Eulerian-on-Lagrangian Cloth Simulation Errata

NICHOLAS J. WEIDNER, Texas A&M University
KYLE PIDDINGTON, California Polytechnic State University
DAVID I.W. LEVIN, The University of Toronto
SHINJIRO SUEDA, Texas A&M University

September 5, 2019

In this document, we list the changes and additions made to our EOL cloth paper [Weidner et al. 2018] in order to improve the results and stability of our described methods.

1 VERTEX DEFORMATION GRADIENT

In our formulation of generalized forces in the paper, we take and element-wise average whenever we need the deformation gradient at a vertex. If we take the numerical average of the faces around a vertex, though, we are averaging the rotation and the non-rotational components of the deformation gradient together. When the faces of a vertex are rotationally different by a significant margin, this can introduce error into the vertex deformation gradient. This occurs in practice because the surrounding faces of EOL vertices are often rotationally misaligned, creating a sharp bending edge or asperity. To avoid this error we decompose and average the deformation gradient into rotational and non-rotational components separately.

For a deformation gradient $F \in \mathbb{R}^{3\times2}$, we can use singular value decomposition to break F down into

$$USV^T = F, \tag{1}$$

where $U \in \mathbb{R}^{3\times3}$, $S \in \mathbb{R}^{3\times2}$, and $V \in \mathbb{R}^{2\times2}$. Appending a zero row onto V we create a $\bar{V} \in \mathbb{R}^{3\times2}$ matrix which can be used to extract the rotational component of F as

$$Q = U\bar{V}^T. \tag{2}$$

This Q is a $\mathbb{R}^{3\times2}$ matrix so to make it a $\mathbb{R}^{3\times3}$ rotation matrix we cross its two columns and store the result in the third column. The remaining (non-rotational) component of the deformation gradient is $P \in \mathbb{R}^{2\times2}$. To form this we need to break S down into its upper left $\mathbb{R}^{2\times2}$ matrix. This $\bar{S} \in \mathbb{R}^{2\times2}$ in Matlab notation is

$$\bar{\texttt{S}} = \texttt{S(1:2,1:2)}, \tag{3}$$

and is used to then generate

$$P = V\bar{S}V^T. \tag{4}$$

Now that we have broken down the components of a single F, we can use this break down to compute a weighted average of multiple Fs from the incident faces of a vertex. For two faces $F_a \in \mathbb{R}^{3\times2}$ and $F_b \in \mathbb{R}^{3\times2}$, we can decompose

Authors' addresses: Nicholas J. Weidner, Department of Computer Science & Engineering, Texas A&M University, College Station, TX, weidnerjn@tamu.edu; Kyle Piddington, Department of Computer Science & Software Engineering, California Polytechnic State University, San Luis Obispo, CA, kpiddy@gmail.com; David I.W. Levin, Department of Computer Science, The University of Toronto, Toronto, ON, diwlevin@cs.toronto.edu; Shinjiro Sueda, Department of Computer Science & Engineering, Texas A&M University, College Station, TX, sueda@tamu.edu.

each into

$$\begin{aligned} F_a &= Q_a P_a \\ F_b &= Q_b P_b. \end{aligned} \tag{5}$$

The rotational component $Q \in \mathbb{R}^{3\times3}$ is averaged as

$$Q = \exp\left(w_a \log\left(Q_a\right) + w_b \log\left(Q_b\right)\right), \tag{6}$$

where w is the incident angle of the corresponding face, and non-rotational part can be linearly added as

$$P = w_a P_a + w_b P_b. \tag{7}$$

Lastly the final deformation gradient F is the product of the first two columns of Q and P. Using Matlab notation again:

$$\begin{aligned} \bar{\texttt{Q}} &= \texttt{Q(:,1:2)} \\ F &= \bar{Q}P. \end{aligned} \tag{8}$$

This vertex wise deformation gradient calculation retains the energy lost from a simple element wise averaging, and allows for more realistic cloth force propagation during simulation.

2 FLAT NULL SPACE

When the cloth is in contact with an object, we construct inequality constraints in order to allow the cloth to lift away from the object without moving inside of it. There is a situation, primarily in the first few timesteps of initial contact with the object, that the cloth is locally flat around the contact point. If any of the object tangents used to generate the inequality constraints are parallel within some margin to the local tangent of the newly introduced EOL vertices, we end up with a one sided null space that allows these nodes to move indefinitely. In order to avoid this, we introduce a check into our constraint construction algorithm.

This algorithm is an augmentation of the Lagrangian constraint construction component of the algorithm discussed in the paper, and we define "locally flat" as no two faces incident to a vertex having an angle between them greater than some threshold. For the box normals $\boldsymbol{n}_1$ and $\boldsymbol{n}_2$ we follow Algorithm 1.

Algorithm 1 EOL Lagrangian Constraint Generation

for each EOL vertex v **do**
  if v colliding with box edge **then**
    if v is locally flat **then**
      Lagrangian Constraint: $\boldsymbol{n}_1^T \dot{\boldsymbol{x}} = 0,\ \boldsymbol{n}_2^T \dot{\boldsymbol{x}} = 0$
    else
      Lagrangian Constraint: $\boldsymbol{n}_1^T \dot{\boldsymbol{x}} \geq 0,\ \boldsymbol{n}_2^T \dot{\boldsymbol{x}} \geq 0$
    end if
  end if
end for

3 POINT CONSTRAINTS

We break our constraints down into two separate cases: contact with a box corner and contact with a box edge. In the case of a box corner, we constructed the Lagrangian constraint of an EOL vertex from an orthogonal frame at the

corner of the box. If we denote the box normals as $\boldsymbol{n}_1$, $\boldsymbol{n}_2$, and $\boldsymbol{n}_3$, then the Lagrangian velocity constraint is defined as $\boldsymbol{n}_1^T \dot{\boldsymbol{x}} \geq 0$, $\boldsymbol{n}_2^T \dot{\boldsymbol{x}} \geq 0$, and $\boldsymbol{n}_3^T \dot{\boldsymbol{x}} \geq 0$. This constraint originates from the fact that we do not want our EOL vertex to move into the box, and the free space described by this constraint region avoids this.

In our results we showcase EOL Cloth sliding smoothly over individual points. These points are much more restrictive than box corners because the cloth can droop around the point in every direction. In this case the only acceptable Lagrangian motion is along the point normal. To construct the Lagrangian constraints of an EOL vertex on a point, we take a point normal $\boldsymbol{n}_1$ and construct two arbitrary orthonormal vectors $\boldsymbol{n}_2$, and $\boldsymbol{n}_3$. The Lagrangian velocity constraint is then $\boldsymbol{n}_1^T \dot{\boldsymbol{x}} \geq 0$, $\boldsymbol{n}_2^T \dot{\boldsymbol{x}} = 0$, and $\boldsymbol{n}_3^T \dot{\boldsymbol{x}} = 0$. Again we do not need any constraints on the Eulerian velocity of a point EOL vertex.

REFERENCES

Nicholas J. Weidner, Kyle Piddington, David I.W. Levin, and Shinjiro Sueda. 2018. Eulerian-on-Lagrangian Cloth Simulation. *ACM Transactions on Graphics* 37, 4 (August 2018), 50:1–50:11.