = \mathbf{q}_1^\theta \tag{31}$$

$$\lambda_2 = \sin^2\theta + \sqrt{\sin^4\theta + 1} \qquad \mathbf{q}_2 = \begin{bmatrix}\mathbf{t}_0 - \lambda_2\mathbf{b}_0\\ \mathbf{t}_1 + \lambda_2\mathbf{b}_1\end{bmatrix} \tag{32}$$

$$\lambda_3 = \sin^2\theta - \sqrt{\sin^4\theta + 1} \qquad \mathbf{q}_3 = \begin{bmatrix}\mathbf{t}_0 - \lambda_3\mathbf{b}_0\\ \mathbf{t}_1 + \lambda_3\mathbf{b}_1\end{bmatrix} \tag{33}$$

$$\lambda_4 = 1 \qquad \mathbf{q}_4 = \begin{bmatrix}\mathbf{t}_0 + \mathbf{b}_0\\ -\mathbf{t}_1 + \mathbf{b}_1\end{bmatrix} \tag{34}$$

$$\lambda_5 = -1 \qquad \mathbf{q}_5 = \begin{bmatrix}\mathbf{t}_0 - \mathbf{b}_0\\ -\mathbf{t}_1 - \mathbf{b}_1\end{bmatrix}, \tag{35}$$

where the first two eigenpairs remained unchanged from $\frac{\partial^2\theta}{\partial\mathbf{T}^2}$.

1.5 The Full Rank-One Updated System

We now return to the full rank-one updated system

$$\frac{\partial^2\Psi_b}{\partial\mathbf{T}^2} = \frac{\partial^2\Psi_b}{\partial\theta^2}\frac{\partial\theta}{\partial\mathbf{T}}\frac{\partial\theta}{\partial\mathbf{T}}^{\top} + \frac{\partial\Psi_b}{\partial\theta}\frac{\partial^2\theta}{\partial\mathbf{T}^2}. \tag{36}$$

The eigenvalues of the blue term are now shifted from ±1 to $\pm\frac{\partial\Psi_b}{\partial\theta}$, and the coefficient of the red term must be taken into account, so the magnitude of the rank-one update becomes $\sqrt{\frac{\partial^2\Psi_b}{\partial\theta^2}}\left\|\frac{\partial\theta}{\partial\mathbf{T}}\right\|$.

These new scalars does not change the overall analysis, because the two key components remain the same: the projections onto $\mathbf{q}_0^\theta$ and $\mathbf{q}_1^\theta$ remain zero, and the repeated (albeit scaled) eigenvalues reduce the 6th order secular to a quadratic. Following the same steps as before yields a system where various terms have been scaled by $\frac{\partial^2\Psi_b}{\partial\theta^2}$ and $\frac{\partial\Psi_b}{\partial\theta}$, but the overall form remains the same. The complete eigensystem is:

$$\lambda_0 = \frac{\partial\Psi_b}{\partial\theta}\lambda_0^\theta \qquad \mathbf{q}_0 = \mathbf{q}_0^\theta \tag{37}$$

$$\lambda_1 = \frac{\partial\Psi_b}{\partial\theta}\lambda_1^\theta \qquad \mathbf{q}_1 = \mathbf{q}_1^\theta \tag{38}$$

$$\lambda_2 = \frac{\partial^2\Psi_b}{\partial\theta^2} + \sqrt{\frac{\partial^2\Psi_b}{\partial\theta^2}^2 + \frac{\partial\Psi_b}{\partial\theta}^2} \qquad \mathbf{q}_2 = \begin{bmatrix}\frac{\partial\Psi_b}{\partial\theta}\mathbf{t}_0 - \lambda_2\mathbf{b}_0\\ \frac{\partial\Psi_b}{\partial\theta}\mathbf{t}_1 + \lambda_2\mathbf{b}_1\end{bmatrix} \tag{39}$$

$$\lambda_3 = \frac{\partial^2\Psi_b}{\partial\theta^2} - \sqrt{\frac{\partial^2\Psi_b}{\partial\theta^2}^2 + \frac{\partial\Psi_b}{\partial\theta}^2} \qquad \mathbf{q}_3 = \begin{bmatrix}\frac{\partial\Psi_b}{\partial\theta}\mathbf{t}_0 - \lambda_3\mathbf{b}_0\\ \frac{\partial\Psi_b}{\partial\theta}\mathbf{t}_1 + \lambda_3\mathbf{b}_1\end{bmatrix} \tag{40}$$

$$\lambda_4 = \frac{\partial\Psi_b}{\partial\theta} \qquad \mathbf{q}_4 = \begin{bmatrix}\mathbf{t}_0 + \mathbf{b}_0\\ -\mathbf{t}_1 + \mathbf{b}_1\end{bmatrix} \tag{41}$$

$$\lambda_5 = -\frac{\partial\Psi_b}{\partial\theta} \qquad \mathbf{q}_5 = \begin{bmatrix}\mathbf{t}_0 - \mathbf{b}_0\\ -\mathbf{t}_1 - \mathbf{b}_1\end{bmatrix}. \tag{42}$$

This concludes the bending energy analysis.

2 MATLAB IMPLEMENTATION AND VERIFICATION

Given the complexity of the prior derivations, it is natural to ask: *how do we know they are correct?* To provide further evidence of eigensystem correctness, we have provided Matlab/Octave implementations that pass finite difference convergence tests when compared against the original energies.

2.1 Implementation of θ

We have implemented both the acos and atan2 versions of θ:

```
function [final] = Theta_Psi(T)
  u = [1 0]';
  v = [0 1]';

  t0 = T(:,1) / norm(T(:,1));
  t1 = T(:,2) / norm(T(:,2));

  x = dot(t0,t1);
  y = norm(cross(t0,t1));

  final = acos(x);

  % can also do atan2 here too, results are the same
  %final = atan2(y,x);
end
```

We also provide its gradient:

```
function [P] = Theta_PK1(T)
  u = [1 0]';
  v = [0 1]';
  I6 = (T * u)' * (T * v);
  I5u = (T * u)' * (T * u);
  I5v = (T * v)' * (T * v);

  anti = [0 1; 1 0];
  uut = [1 0; 0 0];
  vvt = [0 0; 0 1];

  alpha = I6 / (sqrt(I5u) * sqrt(I5v));
  P = (1 / (sqrt(I5u) * sqrt(I5v)) * T * anti - ...
      I6 * (1 / (sqrt(I5v) * I5u^1.5) * T * uut + 1 / (sqrt(I5u) *
       I5v^1.5) * T * vvt));

  P = (-1.0 / sqrt(1 - alpha * alpha)) * P;
end
```

Finally, we provide our analytic eigensystem (Eqns. 2-7):

```
function [H] = Theta_Hessian_Analytic(T)
  t0 = T(:,1);
  t1 = T(:,2);
  b = cross(t0,t1);
  b0 = cross(t0,b / norm(b));
  b1 = cross(t1,b / norm(b));

  cosTheta = dot(t0,t1);

  lambdas = zeros(6,1);
  lambdas(1) = (cosTheta - 1) / norm(b);
  lambdas(2) = (cosTheta + 1) / norm(b);
  lambdas(3) = -1;
  lambdas(4) = 1;
  lambdas(5) = -1;
  lambdas(6) = 1;

  z3 = zeros(3,1);
  q0 = (1 / (sqrt(2) * norm(b))) * [b; b];
  q1 = (1 / (sqrt(2) * norm(b))) * [b; -b];
  q2 = 1 / sqrt(2) * [t0 + b0; z3];
  q3 = 1 / sqrt(2) * [t0 - b0; z3];
  q4 = 1 / sqrt(2) * [z3; t1 - b1];
  q5 = 1 / sqrt(2) * [z3; t1 + b1];

  Q = [q0 q1 q2 q3 q4 q5];

  H = Q * diag(lambdas) * Q';
end
```

Running the script `Verify_Theta.m` performs a finite difference verification of θ against $\frac{\partial \theta}{\partial \mathbf{T}}$, and then $\frac{\partial \theta}{\partial \mathbf{T}}$ against $\frac{\partial^2 \theta}{\partial \mathbf{T}^2}$. A randomized $\mathbf{T}$ matrix is generated every time, so running the script repeatedly verifies against new inputs.

2.2 Implementation of Ψ_b

We provide equivalent implementations of Ψ_b:

```
function [final] = Bending_Psi(T)
  mu = 1;
  theta0 = pi / 2;

  u = [1 0]';
  v = [0 1]';
  I6 = (T * u)' * (T * v);
  I5u = (T * u)' * (T * u);
  I5v = (T * v)' * (T * v);

  theta = acos(I6 / (sqrt(I5u) * sqrt(I5v)));
  final = (mu / 2) * (theta - theta0)^2;
end
```

We also provide its gradient:

```
function [P] = Bending_PK1(T)
  mu = 1;
  theta0 = pi / 2;

  u = [1 0]';
  v = [0 1]';
  I6 = (T * u)' * (T * v);
  I5u = (T * u)' * (T * u);
  I5v = (T * v)' * (T * v);

  anti = [0 1; 1 0];
  uut = [1 0; 0 0];
  vvt = [0 0; 0 1];

  alpha = I6 / (sqrt(I5u) * sqrt(I5v));
  thetaPK1 = (1 / (sqrt(I5u) * sqrt(I5v)) * T * anti - ...
      I6 * (1 / (sqrt(I5v) * I5u^1.5) * T * uut + 1 / (sqrt(I5u) *
       I5v^1.5) * T * vvt));

  thetaPK1 = (-1.0 / sqrt(1 - alpha * alpha)) * thetaPK1;

  theta = acos(I6 / (sqrt(I5u) * sqrt(I5v)));
  P = mu * (theta - theta0) * thetaPK1;
end
```

And finally, its analytic eigensystem (Eqns. 37-42):

```
function [H] = Bending_Hessian_Analytic(T)
  mu = 1;
  theta0 = pi / 2;

  t0 = T(:,1);
  t1 = T(:,2);
  b = cross(t0,t1);
  b0 = cross(t0,b / norm(b));
  b1 = cross(t1,b / norm(b));

  cosTheta = dot(t0,t1);
  theta = acos(cosTheta);

  dTheta = mu * (theta - theta0);
  ddTheta = mu;

  diffSq = (theta - theta0) * (theta - theta0);
  lambdas = zeros(6,1);
  lambdas(1) = dTheta * (cosTheta - 1) / norm(b);
  lambdas(2) = dTheta * (cosTheta + 1) / norm(b);
  lambdas(3) = mu * (1 + sqrt(1 + diffSq));
  lambdas(4) = mu * (1 - sqrt(1 + diffSq));
  lambdas(5) = -dTheta;
  lambdas(6) = dTheta;

  z3 = zeros(3,1);
  q0 = (1 / (sqrt(2) * norm(b))) * [b; b];
  q1 = (1 / (sqrt(2) * norm(b))) * [b; -b];
  q2 = [dTheta * t0 - lambdas(3) * b0;
        dTheta * t1 + lambdas(3) * b1];
  q3 = [dTheta * t0 - lambdas(4) * b0;
        dTheta * t1 + lambdas(4) * b1];
  q4 = [t0 + b0; -t1 + b1];
  q5 = [t0 - b0; -t1 - b1];

  q2 = q2 / norm(q2);
  q3 = q3 / norm(q3);
  q4 = q4 / norm(q4);
  q5 = q5 / norm(q5);

  Q = [q0 q1 q2 q3 q4 q5];

  H = Q * diag(lambdas) * Q';
end
```

Similar to the θ case, the script `Verify_Bending.m` performs a finite difference verification of Ψ_b against $\frac{\partial \Psi_b}{\partial \mathbf{T}}$, and then $\frac{\partial \Psi_b}{\partial \mathbf{T}}$ against $\frac{\partial^2 \Psi_b}{\partial \mathbf{T}^2}$. Again, a randomized $\mathbf{T}$ matrix is generated with every run.

3 GRADIENT OF TWISTING W

To review from the main document, our twisting energy is defined using edges between the vertices,

$$\mathbf{e}_0 = \mathbf{x}_0 - \mathbf{x}_1 \qquad \mathbf{e}_1 = \mathbf{x}_2 - \mathbf{x}_1 \qquad \mathbf{e}_2 = \mathbf{x}_3 - \mathbf{x}_2. \tag{43}$$

which are then projected onto the plane orthogonal to $\mathbf{e}_1$:

$$\mathbf{e}_{0\perp} = \mathbf{e}_0 - \frac{\mathbf{e}_0^\top \mathbf{e}_1}{\|\mathbf{e}_1\|^2}\mathbf{e}_1 \qquad \mathbf{e}_{2\perp} = \mathbf{e}_2 - \frac{\mathbf{e}_2^\top \mathbf{e}_1}{\|\mathbf{e}_1\|^2}\mathbf{e}_1. \tag{44}$$

We then assemble a deformation gradient-like matrix

$$\mathbf{W} = \left[\begin{array}{c|c|c} & & \\ \mathbf{e}_{0\perp} & \mathbf{e}_{2\perp} & \mathbf{e}_1 \\ & & \end{array}\right] \in \mathfrak{R}^{3\times 3}. \tag{45}$$

We perform our eigenanalysis with respect to $\mathbf{W}$, not the original $\mathbf{x}_i$ position variables. Therefore, a change-of-basis matrix $\frac{\partial \mathbf{w}}{\partial \mathbf{x}}$ must be derived, where $\mathbf{w} = \mathrm{vec}(\mathbf{W})$. In the main document, we write this as

$$\frac{\partial \mathbf{w}}{\partial \mathbf{x}} = \begin{bmatrix} 1 & \mathbf{e}_0^\top \mathbf{e}_1/\|\mathbf{e}_1\|^2 - 1 & -\mathbf{e}_0^\top \mathbf{e}_1/\|\mathbf{e}_1\|^2 & 0 \\ 0 & \mathbf{e}_2^\top \mathbf{e}_1/\|\mathbf{e}_1\|^2 & -\mathbf{e}_2^\top \mathbf{e}_1/\|\mathbf{e}_1\|^2 - 1 & 1 \\ 0 & -1 & 1 & 0 \end{bmatrix} \otimes \mathbf{I}_3, \tag{46}$$

but eagle-eyed readers will notice that we seem to have forgotten to differentiate the $\mathbf{e}_0^\top \mathbf{e}_1/\|\mathbf{e}_1\|^2$ and $\mathbf{e}_2^\top \mathbf{e}_1/\|\mathbf{e}_1\|^2$ terms. To the contrary, we will show that the terms arising from these additional derivatives resolve to zero. The full derivative is

$$\frac{\partial \widehat{\mathbf{w}}}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial \mathbf{e}_{0\perp}}{\partial \mathbf{x}_0} & \frac{\partial \mathbf{e}_{0\perp}}{\partial \mathbf{x}_1} & \frac{\partial \mathbf{e}_{0\perp}}{\partial \mathbf{x}_2} & \frac{\partial \mathbf{e}_{0\perp}}{\partial \mathbf{x}_3} \\ \frac{\partial \mathbf{e}_{2\perp}}{\partial \mathbf{x}_0} & \frac{\partial \mathbf{e}_{2\perp}}{\partial \mathbf{x}_1} & \frac{\partial \mathbf{e}_{2\perp}}{\partial \mathbf{x}_2} & \frac{\partial \mathbf{e}_{2\perp}}{\partial \mathbf{x}_3} \\ \frac{\partial \mathbf{e}_1}{\partial \mathbf{x}_0} & \frac{\partial \mathbf{e}_1}{\partial \mathbf{x}_1} & \frac{\partial \mathbf{e}_1}{\partial \mathbf{x}_2} & \frac{\partial \mathbf{e}_1}{\partial \mathbf{x}_3} \end{bmatrix}$$

where the block terms work out to

$$\frac{\partial \mathbf{e}_{0\perp}}{\partial \mathbf{x}_0} = \frac{\partial \mathbf{e}_{2\perp}}{\partial \mathbf{x}_3} = \mathbf{I} - \frac{\mathbf{e}_1 \mathbf{e}_1^\top}{\|\mathbf{e}_1\|^2}$$

$$\frac{\partial \mathbf{e}_{0\perp}}{\partial \mathbf{x}_1} = (\alpha - 1)\,\mathbf{I} - \frac{\mathbf{e}_1}{\|\mathbf{e}_1\|^2}((2\alpha - 1)\mathbf{e}_1 - \mathbf{e}_0)^\top$$

$$\frac{\partial \mathbf{e}_{0\perp}}{\partial \mathbf{x}_2} = -\alpha \mathbf{I} - \frac{\mathbf{e}_1}{\|\mathbf{e}_1\|^2}(\mathbf{e}_0 - 2\alpha \mathbf{e}_1)^\top$$

$$\frac{\partial \mathbf{e}_{2\perp}}{\partial \mathbf{x}_1} = \beta \mathbf{I} - \frac{\mathbf{e}_1}{\|\mathbf{e}_1\|^2}(2\beta \mathbf{e}_1 - \mathbf{e}_2)^\top$$

$$\frac{\partial \mathbf{e}_{2\perp}}{\partial \mathbf{x}_2} = -(\beta+1)\,\mathbf{I} + \frac{\mathbf{e}_1}{\|\mathbf{e}_1\|^2}((2\beta+1)\mathbf{e}_1 - \mathbf{e}_2)^\top$$
$$\frac{\partial \mathbf{e}_{0\perp}}{\partial \mathbf{x}_3} = \frac{\partial \mathbf{e}_{2\perp}}{\partial \mathbf{x}_0} = \frac{\partial \mathbf{e}_1}{\partial \mathbf{x}_0} = \frac{\partial \mathbf{e}_1}{\partial \mathbf{x}_3} = \mathbf{0}$$
$$\frac{\partial \mathbf{e}_1}{\partial \mathbf{x}_1} = -\mathbf{I}$$
$$\frac{\partial \mathbf{e}_1}{\partial \mathbf{x}_2} = \mathbf{I}$$

and we set $\alpha = \frac{\mathbf{e}_0^\top \mathbf{e}_1}{\|\mathbf{e}_1\|^2}$, $\beta = \frac{\mathbf{e}_2^\top \mathbf{e}_1}{\|\mathbf{e}_1\|^2}$.

Comparing $\frac{\partial \widehat{\mathbf{w}}}{\partial \mathbf{x}}$ with $\frac{\partial \mathbf{w}}{\partial \mathbf{x}}$, the full derivative does indeed contain new terms. However, consider a matrix of solely these new terms, $\left(\frac{\partial \mathbf{w}}{\partial \mathbf{x}}\right)_\Delta = \frac{\partial \mathbf{w}}{\partial \mathbf{x}} - \frac{\partial \widehat{\mathbf{w}}}{\partial \mathbf{x}}$. We can show that when multiplied against the gradient of the twisting energy, these new terms all resolve to zero: $\left(\frac{\partial \mathbf{w}}{\partial \mathbf{x}}\right)_\Delta^\top \frac{\partial \Psi_t}{\partial \mathbf{w}} = \mathbf{0}$.

The vectorized gradient, $\mathrm{vec}\left(\frac{\partial \Psi_t}{\partial \mathbf{W}}\right) = \frac{\partial \Psi_t}{\partial \mathbf{w}}$ can be written:

$$\frac{\partial \Psi_t}{\partial \mathbf{w}} = \frac{\mathcal{S}\,(\det \mathbf{W})\,\mu(\tau-\tau_0)}{|\sin(\tau)|\,\|\mathbf{e}_{2\perp}\|\,\|\mathbf{e}_{0\perp}\|} \begin{pmatrix} \mathbf{e}_{2\perp} - \frac{\mathbf{e}_{2\perp}^\top \mathbf{e}_{0\perp}}{\|\mathbf{e}_{0\perp}\|^2}\mathbf{e}_{0\perp} \\ \mathbf{e}_{0\perp} - \frac{\mathbf{e}_{0\perp}^\top \mathbf{e}_{2\perp}}{\|\mathbf{e}_{2\perp}\|^2}\mathbf{e}_{2\perp} \\ \mathbf{0} \end{pmatrix}$$

and the blocks of $\left(\frac{\partial \mathbf{w}}{\partial \mathbf{x}}\right)_\Delta$ are explictly:

$$\left(\frac{\partial \mathbf{e}_{0\perp}}{\partial \mathbf{x}_0}\right)_\Delta = \left(\frac{\partial \mathbf{e}_{2\perp}}{\partial \mathbf{x}_3}\right)_\Delta = \frac{\mathbf{e}_1 \mathbf{e}_1^\top}{\|\mathbf{e}_1\|^2}$$
$$\left(\frac{\partial \mathbf{e}_{0\perp}}{\partial \mathbf{x}_1}\right)_\Delta = \frac{\mathbf{e}_1}{\|\mathbf{e}_1\|^2}((2\alpha-1)\mathbf{e}_1 - \mathbf{e}_0)^\top$$
$$\left(\frac{\partial \mathbf{e}_{0\perp}}{\partial \mathbf{x}_2}\right)_\Delta = \frac{\mathbf{e}_1}{\|\mathbf{e}_1\|^2}(\mathbf{e}_0 - 2\alpha\mathbf{e}_1)^\top$$
$$\left(\frac{\partial \mathbf{e}_{2\perp}}{\partial \mathbf{x}_1}\right)_\Delta = \frac{\mathbf{e}_1}{\|\mathbf{e}_1\|^2}(2\beta\mathbf{e}_1 - \mathbf{e}_2)^\top$$
$$\left(\frac{\partial \mathbf{e}_{2\perp}}{\partial \mathbf{x}_2}\right)_\Delta = -\frac{\mathbf{e}_1}{\|\mathbf{e}_1\|^2}((2\beta+1)\mathbf{e}_1 - \mathbf{e}_2)^\top$$
$$\left(\frac{\partial \mathbf{e}_{0\perp}}{\partial \mathbf{x}_3}\right)_\Delta = \left(\frac{\partial \mathbf{e}_{2\perp}}{\partial \mathbf{x}_0}\right)_\Delta = \left(\frac{\partial \mathbf{e}_1}{\partial \mathbf{x}_0}\right)_\Delta = \left(\frac{\partial \mathbf{e}_1}{\partial \mathbf{x}_1}\right)_\Delta = \left(\frac{\partial \mathbf{e}_1}{\partial \mathbf{x}_2}\right)_\Delta = \left(\frac{\partial \mathbf{e}_1}{\partial \mathbf{x}_3}\right)_\Delta = \mathbf{0}.$$

Every block of $\left(\frac{\partial \mathbf{w}}{\partial \mathbf{x}}\right)_\Delta^\top$ involves an outer product with $\mathbf{e}_1^\top$, while every block of $\frac{\partial \Psi_t}{\partial \mathbf{w}}$ is a linear combination of $\mathbf{e}_{0\perp}$ and $\mathbf{e}_{2\perp}$. By construction, $\mathbf{e}_1$ is orthogonal to $\mathbf{e}_{0\perp}$ and $\mathbf{e}_{2\perp}$, so all the individual blocks will evaluate to $\mathbf{0}$, and it must be that $\left(\frac{\partial \mathbf{w}}{\partial \mathbf{x}}\right)_\Delta^\top \frac{\partial \Psi_t}{\partial \mathbf{w}} = \mathbf{0}$. Therefore, omitting the terms of $\left(\frac{\partial \mathbf{w}}{\partial \mathbf{x}}\right)_\Delta$ will not change the final value of $\frac{\partial \Psi_t}{\partial \mathbf{x}}$.

4 MATLAB IMPLEMENTATION AND VERIFICATION

As with the bending energies, we provide Matlab code that verifies the results of the prior section. The verification script is as follows:

```
1  addpath('./util');
2  addpath('./Twisting');
3  materialName = sprintf('Twisting');
4
5  fprintf('============================================\n');
6  fprintf('Running numerical tests for %s\n', materialName);
7  fprintf('============================================\n');
8  fprintf('Numerically verifying dWhatdX:\n');
9  Verify_Twist_W(@Twist_W, @dWhatdX);
10 fprintf('Numerically comparing dWdX (***THIS SHOULD FAIL***)\n');
11 Verify_Twist_W(@Twist_W, @dWdX);
12 fprintf('Numerically verifying position gradient with dWhatdX:\n')
       ;
13 Verify_Position_Derivative(@Twist_Psi, @Twist_PK1, @dWhatdX);
14 fprintf('Numerically verifying position gradient with dWdX:\n');
15 Verify_Position_Derivative(@Twist_Psi, @Twist_PK1, @dWdX);
16 fprintf('Numerically verifying closeness of dWdX vs. dWhatdX in
       position derivative:\n');
17 Verify_Close_Derivatives(@Twist_PK1, @dWdX, @dWhatdX)
```

The numerical convergence test on line 11 should **fail**, as it does not correspond to the exact derivative of $\mathbf{w}$. However, when multiplied against $\frac{\partial \Psi_t}{\partial \mathbf{w}}$, both $\frac{\partial \widehat{\mathbf{w}}}{\partial \mathbf{x}}$ (line 13) and $\frac{\partial \mathbf{w}}{\partial \mathbf{x}}$ (line 15) produce the exact same result. We further numerically confirm that $\frac{\partial \widehat{\mathbf{w}}}{\partial \mathbf{x}}^\top \frac{\partial \Psi_t}{\partial \mathbf{w}} = \frac{\partial \mathbf{w}}{\partial \mathbf{x}}^\top \frac{\partial \Psi_t}{\partial \mathbf{w}}$ by subtracting them from each other on line 17. Randomized positions are generated every time, so running the script repeatedly verifies against new inputs.

5 PERFORMANCE ANALYSIS

We expand on the performance numbers reported in the main paper.

Wisps	Our Analytic Filter (mm:ss)	Numerical Filter (mm:ss)	Our Speedup
2000	01:48 (7.7%)	06:32 (21.88%)	**3.63×**
4000	03:33 (9.2%)	12:53 (27.04%)	**3.63×**
8000	07:05 (9.0%)	25:16 (24.94%)	**3.57×**

Table 1: Timings over the first 100 frames of the Hairball example, using our analytic eigenvalue filtering, and brute-force numerical clamping. We list both the cumulative running time over the 100 frames, and the percentage of the overall running time. Our Speedup is the speedup from using our analytic approach over the brute-force numerical eigedecomposition.

5.1 Numerical Eigenvalue Clamping

In lieu of our analytic filtering, we also ran a direct, numerical, per-element eigendecomposition of each term in our LC energy. As in previous works [Smith et al. 2019], we found that this can significantly increase the overall running time (Table 1).

Our approach consistently gave at least a **3.57×** speedup in this stage of system assembly. Without our approach, of all the code regions timed, numerical eigenvalue clamping became the largest. Without our analytic filter, numerical eigenvalue computation alone took almost as much time as, and sometimes more than, *the entire PCG solve*.

5.2 Single Strand Examples

Table 2 lists the running times for the single-strand kinematic test examples (§5.2.1) in the main paper. Each is quite fast and robust.

Example	DoFs	System Assembly	PCG	Total Time
Smash	403	68.60%	31.40%	0.0075s
Jitter	403	68.95%	31.05%	0.0075s
Straighten	79	77.38%	22.62%	0.0039s

Table 2: Timings are in seconds for a single $\Delta t = 1/30$ timestep of BDF-1 with 3 Newton iterations. System Assembly is time spent in matrix assembly, and PCG is time spent in the linear solver.

5.3 Block Diagonal Preconditioner

Table 3 shows the overall performance without our preconditioner. When conjugate gradient with no preconditioner is used, the number of iterations explodes by orders of magnitude. Similar explosions in iteration count for un-preconditioned systems were observed in Fei et al. [2017] (Fig. 22 in that paper). In the first timestep of the 2000 wisp simulation, our solver takes 139 iterations to converge. Without our preconditioner, conjugate gradient takes 27785 iterations, a **66.5**× difference in running time. The un-preconditioned solver takes 98.47% of the running time, and totally dominates the other simulation stages. Instead of 17 *seconds*, the first timestep takes over 18 *minutes*. Performance degrades further with the size of the system. For a 4000 wisp simulation, the speedup using our preconditioner becomes **392**×, and for 8000 wisps, **494**×.

Eigen's direct Cholesky solver also did not scale competitively. The first 10 timesteps of the 2000 wisp scene averaged 33 seconds for a Cholesky solve, a **5.76**× slowdown from our approach. The performance worsened on the 4000 wisp simulation, where Cholesky took over *1 hour*, a **482**× slowdown than our approach, and deteriorated further with 8000 wisps, which took over *5 hours*, and was slower than our approach by **1130**×.

We also ran tests with the version of Pardiso [Bollhöfer et al. 2020] in Intel MKL, and the Algebraic Multigrid preconditioner in AMGCL [Demidov 2020]. Neither performed better than Eigen's Cholesky solver, so further timings were not collected.

REFERENCES

Matthias Bollhöfer, Olaf Schenk, Radim Janalik, Steve Hamm, and Kiran Gullapalli. 2020. State-of-the-Art Sparse Direct Solvers. (2020), 3–33. https://doi.org/10.1007/978-3-030-43736-7_1

James R Bunch, Christopher P Nielsen, and Danny C Sorensen. 1978. Rank-one modification of the symmetric eigenproblem. *Numer. Math.* 31, 1 (1978), 31–48.

Denis Demidov. 2020. AMGCL – A C++ library for efficient solution of large sparse linear systems. *Software Impacts* 6 (2020), 100037. https://doi.org/10.1016/j.simpa.2020.100037

Y. Fei, H. Maia, C. Batty, C. Zheng, and E. Grinspun. 2017. A multi-scale model for simulating liquid-hair interactions. *ACM Trans. Graph.* 36, 4 (2017), 1–17.

B. Smith, F. De Goes, and T. Kim. 2019. Analytic eigensystems for isotropic distortion energies. *ACM Trans. Graph.* 38, 1 (2019), 1–15.

Wisps	DoFs	Collision Handling	System Assembly	PCG	Total Time	Cholesky (hh:mm:ss)	Our Speedup	No Precond. (hh:mm:ss)	Our Speedup
2,000	806,000	06.27s (35.4%)	05.71s (32.3%)	05.73s (32.4%)	17.7s	00:00:33	**5.76**×	00:18:40	**195**×
4,000	1,612,000	09.91s (35.9%)	9.27s (33.6%)	08.42s (30.5%)	27.6s	01:07:38	**482**×	00:55:04	**392**×
8,000	3,224,000	19.9s (34.9%)	19.3s (33.9%)	17.8s (31.2%)	57.0s	05:35:32	**1130**×	02:26:38	**494**×

Table 3: Timings are in seconds for a single $\Delta t = 1/30$ timestep of BDF-1 with 3 Newton iterations. Collision Handling includes detection and response, and System Assembly is time spent in matrix assembly, minus collision forces. PCG is time spent in the linear solver. Cholesky lists timings in (hh:mm:ss), if Cholesky is used instead of PCG. No Precond. lists timings in (hh:mm:ss) when no preconditioner is used. Our Speedup is the solver speedup using our PCG version over Cholesky or no preconditioning.