Eurographics/ ACM SIGGRAPH Symposium on Computer Animation (2012)
P. Kry and J. Lee (Editors)

Long Range Attachments - A Method to Simulate Inextensible Clothing in Computer Games

Tae-Yong Kim, Nuttapong Chentanez and Matthias Müller-Fischer[†]

NVIDIA PhysX Research

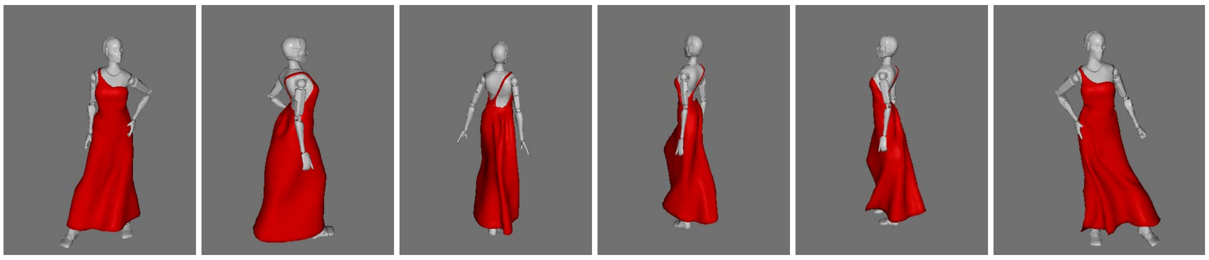

Figure 1: *Simulation of a game character with evening dress containing 1500 vertices. The simulation shown here runs under 1ms per frame on a single CPU core. Our GPU implementation runs up to 16 such characters at comparable frame rate.*

Abstract

Inextensibility is one of the most fundamental properties of cloth. Existing approaches to handle inextensibility often require solving global non-linear systems and remain computationally expensive for computer game uses. Real time performance can be achieved by allowing damping or stretching at reduced solver costs, but these compromise visual realism - the cloth either looks stretchy or fine wrinkles get lost.

Our long range attachment (LRA) method exploits that typical game character clothing tends to be attached to some kinematic parts of the character. LRA method applies unilateral distance constraint between free particles of the cloth to distant attachment point on the character, preventing them from stretching away from the kinematically driven attachments (e.g. shoulder for a cape). This simple step provides an efficient shortcut for enforcing global inextensibility that can be readily implemented into existing game physics methods such as PBD.

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling-Physically Based Modeling—

1. Introduction

The advances in visual simulation of clothing now allow the creation of life-like characters with clothing in feature films and high quality animation productions. In contrast, simulating inextensible clothing in computer games remains challenging, given the stringent performance budget allowed for game physics. Many game scenes consist of more than 10 clothed characters. The maximum time budget for game physics is typically set to about 5ms per frame, of which clothing simulation is only a fraction. Current cloth simulation methods used in games achieve the required performance by allowing cloth to stretch or by resorting to coarse meshes. However, these compromises results in reduced visual realism as well as other artifacts such as penetration of limbs through stretched cloth. Our goal is to develop a simple algorithm that handles inextensibility of typical game character clothing efficiently so that we can use high enough mesh resolution to allow more realistic wrinkle formation within the time budget of computer games.

[†] taeyongk,nchentanez,mathiasm@nvidia.com

DOI: 10.2312/SCA/SCA12/305-310

Our approach builds on the observation that in the context of game character clothing simulation, clothing is always attached to the character, and the violation of inextensibility between free particles and the attached points are main causes of the stretching artifacts. Therefore, we introduce long range attachment (LRA) constraints that limit distance from attachment points to free particles in the mesh, simulating the infinite speed of pressure waves in fully stretched, inextensible cloth. The LRA constraints are unilateral - they get activated only when cloth is stretched, and do not influence the buckling behavior of cloth such as wrinkle formation, so local waves can still occur even when the cloth is completely unfolded, yielding more vivid behavior, see Figure 2.

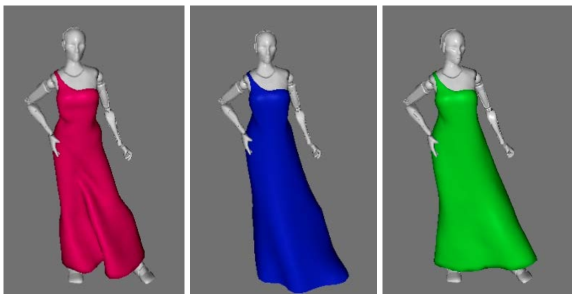
Figure 2: *Comparison of different simulation settings on the same character. Left) Our method, 5 iterations, Middle) PBD with 5 iterations, Right) PBD with 15 iterations. Note how PBD generates stretchy cloth with low iteration counts and flattens out wrinkles with higher numbers of iterations.*

2. Related Work

There exists a large body of work on cloth simulation. Here we review directly related prior work only and refer readers to [CK05, MSJT08] for a thorough survey of the field.

2.1. Handling Inextensiblity

There exist two line of techniques to handle the inextensibility of cloth. In force-based models [CK02, BMF03, GHDS03, BWH*06, VMTF09], stiff spring forces are used to avoid stretching of the cloth material. The resulting stiff systems require small time steps when explicit integrators are used. On the other hand, implicit integrators [BW98, CK02] introduce damping that results in the dissipation of wrinkles and visual detail. Although there exist techniques to mitigate the damping problem [CK02], a typical game physics time budget only allows a small number of iterations, and the resulting solver error yields stretchy and damped cloth.

In strain limiting techniques [Pro95, BFA02, TPS09], cloth material properties are simulated by enforcing constraints on nearby particles. One such constraint is the distance constraint [Jak01] that preserves the original distance between a pair of particles. In this framework, cloth particles are initially allowed to violate the constraints. Then the cloth state is projected onto the constraint manifold at each time step. Velocities are derived from the projected state and previous state, resulting in a stable system that maintains the constraints throughout the simulation.

The non-linear constraint system can be solved either by a series of global linear solves [GHF*07, EB08], or by a series of local projection steps. The position based dynamics method (PBD) [MHHR07] has recently become popular in games due to its simplicity and speed. For typical game assets made of a few thousands vertices, this technique converges reasonably well when executed in proper order (e.g. in direction of shock propagation), and thus has been adopted in many game phsyics engines. As with implicit integration, using global projection methods is computationally too expensive for our purpose. PBD is more attractive from a performance standpoint but tends to produce stretching artifacts when small numbers of local projection iterations are used.

2.2. Up-sampling methods

Recently, there has been increased interests in adding higher realism to game clothing. [KGBS11] used a linear up-sampling operator that relates a low resolution simulation to pre-computed higher resoultion simulations. [MC10] computes wrinkles on high resolution meshes, based on the relationship between compression in coarse meshes and wrinkles in finer meshes. [WHRO10] developed a data-driven method that maps a pre-computed wrinkle database to low resolution simulations based on postures of characters for tight fitting clothing. As in our case, these methods aim at improving the quality of clothing simulations within a given time budget. However, they assume wrinkle formation to be a static phenomenon, independent of the underlying dynamics. Thus, the dynamic behavior of wrinkles is typically less realistic compared to full resolution simulations.

2.3. Multi-grid and Hierarchical methods

The convergence rate of strain limiting solvers can be enhanced significantly by using hierarchical approaches [Mul08, WOR10]. However, hierarchical methods generally use more complex data structures and consume more memory. This limits parallelism and optimization for practical game engines. The simplicity of our method allows more room for parallelism and optimization for game scenarios.

3. Our Method

In typical game assets for clothing, there are particles that are attached to the character (e.g. the top particles of a cape attached to the shoulder). Even for free moving clothing, e.g. a shirt, it is common to lock some particles to reduce number of simulated particles (e.g. around the neck). Since these attached particles are kinematically driven by the character animation, they are assigned infinite mass for the simulation, and thus dominate the tensile energy propagation.

3.1. Long Range Attachment Constraints

We call our method LRA for Long Range Attachments. Let us consider a single inextensible rope. If one end is attached, it follows for all free particles that the distance between the attachment and a specific particle cannot be larger than the initial length of the path from the attachment to the specific particle. (Figure 3). Therefore the LRA method is formalized as follows. For every unconstrained particle i, we pre-compute the initial distance r_i from the particle to the attachment point. During the simulation, if the particle is within this distance, we allow it to freely move. If the particle moves beyond this limit, we project it back to the surface of a sphere centered at the attachment point with radius r_i.

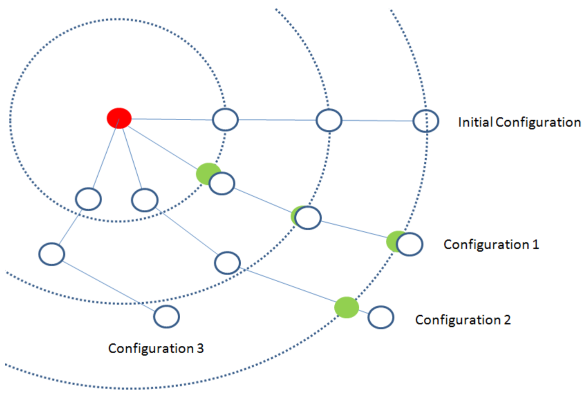

Figure 3: *Each particle is constrained inside a sphere centered at the attachment point (red) whose radius is the initial distance from the particle to the attachment. For each configuration, target positions are shown in green when particles need to be projected. Particles inside the constraint spheres are allowed to move freely.*

Enforcing this simple constraint allows tensile pressure waves to propagate immediately from the source (attachment) to all the free particles in a single step. This constraint is unilateral and applied only to particles whose path to the attachment is fully stretched. When cloth is compressed and bends (e.g. configuration 3 of Figure 3), particles move freely, allowing buckling and wrinkle formation.

Note that each LRA constraint handles stretching of the *entire path* from the attachment to a free particle, but does not prevent neighboring free particles from compressing or stretching w.r.t each other. To properly handle wrinkle formation and local stretching, we also apply the usual bilateral constraints such as stretch, shear, bending to nearby particles [MHHR07]. However, as LRA handles global stretching, we need significantly fewer iterations than usual Gauss-Seidel style iterations (Figure 2).

General character clothing is most likely attached via multiple attachment points. In this case, each LRA constraint represents a separate shock wave to free particles. For a given free particle, we can either process each constraint in order in Gauss-Seidel fashion, or average projection displacements of multiple constraints in Jacobi-style. We use the latter because it produces less biased results.

3.2. Distance Measure between points on cloth surface

When the surface is planar and convex (e.g. a flag), Euclidean distance between the attachment point and a particle accurately determines the radius of the constraint sphere. However, cloth meshes given *a priori* from artists often include pre-sculpted foldings and curved surfaces. In this case, Euclidean distance may incorrectly estimate stretching condition for tensile stiffness (Figure 4).

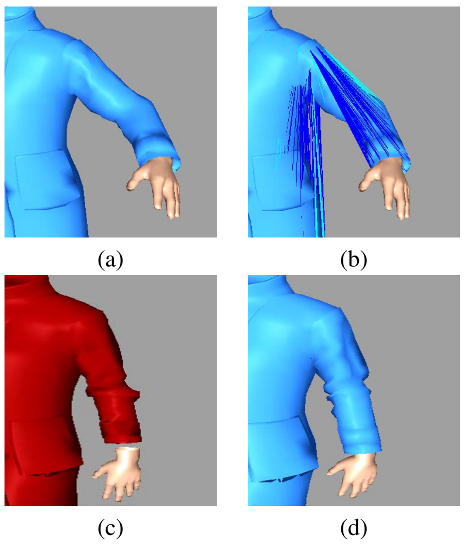

Figure 4: *(a) The lower arm is bent in the bind pose (b) Cloth particles of sleeves are attached to the shoulder (c) The Euclidean distance measure yields a constraint rest distance that is too small when the arm is straightened during a subsequent animation (d) Using the geodesic distance allows more room for unfolding.*

For general meshes, we use the shortest path from the attachment to the particle *along* the surface. This geodesic problem is well studied in graphics [SSK*05], and we use an implementation of the MMP algorithm [MMP87], available as an open source project (http://code.google.com/p/geodesic). Note that the radii of LRA constraint spheres do not need to be updated during simulation, and can be thus pre-computed.

3.3. The Simulation Algorithm

As in [MHHR07], we first apply external forces and gravity to the particles' velocities and integrate their positions in time using an explicit Euler step to get unconstrained (predicted) positions. We then perform the constraint projection step (Algorithm 1), which includes LRA as well as other local constraints. The new velocities are then derived from the previous and the projected positions. We typically use 3-5 iterations per frame. Using less iteration results in collision detection failure for typical (fast) game character motion.

Algorithm 1 Our Constraint Projection Step

```
for all constraints between particle $p_i$ and $p_j$ do
    apply local constraint projection to $p_i$ and $p_j$ [MHHR07].
end for
for all unconstrained particle $p_i$ do
    $d_i \leftarrow 0$
    $N \leftarrow$ number of LRA constraints assigned to $p_i$
    for all LRA constraint $c_{ij}$ do
        apply constraint and compute displacement $d_{ij}$
        $d_i \leftarrow d_i + d_{ij}$
    end for
    if $N \neq 0$ then
        $p_i \leftarrow p_i + d_i/N$
    end if
end for
apply collision and other constraints
```

3.4. Pruning Attachment Points

The overall time to handle the LRA constraints scales linearly with the number of attachments assigned to each particle. In practice, using a subset of the attachment points per each free particle produces good results. For example, when all the attachments are at the top (e.g. a cape attached to the shoulder), assigning only the closest attachment point to each particle works well. When multiple corner points are pinned, however, each attachment contributes significantly to the tensile energy transfer. In general, these attachments tend to form a set of connected islands (e.g. waistline for a skirt, shoulder for sleeves, a group of pinned points). We found that identifying such islands and assigning the closest attachment point from each island works well in practice. When there are many such islands, we use the N closest islands (typically, N = 4). See Algorithm 2.

3.5. Controlled Stretchiness

We provide artistic control for an additional amount of stretchines by simply inflating the radius of each constraint sphere (Figure 5). Game assets often yield over-constrained situation where there is no collision free configuration of cloth that satisfies all the LRA constraints. Allowing a small amount of stretching helps to relax the problem, mitigating visual artifacts in such cases. (Figure 6).

Algorithm 2 Assigning attachment to each particle

```
Start with an empty set of island $S$
for all attached particle $p_c$ do
    if $p_c$ is connected to any point in $S_i$, where $S_i \in S$ then
        add $p_c$ to $S_i$
    else
        create an empty set $S_i$ in $S$, and add $p_c$ to $S_i$
    end if
end for
for all unconstrained particle $p_i$ do
    $C_i \leftarrow empty$
    for all $S_i$ in $S$ do
        find closest attachement point $p_c$ in $S_i$, add it to $C_i$
    end for
    sort $C_i$ based on distance to $p_i$, and output N closest points for $p_i$.
end for
```

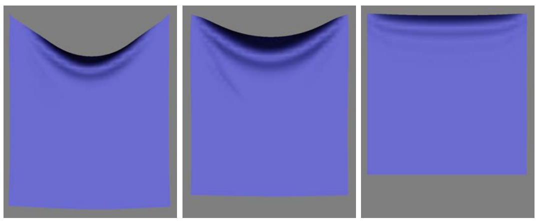
Figure 5: *Simulation of a hanging cloth (1600 vertices) under gravity. Radius for the LRA spheres was inflated by left) 20%, middle) 10%, and right) 0%, respectively. Each simulation used 2 iterations per frame.*

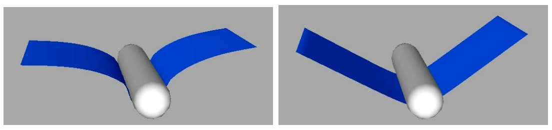
Figure 6: *Cloth is constrained at both ends, and a capsule collider forcefully push the cloth to stretch. Allowing 20% more stretching (right image) yields more realistic results.*

4. Implementation

We implemented the methods both on the CPU and GPU. We optimized the CPU implementation using SSE (4 way SIMD), and used CUDA for the GPU implementation. We store indices and distances for up to 4 attachments per each free particle. With 16 bit indexing and 16 bit quantization of the distance per attachment, we use 16 bytes per free particle. As each particle is processed independently, LRA can be efficiently implemented using both SSE and CUDA.

The LRA constraints computation consumes less than 2% of overall simulation cost.

For local constraints of the full simulation cycle (Algorithm 1), we represent stretch and shear constraints as distance constraints and use an angle based formula for bending. For the CPU implementation, we use a Gauss-Seidel (G-S) procedure that processes each constraint in serial order from one end of cloth to another, each constraint immediately using results from previous projections. For the GPU implementation, we use a red-black Gauss-Seidel (RB-GS) method to maximize parallelism. In RB-GS, each independent set of constraints is computed in parallel in two passes, interleaving odd numbered constraints with even numbered constraints to avoid conflicts in updates. In the case of bending, we use three such passes.

As common in game physics, we represent the collision volume of the character with capsules, but do not use self collision. Other constraints we implemented include a motion constraint which constrains each particle to stay within small radius of the skinned position [MC10]. All these constraints are independently processed for each particle, thus are easily parallelized.

5. Result

We tested our implementation on a machine with Intel Core i7-930 CPU and NVIDIA GeForce GTX 470 GPU. Figure 1 and 7 show typical game characters with garments made of about 1500 vertices. In our multi-threaded CPU implementation, each CPU core simulates one garment (with SSE), so the performance scales with the number of available CPU cores. We optimized the GPU implementation to handle multiple cloth instances in parallel, simulating one cloth instance per SM block. On a GTX470 with 14SMs (each SM with 32 CUDA cores), we batch process multiples of 14 characters per pass. We can therefore simulate up to 14 characters within the same time budget. With higher end cards such as GTX580, we can handle up to 16 characters per GPU pass. In the scene shown in Figure 7, the simulation of one character on one CPU core takes an average of 0.91 ms per frame. Simulating the entire scene of 14 characters (total of about 20K vertices) takes 0.8 ms per frame using a single GPU pass.

We also tested our method on high resolution meshes. For this, we used a slightly different GPU implementation which uses all the GPU cores for a single instance of cloth. With this implementation, we achieved 20fps for a cloth object with 90K (300x300) vertices using 10 iterations per frame (Figure 8).

6. Limitations and Future Work

Our method provides a simple, yet efficient solution to the common stretching problem in game character clothing. It is best suited for inextensible character clothing such as dress, cape, or coat. For unattached environmental cloth (e.g. flying papers), it does not provide benefits. As with existing methods, materials with high bending stiffness [GHDS03] need more bending constraint iterations regardless of inextensibility, a problem which we plan to work on in the future.

Our method allows higher level of realism for common game assets with meshes of a few thousands vertices. For more visual details, it may be beneficial to use up-sampling methods [MC10, KGBS11]. We currently handle the longest (LRA) and the shortest (local) range of constraints, which seems adequate for our target uses, but probably not enough for very high resolution (Figure 8). To further accelerate convergence of all the constraints at multiple spectrum, our method may be combined with hierarchical methods [Mul08, WOR10] for higher fidelity simulations as in movie industry.

References

[BFA02] Bridson R., Fedkiw R., Anderson J.: Robust treatment of collisions, contact and friction for cloth animation. *ACM Trans. Graph. 21* (July 2002), 594–603. 2

[BMF03] Bridson R., Marino S., Fedkiw R.: Simulation of clothing with folds and wrinkles. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation* (Aire-la-Ville, Switzerland, Switzerland, 2003), SCA '03, Eurographics Association, pp. 28–36. 2

[BW98] Baraff D., Witkin A.: Large steps in cloth simulation. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 1998), SIGGRAPH '98, ACM, pp. 43–54. 2

[BWH*06] Bergou M., Wardetzky M., Harmon D., Zorin D., Grinspun E.: A quadratic bending model for inextensible surfaces. In *Proceedings of the fourth Eurographics symposium on Geometry processing* (Aire-la-Ville, Switzerland, Switzerland, 2006), SGP '06, Eurographics Association, pp. 227–230. 2

[CK02] Choi K.-J., Ko H.-S.: Stable but responsive cloth. *ACM Trans. Graph. 21*, 3 (2002), 604–611. 2

[CK05] Choi K.-J., Ko H.-S.: Research problems in clothing simulation. *CAD 37* (2005), 585–592. 2

[EB08] English E., Bridson R.: Animating developable surfaces using nonconforming elements. In *ACM SIGGRAPH 2008 papers* (New York, NY, USA, 2008), SIGGRAPH '08, ACM, pp. 66:1–66:5. 2

[GHDS03] Grinspun E., Hirani A. N., Desbrun M., Schröder P.: Discrete shells. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation* (Aire-la-Ville, Switzerland, Switzerland, 2003), SCA '03, Eurographics Association, pp. 62–67. 2, 5

[GHF*07] Goldenthal R., Harmon D., Fattal R., Bercovier M., Grinspun E.: Efficient simulation of inextensible cloth. *ACM Trans. Graph. 26* (July 2007). 2

[Jak01] Jakobsen T.: Advanced character physics. Game Developer Conference. 2

[KGBS11] Kavan L., Gerszewski D., Bargteil A. W., Sloan P.-P.: Physics-inspired upsampling for cloth simulation in games. *ACM Trans. Graph. 30* (Aug. 2011), 93:1–93:10. 2, 5

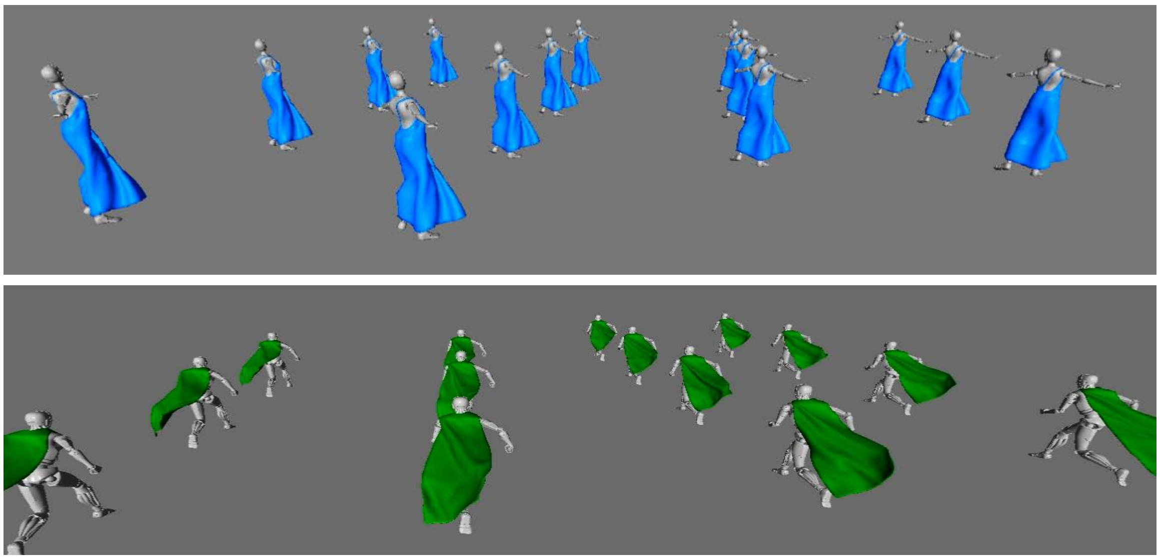

Figure 7: *Simulation of game characters on a GPU. Each character's clothing is simulated in parallel and the entire simulation is completed under 1ms/frame with our GPU impelementation. We use 5 iterations per frame at 60Hz frame rate for both display and simulation.*

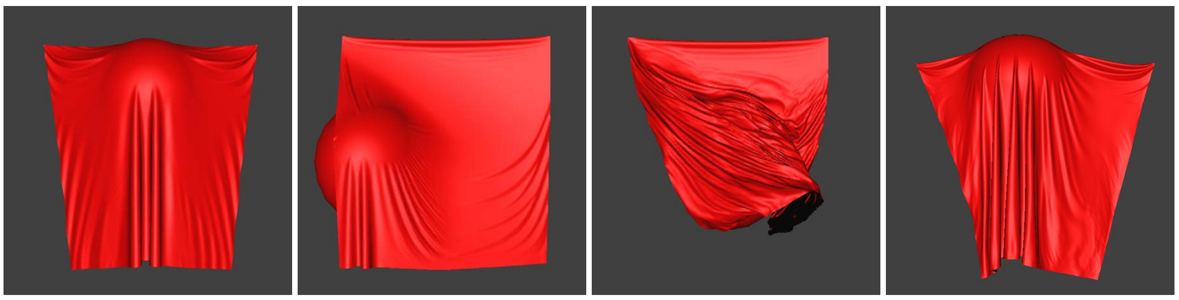

Figure 8: *Simulation of 90K cloth at 20fps on a GPU. Two upper corners of the cloth are pinned. 10 iterations per frame.*

[MC10] Müller M., Chentanez N.: Wrinkle meshes. In *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Aire-la-Ville, Switzerland, Switzerland, 2010), SCA '10, Eurographics Association, pp. 85–92. 2, 5

[MHHR07] Müller M., Heidelberger B., Hennix M., Ratcliff J.: Position based dynamics. *J. Vis. Comun. Image Represent. 18* (April 2007), 109–118. 2, 3, 4

[MMP87] Mitchell J. S. B., Mount D. M., Papadimitriou C. H.: The discrete geodesic problem. In *SIAM J. of Computing* (1987), vol. 16 of *SIAM J. of Computing*. 3

[MSJT08] Müller M., Stam J., James D., Thürey N.: Real time physics: class notes. In *ACM SIGGRAPH 2008 classes* (New York, NY, USA, 2008), SIGGRAPH '08, ACM, pp. 88:1–88:90. 2

[Mul08] Muller M.: Hierarchical Position Based Dynamics. In *VRIPHYS* (2008), pp. 1–10. 2, 5

[Pro95] Provot X.: Deformation constraints in a mass spring model to describe rigid cloth behavior. Graphics Interface 95, pp. 147–154. 2

[SSK*05] Surazhsky V., Surazhsky T., Kirsanov D., Gortler S., Hoppe H.: Fast exact and approximate geodesics on meshes. In *ACM SIGGRAPH Proceedings* (2005), ACM SIGGRAPH 2005. 3

[TPS09] Thomaszewski B., Pabst S., Strasser W.: Continuum-based strain limiting. Eurographics 2009. 2

[VMTF09] Volino P., Magnenat-Thalmann N., Faure F.: A simple approach to nonlinear tensile stiffness for accurate cloth simulation. *ACM Trans. Graph. 28* (September 2009), 105:1–105:16. 2

[WHRO10] Wang H., Hecht F., Ramamoorthi R., O'Brien J.: Example-based wrinkle synthesis for clothing animation. *ACM Trans. Graph. 29* (July 2010), 107:1–107:8. 2

[WOR10] Wang H., O'Brien J., Ramamoorthi R.: Multi-resolution isotropic strain limiting. *ACM Trans. Graph. 29* (December 2010), 156:1–156:10. 2, 5