Selective and Dynamic Cloth Fold Smoothing with Collision Resolution

Arunachalam Somasundaram*
DreamWorks Animation

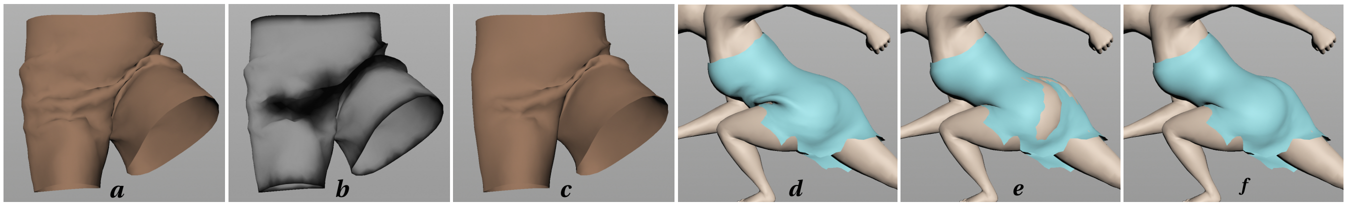

Figure 1: ***a)*** *Original cloth with folds,* ***b)*** *Larger folds and boundary detected and smooth weights w_s visualized,* ***c)*** *Smaller folds smoothed out,* ***d)*** *Original cloth with folds,* ***e)*** *Smoothed cloth with interpenetrations, and* ***f)*** *Smoothed cloth with collision resolution*

Abstract

We present techniques to selectively and dynamically detect and smooth folds in a cloth mesh after simulation. This gives artists the controls to emphasize or de-emphasize certain folds, cleanup simulation errors that can cause crumpled cloth, and resolve cloth-body interpenetrations that can happen during smoothing. These techniques are simple and fast, and help the artist to direct, cleanup, and enrich the look of simulated cloth.

Keywords: cloth, mesh, simulation, fold, smoothing, collision, interpenetration

Concepts: •**Computing methodologies** → *Physical simulation; Motion processing; Collision detection;*

1 Motivation

The dynamic folds that appear in simulated cloth may need to be selectively smoothed for artistic and technical reasons such as:

- *Simplicity and Emphasis*: Keep only the big folds to emphasize them, similar to traditional 2-D animation.
- *Fold Size Reduction*: Reduce the big folds to avoid distraction.
- *Cleanup*: Remove simulation errors such as unnecessary crumpling that can happen, for example, after the cloth gets stuck in between two overlapping collision objects.

The smoothing may produce interpenetrations with the body which need to be resolved. Given the dynamic nature of the cloth folds, without an automated process for all the above, an artist would have to manually smooth different regions over animated frames, and cleanup interpenetrations if any, all of which can be painstaking.

Techniques such as [Jones et al. 2003] are used for smoothing high frequency noise in meshes such as those that are generated from geometry scanners while preserving larger features. The techniques presented here are more suitable for smoothing and/or emphasizing dynamic features in meshes such as simulated cloth folds, while resolving interpenetrations with collision objects if needed. Methods such as [Baraff et al. 2003] try to untangle cloth pinched between collision objects as part of the simulation process, whereas techniques presented here are applied post simulation to remove unwanted cloth crumpling in simulated cloth.

2 Fold Detection

In order to selectively smooth folds, we first detect the folds based on user input. We describe two methods to detect folds. The fold detection methods calculate vertex weights w_f as a measure of a fold for that vertex. A w_f of 1.0 corresponds to that vertex being part of a cloth fold and a w_f of 0.0 corresponds to no fold.

2.1 Position Based Initial Detection

For every frame, we first smooth the entire original mesh msh_o, and create a smooth mesh msh_d internally for detection of folds. Then, for each vertex, we calculate the distance d_{od} between msh_o and msh_d. This distance can be the vector distance between the two vertices, or the distance along the vertex normal of msh_d. Any vertex whose d_{od} is greater than a user specified threshold d_{max}, is marked as belonging to a fold. This relies on the fact that relatively flat areas deform less when smoothed, while areas with folds deform more when smoothed. These initially detected vertices are given a w_f of 1.0, corresponding to being part of a cloth fold.

2.2 Angle Based Initial Detection

For angle based detection, we can use the smoothed mesh per frame or the reference (T-pose) mesh as the detection mesh msh_d. For each edge we calculate the angle between faces sharing that edge for both msh_o and msh_d, namely ang_o and ang_d. We then calculate the angular change ang_{do}, where ang_{do} = $|ang_d$ - $ang_o|$. If ang_{do} is larger than a user specified threshold ang_{max} , the vertices shared by that edge are marked as belonging to a fold, and given a w_f of 1.0. This relies on the fact that appearing folds are caused by relative changes between angles of faces shared by a fold edge.

2.3 Final Detection: Weight Spreading and Smoothing

The detected fold vertices may be a little sparse as sometimes only the peak of the folds maybe detected. So, an option to iteratively spread the fold to neighboring vertices of the sparsely detected fold vertices is provided. These neighboring fold vertices are initially marked with the same w_f as the original fold vertices themselves. An option to linearly falloff this w_f for the neighboring fold vertices is also provided. Once these vertices are all marked, the other vertices are marked with a w_f of 0.0. The fold weights are then spatially smoothed, without affecting the initially detected fold vertices. This produces a smooth falloff of the fold weights. This avoids artifacts in boundaries of smoothing areas and helps to maintain temporal continuity of dynamic smoothing across frames. These fold weights are used for selectively smoothing the mesh.

3 Selective Fold Smoothing

Once the folds are detected and fold weights w_f are calculated, we can use those weights w_f or $1.0 - w_f$ as smoothing weights w_s. A fully smoothed mesh msh_s is created based on smoothing methods such as simple or *Fujiwara* [Fujiwara 1995] weighted *Laplacian*, *Taubin* [Taubin 1995] or *Delta Mush* [Mancewicz et al. 2014], and includes boundary preservation option. w_s is used to blend between msh_o and msh_s vertices on every iteration of the smoothing process. Larger folds can be smoothed out if $w_s = w_f$ and smaller folds can be smoothed out if $w_s = 1.0 - w_f$.

4 Collision Resolution

The smoothed cloth mesh can interpenetrate the body which needs to be resolved. For each smoothing iteration, vertices colliding with the body are detected. Each colliding vertex is first pushed out to a point p_b on the surface of the body. Each colliding vertex is then offset off p_b along the direction of the initial push out and given an offset value of o_b specified by the user. The rest of the vertices receive an o_b of 0.0. The offset values o_b alone (and not the vertex positions) are then spatially smoothed. This helps to smoothly transition the collision resolved areas into the neighborhood. Smoothing the offsets can introduce minor interpenetrations with the collision object. A final collision resolution pass is performed where the colliding cloth vertices, if any still left, are pushed just out of the collision object without smoothing the minor offsets that they would have to undergo to just get out of the collision object. Since the initial collision resolution pass with smoothing of offsets would have resolved most collisions, the offsets that the colliding vertices would have to undergo in the final collision resolution pass are minor and do not need smoothing. This whole process guarantees a collision resolved smoothed mesh.

5 Implementation and Usage

This mesh smoothing technique is implemented as a custom node in a third party procedural package and is multi-threaded. The artist can control the various parameters of the node, which include the auto fold selection parameters and collision resolution parameters mentioned in the prior sections in this abstract, as well as the smoothing parameters of the different smoothing methods. In simulated cloth, folds can appear and disappear in different areas of the cloth over time. The parameters are used to automatically and dynamically select which areas of the simulated cloth need to be smoothed and smooth them accordingly. The artist can also manually restrict the auto smoothing process to affect only a section of the cloth by painting vertex weights. The node can also be simply run as a generic smoothing node without any fold detection. For cloth setups, this node is added after the cloth simulation step. The parameters are tweaked to get the appropriate folds in the garment which may correspond to large fold emphasis by smoothing smaller folds, or large folds smoothing to avoid distraction. This node can also be added in the procedural graph as and when needed by the artist to cleanup unwanted crumpling issues that can be seen after cloth simulation. Since the nature of a fold can vary per garment type and artistic direction, the artist may need to tune the parameters of the node to the corresponding garment type and artistic direction. Multiple such smoothing nodes can also be added in the procedural graph to achieve the desired accumulative result. For example, one smoothing node can be added to smooth out the smaller folds and another smoothing node can be used downstream of the procedural graph to remove the unwanted crumpling in cloth.

References

BARAFF, D., WITKIN, A., AND KASS, M. 2003. Untangling cloth. In *ACM SIGGRAPH 2003 Papers*, ACM, New York, NY, USA, SIGGRAPH '03, 862–870.

FUJIWARA, K. 1995. Eigenvalues of laplacians on a closed riemannian manifold and its nets. *Proceedings of the American Mathematical Society 123*, 8, 2585–2594.

JONES, T. R., DURAND, F., AND DESBRUN, M. 2003. Non-iterative, feature-preserving mesh smoothing. In *ACM SIGGRAPH 2003 Papers*, SIGGRAPH '03, ACM, 943–949.

MANCEWICZ, J., DERKSEN, M. L., RIJPKEMA, H., AND WILSON, C. A. 2014. Delta mush: Smoothing deformations while preserving detail. In *Proceedings of the Fourth Symposium on Digital Production*, DigiPro '14, ACM, 7–11.

TAUBIN, G. 1995. Curve and surface smoothing without shrinkage. In *Computer Vision, 1995. Proceedings., Fifth International Conference on*, 852–857.