Pacific Graphics 2018
H. Fu, A. Ghosh, and J. Kopf
(Guest Editors)

Volume 37 (2018), Number 7

Parallel Multigrid for Nonlinear Cloth Simulation

Zhendong Wang[1,2†], Longhua Wu[2], Marco Fratarcangeli[3], Min Tang[1], and Huamin Wang[2‡]

[1]Zhejiang University, China
[2]The Ohio State University, USA
[3]Chalmers University of Technology, Sweden

(a) An early state (b) A middle state (c) A late state

Figure 1: *The animation result of a funnel example with 120K vertices and 238K triangles, produced by a novel nonlinear adaptive multigrid method. This method organically integrates nonlinearity adaptive smoothing into a geometric multigrid framework and it is compatible with parallelization on the GPU. Thanks to this method, our simulator solves cloth dynamics for each frame of this example in 0.2 seconds, when using a large time step $\Delta t = 1/90s$.*

Abstract

Accurate high-resolution simulation of cloth is a highly desired computational tool in graphics applications. As single-resolution simulation starts to reach the limit of computational power, we believe the future of cloth simulation is in multi-resolution simulation. In this paper, we explore nonlinearity, adaptive smoothing, and parallelization under a full multigrid (FMG) framework. The foundation of this research is a novel nonlinear FMG method for unstructured meshes. To introduce nonlinearity into FMG, we propose to formulate the smoothing process at each resolution level as the computation of a search direction for the original high-resolution nonlinear optimization problem. We prove that our nonlinear FMG is guaranteed to converge under various conditions and we investigate the improvements to its performance. We present an adaptive smoother which is used to reduce the computational cost in the regions with low residuals already. Compared to normal iterative solvers, our nonlinear FMG method provides faster convergence and better performance for both Newton's method and Projective Dynamics. Our experiment shows our method is efficient, accurate, stable against large time steps, and friendly with GPU parallelization. The performance of the method has a good scalability to the mesh resolution, and the method has good potential to be combined with multi-resolution collision handling for real-time simulation in the future.

CCS Concepts

•***Computing methodologies → Physical simulation;***

1. Introduction

Recently, the use of physics-based cloth simulation is exploding in applications related to virtual dressing room, customizable fashion design, and virtual human modeling. Many of these applications desire physics-based cloth simulation be fast yet realistic, so as to

† wangzhendong@zju.edu.cn
‡ whmin@cse.ohio-state.edu

© 2018 The Author(s)
Computer Graphics Forum © 2018 The Eurographics Association and John Wiley & Sons Ltd. Published by John Wiley & Sons Ltd.

produce high-quality results without spending too much computational resource. The quality of cloth simulation is determined mostly by the mesh resolution and the simulation accuracy. The mesh resolution affects not only the smoothness of fine wrinkles such as those in Fig. 1, but also the locking issue, which restricts wrinkles from happening in low-resolution simulation. The simulation accuracy measures how exactly the nonlinear simulation problem gets solved in every time step. Without sufficient accuracy, the simulation can cause various artifacts and fail to meet its requirement in design and measurement-related applications. How to increase the mesh resolution and improve the simulation accuracy without introducing too much computational cost is the open problem in physics-based cloth simulation that needs more investigation.

As it becomes more and more difficult for single-resolution simulation to further explore computational power nowadays, we believe the future of cloth simulation is in multi-resolution simulation. By using a simulation hierarchy, multi-resolution simulation can efficiently reduce low-frequency errors, which are expensive to handle in high resolution. Perhaps the most popular and systematic way of implementing this idea is known as *multigrid*. In general, there are two types of multigrid methods: *geometric multigrids*, which build the simulation hierarchy geometrically from a mesh hierarchy; and *algebraic multigrids*, which create the simulation hierarchy algebraically by collapsing the system matrix [TJM15]. While multigrid methods are popular in mechanical engineering and computational physics, their research and usage in computer graphics is rather limited.

In this paper, we focus our research on the development of a novel geometric multigrid method. The reason we favor geometric multigrid over algebraic multigrid is because it is more convenient for geometric multigrid to accept nonlinearity. Although algebraic multigrid with smoothed aggregation [TJM15] is efficient in linear cloth simulation with small time steps, it is not efficient when apply it to nonlinear cloth simulation because the computational cost of the aggregation phase is very high.

Nonlinearity is related to the simulation accuracy. Researchers have been aware of the nonlinearity in cloth simulation for decades, but most simulators still choose to solve a linearized system in every time step. Mathematically equivalent to solving a nonlinear system by one iteration of Newton's method, this practice suffers from error accumulation or divergence, when the time step is large. Recently, Liu and colleagues [LBOK13] and Bouaziz and collaborators [BML*14] developed projective dynamics for fast simulation of elastic solids under a specific nonlinear elastic model. Many researchers [Wan15, NOB16, FTP16, WY16] pointed out that projective dynamics is a special case of nonlinear optimization methods since then. Their research inspires us to consider multigrid from a nonlinear optimization perspective and to develop a novel nonlinear multigrid method.

Our research is focused on integrating nonlinearity, adaptive smoothing, and multigrid into the development of a novel simulation method, for fast and accurate simulation of unstructured cloth meshes. While each of these topics has been investigated individually, we would like to address their joint implementation under a natural and unified framework. Our technical contributions can be summarized as follows.

- *Full multigrid.* We propose to formulate our method under a full multigrid (FMG) framework. The basic idea behind FMG is to solve low-resolution simulation by multigrid and treat its interpolated result as initialization to high-resolution simulation.
- *Nonlinearity.* We develop a novel nonlinear FMG method for fast cloth simulation. A unique feature of this method is that it considers the smoothing process at each resolution level as calculating a search direction at different frequency rates. Since it does not discretize the simulation problem at coarse levels as the full approximation scheme (FAS) does, its performance cannot be affected by the locking issue.
- *Adaptive smoothing.* Our adaptive smoother, implemented on the GPU, allows computational power to be concentrated on the regions with large residuals.

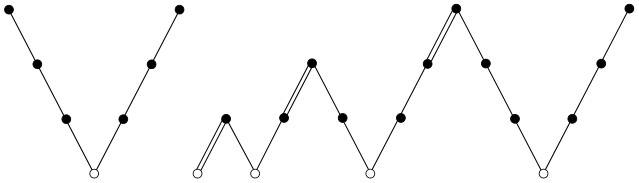
Our experiment reveals that the GPU implementation of the proposed nonlinear adaptive FMG method is fast and stable, even when it handles large time steps, as Fig. 1 shows. The performance of the whole method has a good scalability to the mesh resolution, as shown in Fig. 14.

2. Related Work

Physics-based cloth simulation The pioneer work by Baraff and Witkin [BW98] has inspired graphics researchers to study physics-based cloth simulation for decades. According to how planar elasticity of cloth is represented, cloth simulation can use either mass-spring systems [CK02, BMF03], continuum-based systems [WOR11, NSO12], or even yarn-based systems [KJM10, CLMMO14]. In recent years, graphics researchers have gained interests in a variety of new topics related to the efficiency and the accuracy of cloth simulation, including elasticity measurement [WOR11, MBT*12], internal and external frictions [CFW13, MTB*13], nonlinear elasticity [VMTF09, LBOK13, BML*14], GPU-based implementation [Wan15, FTP16], and collision handling [BFA02, WTTM15, TWT*16]. Being focused on solving the nonlinear system raised in physics-based cloth simulation, our work is relatively orthogonal to these research topics. Nevertheless, we hope our method can serve as a basis for other cloth simulation research in the future.

Adaptive refinement of unstructured meshes Adaptive mesh refinement has been an active research topic in physics-based simulation recently. A detailed survey can be found in [MWN*17]. In this work, we are specifically interested in adaptive refinement of unstructured meshes, since they are more suitable for representing irregular clothing shapes. A bottom-up approach of implementing adaptive mesh refinement is to remesh a given coarse mesh on the fly, by subdivision schemes [LV05, BD12] or edge operations [SLD09, NSO12]. The strength of the bottom-up approach is it is free of pre-processing, which is highly desired in the simulation of tetrahedral meshes [WRK*10, CWSO13]. On the other hand, the approach is unfriendly with parallelization and its refinement criteria are based on coarse simulation, which is inaccurate at the first place. An alternative approach to adaptive mesh refinement is to precompute a mesh hierarchy and then determine active simulation regions locally at each resolution level, as experimented in [WDGT01, DDCB01]. This approach is closely related

© 2018 The Author(s)
Computer Graphics Forum © 2018 The Eurographics Association and John Wiley & Sons Ltd.



(a) A V-cycle (b) A full multigrid case

Figure 2: *The structures of multigrid and full multigrid. Here • represents smoothing, ∘ represents an exact solver, \ and / represent restriction and interpolation, and // represents high-order interpolation.*

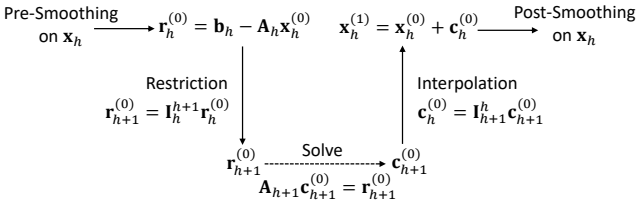


Figure 3: *The two-level structure of a linear multigrid method. It transfers the correction* **c** *and the residual* **r**, *rather than the solution* **x** *and the constant* **b**, *between the two levels.*

to multigrid, which needs to build a mesh hierarchy as well. In fact, geometric adaptivity is one of the goals set in the development of full-approximation storage (FAS) multigrid [Bra77]. Otaduy and colleagues [OGRG07] investigated this idea in the simulation of unstructured tetrahedral meshes, together with adaptivity-aware collision detection techniques.

Multi-resolution elastic body simulation Multi-resolution elastic body simulation is naturally more difficult to develop for unstructured meshes than structured ones [ZSTB10]. In the past, researchers have studied multigrid simulation of thin shells [GTS02], Newton-multigrid for nonlinear elastic solids [GW06], generic linear multigrid for unstructured surface meshes [AKS05], and linear multigrid for cloth simulation [ONW08, JCK*13]. For fast approximation of elastic behaviors, graphics researchers have also studied handling positional constraints in a multi-resolution fashion [Mül08, WOR10]. While their techniques share many similarities with multigrid, they typically ignore pre-smoothing and run only one resolution cycle.

3. Background

The multigrid technique was originally developed for speeding up stationary iterative solvers, which tend to smooth the residual after a few number of iterations. Its idea is to downsample the problem after the residual gets smoothed each time, so that the smoothed and down-sampled residual can be reduced faster on a coarse mesh. A typical multigrid method contains four components: *smoothing*, i.e., a few stationary iterations; *restriction*, which downsamples the residual from a fine mesh to a coarse mesh; *interpolation*, which upsamples the correction from a coarse mesh back to a fine mesh; and *solver*, which solves the problem exactly at the coarsest level. These components are organized into a V-cycle, as Fig. 2a shows.

3.1. Linear Multigrid

Let $\mathbf{I}_h^{h+1}$ and $\mathbf{I}_{h+1}^h$ be the restriction and interpolation operators between two consecutive levels h and h+1. When the problem at the finer level h is linear: $\mathbf{A}_h\mathbf{x}_h = \mathbf{b}_h$, we can interpret the problem as: $\mathbf{x}_h = \mathbf{x}_h^{(0)} + \mathbf{c}_h$ and $\mathbf{A}_h\mathbf{c}_h = \mathbf{b}_h - \mathbf{A}_h\mathbf{x}_h^{(0)} = \mathbf{r}_h^{(0)}$, in which $\mathbf{x}_h^{(0)}$ is the initial result, $\mathbf{r}_h^{(0)}$ is its initial residual, and $\mathbf{c}_h$ is the correction to $\mathbf{x}_h^{(0)}$. In a coarse correction step, a multigrid method tries to estimate $\mathbf{c}_h$ by solving the problem at the coarser level: $\mathbf{A}_{h+1}\mathbf{c}_{h+1} = \mathbf{r}_{h+1}^{(0)}$, in which $\mathbf{r}_{h+1}^{(0)} = \mathbf{I}_h^{h+1}\mathbf{r}_h^{(0)}$. The coarse correction result $\mathbf{c}_{h+1}$ is then upsampled by $\mathbf{I}_{h+1}^h$ to obtain an estimation of $\mathbf{c}_h$ at the finer level, as Fig. 3 shows.

An important question is how to formulate $\mathbf{A}_{h+1}$. One straightforward approach is to discretize the PDE problem on the coarse grid and derive the coarse linear system directly. However, doing this suffers from the inconsistency between the coarse linear system and the fine linear system, especially the locking issue in cloth simulation. Therefore, we choose to apply the *Galerkin* condition: $\mathbf{A}_{h+1} = \left(\mathbf{I}_{h+1}^h\right)^\mathsf{T}\mathbf{A}_h\mathbf{I}_{h+1}^h$, with an additional assumption $\mathbf{I}_h^{h+1} = \left(\mathbf{I}_{h+1}^h\right)^\mathsf{T}$.

Interpolation and restriction The performance of linear multigrid relies on the interpolation operator $\mathbf{I}_{h+1}^h$. For an unstructured triangle mesh $\{\mathcal{X}_0, \mathcal{T}_0\}$, in which $\mathcal{X}_0$ is the set of its vertices and $\mathcal{T}_0$ is the set of its triangles, we first construct a mesh hierarchy, by recursively clustering vertices and re-triangulating the surface area in the 2D reference space. The result is a nested mesh hierarchy satisfying: $\mathcal{X}_0 \supset \mathcal{X}_1 \supset \ldots \supset \mathcal{X}_H$. We typically reduce the number of vertices by a factor of four from one level to another, and the coarsest mesh has approximately 100 to 400 vertices. More details about constructing a mesh hiearchy are in the Appendix.

Given the mesh hierarchy, we use the barycentric coordinates to formulate a piecewise linear interpolation operator for every vertex $i \in \mathcal{X}_h$:

$$f(\mathbf{u}_i) = \begin{cases} f_i, & \text{if } i \in \mathcal{X}_{h+1}, \\ B_a(\mathbf{u}_i)f_a + B_b(\mathbf{u}_i)f_b + B_c(\mathbf{u}_i)f_c, & \text{otherwise}, \end{cases} \quad (1)$$

in which $\mathbf{u}_i$ is vertex i's reference position, a, b, and c are the vertices of the triangle in $\mathcal{T}_{h+1}$ containing or close to vertex i in the reference space, f_a, f_b, and f_c are their function values, and $B_a(\mathbf{u}_i)$, $B_b(\mathbf{u}_i)$, and $B_c(\mathbf{u}_i)$ are their barycentric coordinates. We use this linear interpolation operator to formulate the full rank matrix $\mathbf{I}_{h+1}^h$ in the V-cycles for its simplicity and sparsity.

Smoothing While linear multigrid can use any stationary iterative solver as its smoother, we choose to use symmetric Gauss-Seidel in our method. This choice is not arbitrary and it is related to the convergence condition of our nonlinear FMG method, as shown in Subsection 4.2. To implement symmetric Gauss-Seidel on the GPU, we apply the multi-coloring strategy [FTP16]. The symmetry of Gauss-Seidel is achieved by handling colored vertices twice per iteration: once in the forward color order and once in the backward color order.

Solver We use sparse solvers by LU factorization to solve the linear system at the coarsest level. In our experiment, we notice that it is

© 2018 The Author(s)
Computer Graphics Forum © 2018 The Eurographics Association and John Wiley & Sons Ltd.

more efficient to use CPU solvers, such as MKL PARDISO. Since the linear system at the coarsest level is small, the data transfer time between the GPU and the CPU is negligible.

4. Our System

In this section, we will first introduce nonlinear cloth simulation in Subsection 4.1. We will then introduce the full multigrid (FMG) concept and generalize FMG into nonlinear FMG in Subsection 4.2, by formulating it as a line search method. We will analyze the convergence of nonlinear FMG and investigate the choices involved in its implementation. In Subsection 4.3, we will present an adaptive smoothing approach to accelerate our nonlinear FMG method. In Subsection 4.4, we will discuss how to applied our nonlinear FMG method to Projective Dynamic and analyze its performance. Finally, we will present how to handling collisions and contacts in the nonlinear FMG in Subsection 4.5.

4.1. Nonlinear Cloth Simulation

Cloth is a kind of highly nonlinear elastic body. As presented in [WY16], nonlinear elastic body simulation from time t to $t+1$ can be formulated as an unconstrained nonlinear optimization problem: $\mathbf{q}_{t+1} = argmin\ \varepsilon(\mathbf{q})$,

$$\varepsilon(\mathbf{q}) = \frac{1}{2h^2}(\mathbf{q}-\mathbf{q}_t-h\mathbf{v}_t)^T\mathbf{M}(\mathbf{q}-\mathbf{q}_t-h\mathbf{v}_t)+E(\mathbf{q}), \qquad (2)$$

in which $\mathbf{q} \in \mathbb{R}^{3N}$ and $\mathbf{v} \in \mathbb{R}^{3N}$ be the vertex position and velocity vectors of cloth, $\mathbf{M} \in \mathbb{R}^{3N\times 3N}$ is the mass matrix, h is the time step, $E(\mathbf{q})$ is the total potential energy evaluated at $\mathbf{q}$. The term $\mathbf{q}_t + h\mathbf{v}_t$ is the inertial and usually token as the initialization of descent optimization methods. This nonlinear optimization problem is usually solved by using the Newton's method. However, it needs to solve a linear system at each iteration. This would be very time consuming when the resolution of cloth is high. Projective Dynamics [LBOK13, BML*14] overcomes this problem by solving this optimization problem with a local step constraint projection and a global step distance minimization. In the global step, projective dynamics also needs to solve a linear system. But the system matrix is constant, so it can be prefactored at initialization. Though projective dynamics is very efficient, it usually suffers convergence issue even using direct methods to solve the global step, as shown in [Wan15]. We propose a novel nonlinear FMG method which can be applied to both Newton's method and projective dynamics to improve their performance.

4.2. Nonlinear FMG

While V-cycles can be applied to solve the high-resolution problem immediately, they can be used in a better way, known as *full multigrid* (FMG). Its basic idea is to solve the coarser problem sufficiently well first, interpolate the correction to the finer level as initialization, and then solve the finer problem, as visualized in Fig. 2b. The success of FMG depends on the quality of the initialization interpolator. Since it is applied only once between two resolution levels, it can be more advanced than the linear interpolator used in V-cycles, as explained in the following.

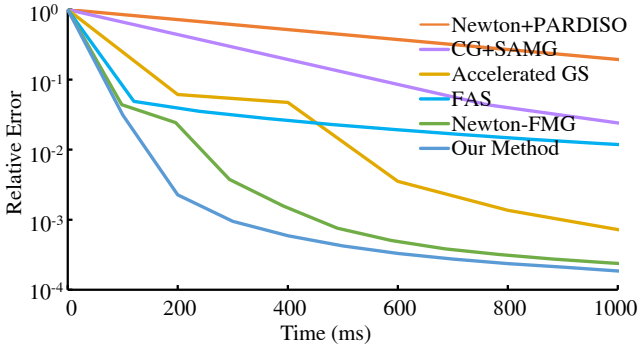


Figure 4: *The convergence of several nonlinear methods in the Sphere example with a 120K vertices cloth when using a large time step $\Delta t = 1/30s$. By default, we define the error as the relative energy loss: $(\varepsilon^{(k)}-\varepsilon^*)/(\varepsilon^{(0)}-\varepsilon^*)$, in which $\varepsilon^{(k)}$ and $\varepsilon^{(0)}$ are the current and initial energies, and ε^* is the estimated minimal energy. We typically do not use the residual magnitude for evaluation purposes, since it can fluctuate.*

There are two typical approaches to introduce nonlinearity into multigrid methods. The first approach, known as *Newton-multigrid*, simply applies multigrid to solve the linearized systems produced by Newton's method. Since this approach relies entirely on Newton's method, its performance becomes unsatisfactory when the time step is large, as shown in Fig. 4. The other approach, known as the *full approximation scheme* (FAS), transfers the positional solution $\mathbf{x}$, not the correction $\mathbf{c}$, between two resolution levels. At each level, FAS discretizes and solves the simulation problem for new $\mathbf{x}$ directly. While FAS is effective in FEM simulation of tetrahedral meshes [OGRG07], its performance in cloth simulation is inevitably affected by the locking issue in the coarse problem and incorporating the discretization error into the residual does not help much. As a result, FAS cannot benefit as much as our method from multi-resolution cloth simulation, as Fig. 4 shows. We compare the convergence of several nonlinear methods in Fig. 4, in which Newton+PARDISO and CG+SAMG are implemented on CPU, the other four methods are implemented on GPU. Newton's method has the lowest performance due to its large computational cost per iteration; congugate gradient method with a SAMG (smoothed aggregation multigrid) [TJM15] preconditioner, used as a linear solver for Newton's method, is not efficient either due to its difficulty in parallelism; Gauss-Seidel method has difficulty in handling high-resolution meshes even with Chebyshev acceleration; in contrast, our method implemented on the GPU achieves the highest performance.

Different from FAS and Newton-multigrid, our method can be fundamentally considered as a line search method for nonlinear optimization. The corrections at multiple resolution levels are the search directions at various frequency rates for the original problem at the finest level. Since no coarse problem is involved, the method is naturally free of the locking issue. Specifically, let $\mathbf{x}_0 = \arg\min \varepsilon(\mathbf{x}_0)$ be the original nonlinear optimization problem and $\mathbf{A}_0\mathbf{c}_0 = \mathbf{r}_0$ be the current linearized system, in which $\mathbf{r}_0 = -\nabla E$ is the residual and $\mathbf{A}_0$ is the Hessian or Hessian approximation of $\varepsilon(\mathbf{x}_0)$. In Newton-FMG method, the system at each hth level $\mathbf{A}_h\mathbf{c}_h = \mathbf{r}_h$ keeps the same inside a linear FMG, which is used as a linear solver.

© 2018 The Author(s)
Computer Graphics Forum © 2018 The Eurographics Association and John Wiley & Sons Ltd.

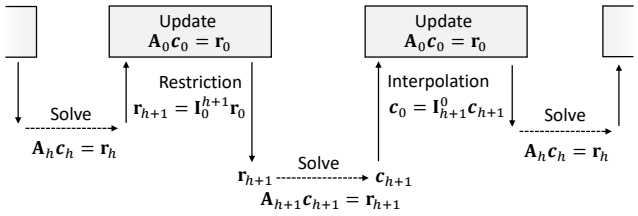


Figure 5: *The structure of our nonlinear FMG method. It updates the system matrix and the residual at the finest level only. This helps the method avoid the discretization error, i.e., the locking issue in coarse simulation.*

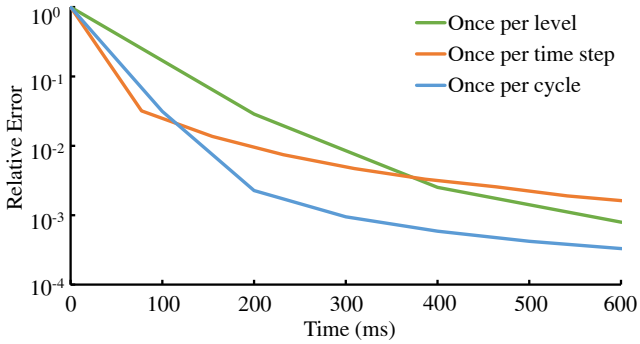


Figure 6: *Sphere example with a 120K vertices cloth using a large time step $\Delta t = 1/30$s. The convergence of our nonlinear FMG method with different choices of the system matrix. It shows that the optimal choice is to update the system matrix once per FMG cycle.*

But in our nonlinear FMG, the system at each hth level $\mathbf{A}_h\mathbf{c}_h = \mathbf{r}_h$ can be updated and no longer linear. The downsampled version of the finest system at the h-th level is:

$$\mathbf{A}_h\mathbf{c}_h = \mathbf{I}_0^h\mathbf{r}_0, \quad \text{for } \mathbf{A}_h = \mathbf{I}_0^h\mathbf{A}_0\mathbf{I}_h^0, \tag{3}$$

in which $\mathbf{I}_0^h = \mathbf{I}_{h-1}^h ... \mathbf{I}_1^2\mathbf{I}_0^1$ is also a full rank matrix. Equation 3 is not solved exactly (except at the coarsest level), but smoothed. Let $\tilde{\mathbf{A}}_h^{-1}$ be a matrix representing the smoothing process, such that: $\mathbf{c}_h = \tilde{\mathbf{A}}_h^{-1}\mathbf{I}_0^h\mathbf{r}_0$. The interpolated correction at the finest level becomes: $\mathbf{c}_0 = \mathbf{I}_h^0\mathbf{c}_h = \mathbf{I}_h^0\tilde{\mathbf{A}}_h^{-1}\mathbf{I}_0^h\mathbf{r}_0$. We apply this correction to $\mathbf{x}_0$ after proper scaling, update the residual $\mathbf{r}_0$ (and the matrix $\mathbf{A}_0$ if needed), and then find another correction at a different level, as Fig. 5 shows.

To ensure the convergence of the method, we must show $\mathbf{c}_0 \cdot \mathbf{r}_0 > 0$, which is true if $\tilde{\mathbf{A}}_h^{-1}$ is positive definite. In the following theorem, we prove that if $\mathbf{A}_0$ and $\mathbf{A}_h$ are positive definite, then $\tilde{\mathbf{A}}_h^{-1}$ must also be positive definite for symmetric Gauss-Seidel. Furthermore, we prove that $\tilde{\mathbf{A}}_h^{-1}$ is also positive definite for symmetric Gauss-Seidel with Chebyshev acceleration [GVL96] under the same condition.

THEOREM 4.1. *Let $\mathbf{Ax} = \mathbf{b}$ be a linear system and $\mathbf{A}$ be positive definite. If $\mathbf{x}^{(0)} = \mathbf{0}$ and $\mathbf{x}^{(K)} = \tilde{\mathbf{A}}^{-1}\mathbf{b}$ is the result after K symmetric Gauss-Seidel iterations, then $\tilde{\mathbf{A}}^{-1}$ must be positive definite. If $\mathbf{y}^{(K)} = \tilde{\mathbf{A}}^{-1}\mathbf{b}$ is the result after K symmetric Gauss-Seidel iterations with Chebyshev acceleration, then $\tilde{\mathbf{A}}^{-1}$ must be also positive definite. The proof is in the Appendix A.*

Like other line search methods, our method needs a good balance between the convergence rate and the computational cost per search for optimal performance. In the rest of the subsection, we will study this balance from the calculations of $\mathbf{A}_0$ and $\mathbf{b}_0$. We will also discuss the implementation of backtracking line search.

4.2.1. The choices of the system matrix

The choices of the system matrix $\mathbf{A}_0$ affect the convergence speed, but not the convergence guarantee, as long as $\mathbf{A}_0$ is positive definite. In our system, we define $\mathbf{A}_0$ as the modified[†] Hessian matrix, but we do not update every time a correction gets updated. In other words, $\mathbf{A}_0$ is an Hessian matrix approximation since the last update, which must still be positive definite. The question though is how often should we perform the update? It is expected that we should not update $\mathbf{A}_0$ at every resolution level, since that introduces a large computational cost per search and slows down the system performance, as shown in Fig. 6. On the other hand, if we update $\mathbf{A}_0$ only once at the beginning of the time step, the method becomes a nonlinear solver accelerated by a constant preconditioner matrix, whose convergence rate is not optimal either.

Our idea is to restructure the V-cycles in our nonlinear FMG method. Specifically, instead of running small V-cycles for coarse problems multiple times first and then large V-cycles for fine problems, we propose to interleave small V-cycles with large V-cycles. Intuitively, this can be considered as solving the whole problem by multiple FMG cycles, each of which uses only one V-cycle at every level, such as the one shown in Fig. 2b. We can then conveniently update $\mathbf{A}_0$ at the beginning of every FMG cycle and Fig. 6 shows this approach achieves a fast convergence speed. The reason we do not use the standard FMG formulation is because it would be too expensive to update $\mathbf{A}_0$ solely for small V-cycles. In contrast, our formulation applies the same matrix update to both small and large V-cycles, which allows small V-cycles to be more effective.

4.2.2. The choices of updating residual

The next question we would like to study is whether we should incorporate nonlinearity into every level, or just a small number of fine levels. The reason we have this question is because the interpolation and the restriction can cause a nonneglectable computational costs especially at coarse levels. If we restrict the recalculation of the residual to fine levels only, we can effectively reduce these costs. Intuitively, this can be considered as using linear multigrid to smooth the problem at the bottom level. The question now becomes whether the matrix representing the smoothing process of linear multigrid is positive definite or not. The following theorem proves that it is true, if pre-smoothing and post-smoothing are symmetric.

THEOREM 4.2. *Suppose that the linear system is positive definite. If the matrices representing the pre-smoothing process and the post-smoothing process at the same level are identical and positive definite, then the matrix representing the whole smoothing process*

[†] The Hessian matrix of an elastic solid is often not positive definite. It needs modifications to be positive definite. See [CK02, TSIF05] for more details.

© 2018 The Author(s)
Computer Graphics Forum © 2018 The Eurographics Association and John Wiley & Sons Ltd.

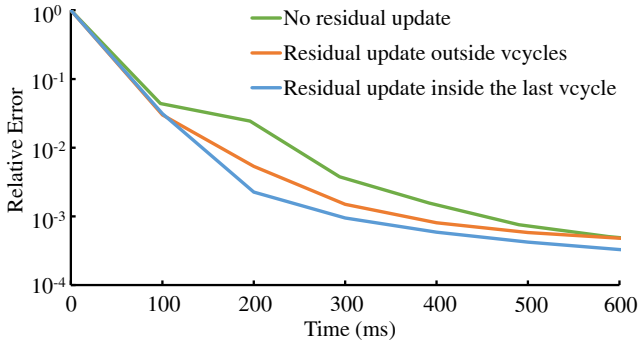


Figure 7: *Sphere example with a 120K vertices cloth using a large time step $\Delta t = 1/30s$. The convergence of our nonlinear FMG method with different choices of the residual update. It shows that the optimal choice is to update the residual in the last biggest V-Cycle.*

provided by linear multigrid is also positive definite. The proof is in the Appendix A.

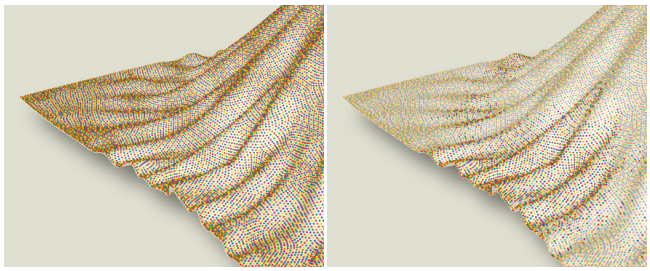
Thanks to Theorem 4.2, we can now freely choose the number of fine levels involved in residual recalculation. Fig. 7 compares three choices of the residual update. It indicates that the optimal choice is to recalculate the residual at an intermediate level in the last bigest V-cycle, so that the nonlinearity and the interpolation/restriction cost among resolution levels can be well balanced. Another choice we have made is that we only use linear interpolation scheme in our nonlinear FMG. In our experiments, we found high-order interpolation schemes help little for the convergence of our nonlinear FMG. The restriction and interpolation matrices must satisfy the Galerkin condition to keep our nonlinear FMG stable and efficient. If the restriction matrices are from high-order interpolation matrices, the matrices $\mathbf{A}_h$ at coarse level would be very dense. This reduces the performance of multigrid method.

4.2.3. Global convergence

As a line search method, our nonlinear FMG method must scale each correction by a proper step length. Here we choose to use backtracking line search. An interesting phenomenon we noticed from our experiment is that the corrections calculated at coarser levels tend to need smaller step lengths when only update the residuals but not Hessian matrices. When increasing the frequency of updating the Hessian matrices, the step length of each correction increases to one. But updating Hessian matrices is very time consuming compared to updating residuals. So we only update Hessian matrices only when FMG back to the finest level. This also inspires us to use different initial step lengths at different levels. Doing so reduces the computational cost spent on backtracking steps by approximately 10 percent.

4.3. Adaptive Smoother

Multigrid applies the smoothing operator, i.e., the stationary iterative solver, to find a correction that reduces the residual. Therefore, we can naturally develop an adaptive smoother to skip smoothing



(a) Without adaptive smoothing (b) With adaptive smoothing

Figure 8: *The vertices visualized in their colors, without and with adaptive smoothing. The desaturated vertices in (a) are those with low residuals. Our adaptive smoother skips them from the smoothing process.*

where the residual has dropped below a certain threshold. This idea, also explored in [OGRG07] has a unique advantage: the threshold places an upper bound on the error caused by adaptive smoothing.

The adaptive smoother is highly compatible with multi-color symmetric Gauss-Seidel that we propose to use in Subsection 3.1. By definition, the vertices in the same color are not adjacent according to the connectivity of the system matrix. This means the corresponding equations are independently, so when we calculate the residual magnitude at a vertex to determine if it needs a correction, we do not need to worry about its residual being affected by corrected vertices in the same color. Fig. 8 illustrates the mesh used in the sphere example drawn in six different colors. The colorful vertices are the ones being corrected, and the desaturated vertices are the ones being skipped. In average, our adaptive smoother can reduce the computational time spent on smoothing by approximately 30 percent.

We note that the adaptive smoother does not need to specifically arrange the vertices for GPU kernels and it can work efficiently as it is. In fact, resorting the vertices would lower the system performance because its computational overhead, so we do not recommend it.

4.4. Nonlinear FMG for Projective Dynamics

Projective Dynamics is very efficient to simulate constraint based soft bodies. The nonlinearity of constraints is handled in the local step projection. In the global step, it solves a sparse linear system with a constant matrix. This matrix can be prefractored at the initialization . Using only a few iterations, projective dynamics can get an approximate solution which is very similar to that of Newton's method. However, as the cloth resolution and stiffness increasing and the time step becoming larger such as $33ms$, the convergence of projective dynamics reduces. Obvious artifacts appear even using direct methods to solve the global step, as demonstrated in [Wan15]. Using traditional linear multigrid method to solve the global step also suffer the same issue.

[Wan15] proposed to solve the linear sparse system by parallel Jacobi method and used a series of Chebyshev weights to accelerate the convergence of projective dynamics. [FTP16] adapted parallel

© 2018 The Author(s)
Computer Graphics Forum © 2018 The Eurographics Association and John Wiley & Sons Ltd.

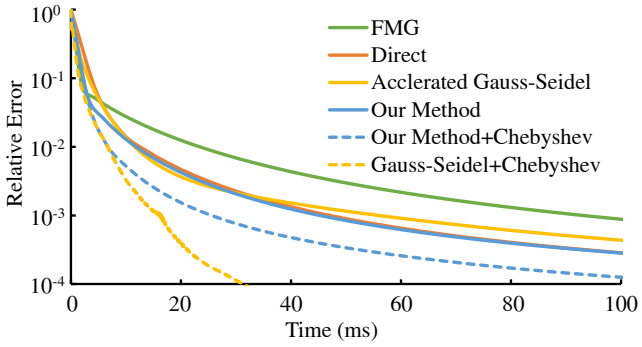


Figure 9: *Tablecloth example with a 10K vertices cloth simulated by Projective Dynamics using a large time step $\Delta t = 1/30s$. Though our nonlinear FMG method with Chebyshev acceleration can converge very fast, it is not the best choice when simulating relative low-resolution cloth.*

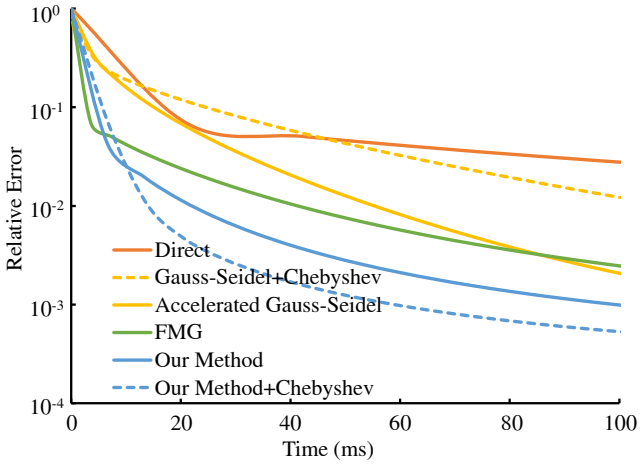


Figure 10: *Tablecloth example with a 40K vertices cloth simulated by Projective Dynamics using a large time step $\Delta t = 1/30s$. Compared to the low-resolution example in Fig. 9, multigrd methods have obvious advantages over the other three methods when simulating a relative high-resolution cloth. Our nonlinear FMG method and its counterpart with Chebyshev acceleration achieve the best convergences.*

Gauss-Seidel method and achieved better convergence than Jacobi method. Although iterative methods may not solve the linear system exactly, they allow more frequently local step constraints projection to handle the nonlinearity. Fig. 9 shows Gauss-Seidel method can achieve very fast convergence when Chebyshev acceleration is applied between the local step and the global step. In contrast, when using accelerated Gauss-Seidel method as a linear solver for the global step, i.e. the Chebyshev acceleration is applied inside the global step, the convergence slows down a lot. However, the convergence of the Gauss-Seidel method with Chebyshev acceleration drops very fast when simulating high-resolution cloth, as illustrated in Fig. 10. This is because the global step requires more accuracy when the linear system becomes larger as the resolution of cloth increasing.

In our experiments, we observed that the local step and global step must be balanced. On one hand, if more computational cost is spent on global step, then nonlinearity cannot be handled well. This will cause over-relaxation. On the other hand, if the global step can not be solved to enough precision, it would also cause less-relaxation. The cloth may be too loose in the area with hard constraints, such as fixed points. Based on this observation, we find our nonlinear FMG method can significantly speedup the convergence of projective dynamics because it provides a good strategy for balancing the local step and global step.

It is very easy to integrate our nonlinear FMG method with projective dynamics. As describe in Sec. 4.2.2, when the residual equations $\mathbf{A}_h\mathbf{x}_h = \mathbf{b}_h$ on multi-resolution coarse levels have been solved or smoothed, we interpolate the corrections $\mathbf{x}_h$ back to the finest level $\mathbf{x}_0 = \mathbf{x}_0 + \mathbf{I}_h^0\mathbf{x}_h$ and update the right hand residuals $\mathbf{b}_0$. In projective dynamics, updating the right hand term $\mathbf{b}_0$ is actually doing a local step constraint projection. Chebyshev weights can also be applied into our nonlinear FMG to speed-up the convergence of projective dynamics. And the correcting process becomes $\mathbf{x}_0^{(k+1)} = \omega[(\mathbf{x}_0^{(k)} + \mathbf{I}_h^0\mathbf{x}_h) - \mathbf{x}_0^{(k-1)}] + \mathbf{x}_0^{(k-1)}$, in which $\mathbf{x}_0^{(k)}$ is the current solution at the finest level and ω is the Chebyshev weight. By adjusting the smoothing iterations used on each level, the local step constraint projection and the global step linear system smoothing get to a balance point in our nonlinear FMG framework. Different from the Newton's method, we do not do line search in projective dynamics when doing corrections to the solution at the finest level. Even though the energy of the optimization problem Eq. 2 may increase, the local step projection can ensure the stability of projective dynamics. This allows our nonlinear FMG method to be very efficient in projective dynamics.

4.5. Collisions and Contacts

While our research is focused on dynamic simulation, not collision handling, the mesh hierarchy we built for multigrid can be used to facilitate the handling of cloth self collisions as well. To do so, we simply downsample the mesh from fine to coarse, and then handle self collisions from coarse to fine. In our examples, we use our in-house discrete collision handling system based on intersection contour minimization [VMT06]. We note that our self collision treatment is far from being optimal and we would love to explore the relationship between multigrid and multi-resolution collision handling even further.

To handle cloth collisions with rigid bodies, we use a signed distance field for fast detection, and both projection and penalty force for removing collisions. The penalty force can be naturally integrated into the nonlinear system solving process. To achieve frictional effects, we simply apply velocity filtering on each vertex. One nice property of our nonlinear FMG method is that we always return to the finest level for updating $\mathbf{x}$ after each nonlinear smoothing process. This allows collision projection to be injected into the solver for quick response. In contrast, standard multigrid methods handle collisions only at the beginning or the end of each V-cycle, which cause the coarse levels to be ineffective due to their ignorance of collisions.

© 2018 The Author(s)
Computer Graphics Forum © 2018 The Eurographics Association and John Wiley & Sons Ltd.

(a) GS+Chebyshev (b) Ours (c) Ours+Chebyshev

(d) GS+Chebyshev (e) Ours (f) Ours+Chebyshev

Figure 11: *Tablecloth examples simulated by Projective Dynamics using a large time step $\Delta t = 1/30$s and the same real-time budgets. Cloth in the first row have 10k vertices. Cloth in the second row have 40k vertices.*

5. Results

(Please watch the supplemental video for animation examples.) In the implementation of our method, we use the Intel MKL PARDISO library for CPU computation and the CUDA library for GPU computation. Our experiment runs on an Intel Core i7-5930K 3.5GHz processor and an NVIDIA GeForce GTX 1080 graphics card. We typically use $\Delta t = 1/180$ to $1/30$s as our time step, and two to three FMG cycles per time step, which is sufficient for the relative residual to be less than one percent of the initial residual. Our multigrid typically uses three to four symmetric Gauss-Seidel iterations in every smoothing process. Table 1 summarizes the statistics and the timings of our examples. The two bottlenecks in our simulator are the symmetric Gauss-Seidel smoother, which consumes nearly 50 percent of the computational cost, and the restriction operation applied to the matrix (not in projective dynamics), which consumes about 30 percent of the computational cost. When do coloring for matrice on different resolution levels, the number of the colors varies from level to level, since the matrix sparsity is under the influence of the interpolator. In our experiment, this number can be as high as 14 at the coarsest level. For a sparse matrix, its CSR format is the same as the CSC format of its transpose. We take advantage of this property to accelerate the computation of ${\mathbf{I}_{h+1}^{h}}^T \mathbf{A}_h \mathbf{I}_h^{h+1}$, which is about 4 times faster than the sparse matrix multiplication operation in CuSparse.

Dressed character examples Fig. 12 reveals the ability of our simulator in animating a gown mesh and a secretary dress mesh, worn by a virtual walking mannequin. In these examples, the garments are modeled from commercial sewing patterns, so they fit the virtual body well and they do not demonstrate excessive wrinkles. But our simulator is fully capable of simulating complex garments designed with fine shirring wrinkles as well, as long as collision handling is not an issue.

Name	#Vert	#Tri	#Color	Cost/Cycle	Cost/Step
Sphere	120K	238K	6 to 13	100ms	200ms
Funnel	120K	238K	6 to 13	100ms	200ms
Dress	132K	264K	5 to 14	111ms	333ms
Gown	104K	208K	5 to 12	84ms	252ms
Curtain	40K	80K	5 to 11	33ms	33ms

Table 1: *Statistics and timings of our examples. Here the timings do not include the computational time spent on collision handling.*

(a) A gown example with 104K vertices (b) A dress example with 132K vertices

Figure 12: *Garment examples. These examples demonstrate the ability of our simulator in animating virtual garments dressed on a walking character.*

An interactive curtain demo While our simulator can be used to efficiently simulate high-resolution cloth meshes, it can also be used to simulate meshes in relatively lower resolutions at a real time or interactive rate. For example, Fig. 13 shows that the simulator can handle the simulation of a rectangular mesh with 40K vertices at 31 FPS. Such examples would be too expensive to handle by previous single-resolution simulators on the GPU [FTP16, WY16], without overly stretching artifacts.

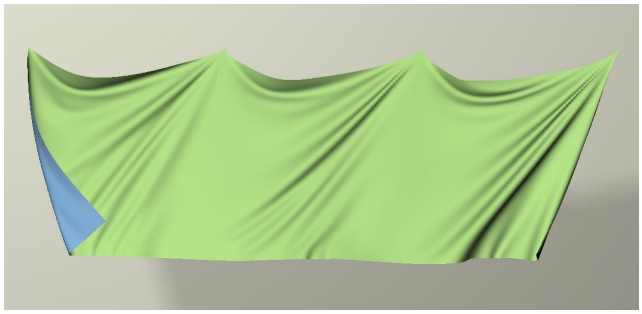
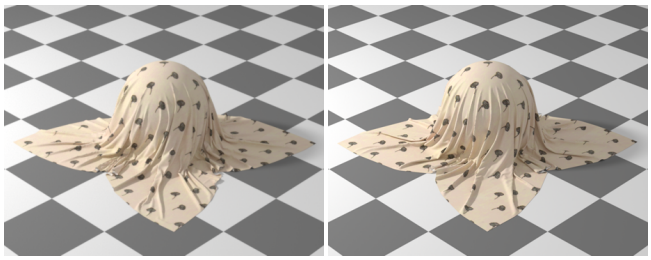
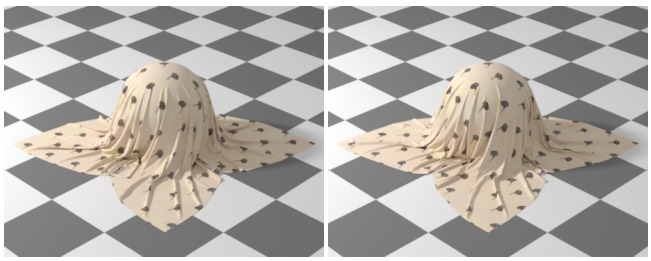


Figure 13: *The curtain example. Our nonlinear FMG simulator is able to handle this example with 40K vertices and 80K triangles in real time.*

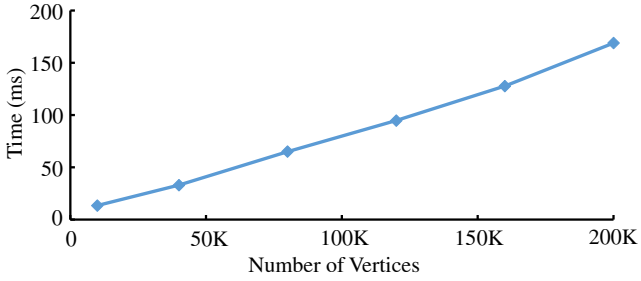
© 2018 The Author(s)
Computer Graphics Forum © 2018 The Eurographics Association and John Wiley & Sons Ltd.



(a) A cloth mesh with 80K vertices (b) A cloth mesh with 120K vertices



(c) A cloth mesh with 160K vertices (d) A cloth mesh with 200K vertices



(e) The relationship between the mesh resolution and the cost per time step

Figure 14: *The sphere example. In this example, we test the scalability of our method, by dropping the same cloth model in four different resolutions onto a rotating sphere. Our method exhibits nearly linear scalability.*

Scalability to the mesh resolution In this experiment, we would like to evaluate the scalability of our method to different mesh resolutions. Fig 14a to 14d are four animation results, in which the same tablecloth in four mesh resolutions fall onto a unit sphere. Fig 14e shows that their computational costs per time step are nearly linear to the numbers of vertices, when using the same relative residual threshold as the convergence condition. We note that the simulator runs an integral number of FMG cycles. When the demand on FMG cycles increases as the mesh resolution grows, the computational cost will experience a sudden jump, which is not demonstrated in Fig 14e. This is why we typically use a fixed number of FMG cycles, rather than the residual threshold, as the convergence condition in our experiment.

5.1. Limitations

As geometric multigrid, our method needs to construct the mesh hierarchy ahead of time, which demands extra implementation effort. The mesh hierarchy construction process implicitly assumes that mesh vertices are uniformly distributed, which may not always be the case. Fundamentally as a line search method, our nonlinear FMG method has been proven to be convergent for a limited number of smoothers. The adaptive smoother is not optimal and we believe it still has space for further improvement. Finally, while the multi-resolution structure of our method naturally enables multi-resolution collision handling, it has not fully explored the strength of this idea. As a result, the performance gain provided by multi-resolution simulation diminishes in complex collision cases, when most computational cost is spent on collision handling.

6. Conclusions and Future Work

In this paper, we present a nonlinear, adaptive, geometric multigrid method for cloth simulation on the GPU. As our first step toward fast simulation with low resolution dependency, it has demonstrated its efficiency and robustness in handling high-resolution meshes and large time steps. Our experiment confirms that geometric multigrid is compatible with geometrically adaptive techniques. In particular, high curvature regions, i.e., wrinkles, are the places where larger residuals occur and more computational costs are needed.

Looking into the future, we plan to study other smoother options and better ways of achieving geometric adaptivity. We will then explore the combination of our method with multi-resolution collision handling techniques for fast simulation of complex collision cases. We are also interesting in knowing how multigrid can be applied to accelerate the simulation of unstructured volumetric meshes. Finally, we will investigate the use of multigrid in simulation-related problems, including space-time optimization and inverse elastic shape design.

Acknowledgement

This research is supported by China Scholarship Council. The project is partially funded by NSF grant IIS-1524992. We also would like to thank Adobe and NVIDIA for their funding and equipment support. Min Tang is supported in part by NSFC (61732015, 61572423).

References

[AKS05] AKSOYLU B., KHODAKOVSKY A., SCHRÖDER P.: Multilevel solvers for unstructured surface meshes. *SIAM J. Sci. Comput. 26*, 4 (Apr. 2005), 1146–1165. 3

[BD12] BENDER J., DEUL C.: Efficient Cloth Simulation Using an Adaptive Finite Element Method. In *Workshop on Virtual Reality Interaction and Physical Simulation* (2012), The Eurographics Association, pp. 21–30. 2

[BFA02] BRIDSON R., FEDKIW R., ANDERSON J.: Robust treatment of collisions, contact and friction for cloth animation. *ACM Trans. Graph. (SIGGRAPH) 21*, 3 (July 2002), 594–603. 2

[BMF03] BRIDSON R., MARINO S., FEDKIW R.: Simulation of clothing with folds and wrinkles. In *Proceedings of SCA* (2003), pp. 28–36. 2

[BML*14] BOUAZIZ S., MARTIN S., LIU T., KAVAN L., PAULY M.: Projective dynamics: Fusing constraint projections for fast simulation. *ACM Trans. Graph. (SIGGRAPH) 33*, 4 (July 2014), 154:1–154:11. 2, 4

[Bra77] BRANDT A.: Multi-level adaptive solutions to boundary-value problems. *Mathematics of Computation 31*, 138 (1977), 333–390. 3

[BW98] BARAFF D., WITKIN A.: Large steps in cloth simulation. In *Proceedings of the 25th annual conference on Computer graphics and*

© 2018 The Author(s)
Computer Graphics Forum © 2018 The Eurographics Association and John Wiley & Sons Ltd.

interactive techniques (New York, NY, USA, 1998), SIGGRAPH '98, ACM, pp. 43–54. 2

[CFW13] CHEN Z., FENG R., WANG H.: Modeling friction and air effects between cloth and deformable bodies. *ACM Trans. Graph. (SIGGRAPH) 32*, 4 (July 2013), 88:1–88:8. 2

[CK02] CHOI K.-J., KO H.-S.: Stable but responsive cloth. *ACM Trans. Graph. (SIGGRAPH) 21*, 3 (July 2002), 604–611. 2, 5

[CLMMO14] CIRIO G., LOPEZ-MORENO J., MIRAUT D., OTADUY M. A.: Yarn-level simulation of woven cloth. *ACM Trans. Graph. (SIGGRAPH Asia) 33*, 6 (Nov. 2014), 207:1–207:11. 2

[CWSO13] CLAUSEN P., WICKE M., SHEWCHUK J. R., O'BRIEN J. F.: Simulating liquids and solid-liquid interactions with lagrangian meshes. *ACM Trans. Graph. 32*, 2 (Apr. 2013), 17:1–15. 2

[DDCB01] DEBUNNE G., DESBRUN M., CANI M.-P., BARR A. H.: Dynamic real-time deformations using space & time adaptive sampling. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques* (2001), SIGGRAPH '01, pp. 31–36. 2

[FTP16] FRATARCANGELI M., TIBALDO V., PELLACINI F.: Vivace: A practical Gauss-Seidel method for stable soft body dynamics. *ACM Trans. Graph. (SIGGRAPH Asia) 35*, 6 (Nov. 2016), 214:1–214:9. 2, 3, 6, 8

[GTS02] GREEN S., TURKIYYAH G., STORTI D.: Subdivision-based multilevel methods for large scale engineering simulation of thin shells. In *Proceedings of SMA* (2002), pp. 265–272. 3

[GVL96] GOLUB G. H., VAN LOAN C. F.: *Matrix computations (3rd Ed.)*. Johns Hopkins University Press, Baltimore, MD, USA, 1996. 5

[GW06] GEORGII J., WESTERMANN R.: A multigrid framework for real-time simulation of deformable bodies. *Comput. Graph. 30*, 3 (June 2006), 408–415. 3

[JCK*13] JEON I., CHOI K.-J., KIM T.-Y., CHOI B.-O., KO H.-S.: Constrainable multigrid for cloth. *Computer Graphics Forum (Pacific Graphics) 32*, 7 (2013), 31–39. 3

[KJM10] KALDOR J. M., JAMES D. L., MARSCHNER S.: Efficient yarn-based cloth with adaptive contact linearization. *ACM Trans. Graph. (SIGGRAPH) 29*, 4 (July 2010), 105:1–105:10. 2

[LBOK13] LIU T., BARGTEIL A. W., O'BRIEN J. F., KAVAN L.: Fast simulation of mass-spring systems. *ACM Trans. Graph. (SIGGRAPH Asia) 32*, 6 (Nov. 2013), 214:1–214:7. 2, 4

[LV05] LI L., VOLKOV V.: Cloth animation with adaptively refined meshes. In *Proceedings of the Twenty-eighth Australasian Conference on Computer Science - Volume 38* (2005), pp. 107–113. 2

[Mül08] MÜLLER M.: Hierarchical position based dynamics. In *Proceedings of VRIPHYS* (2008), pp. 1–10. 3

[MBT*12] MIGUEL E., BRADLEY D., THOMASZEWSKI B., BICKEL B., MATUSIK W., OTADUY M. A., MARSCHNER S.: Data-driven estimation of cloth simulation models. *Comput. Graph. Forum (Eurographics) 31*, 2 (May 2012), 519–528. 2

[MTB*13] MIGUEL E., TAMSTORF R., BRADLEY D., SCHVARTZMAN S. C., THOMASZEWSKI B., BICKEL B., MATUSIK W., MARSCHNER S., OTADUY M. A.: Modeling and estimation of internal friction in cloth. *ACM Trans. Graph. (SIGGRAPH Asia) 32*, 6 (Nov. 2013), 212:1–212:10. 2

[MWN*17] MANTEAUX P.-L., WOJTAN C., NARAIN R., REDON S., FAURE F., CANI M.-P.: Adaptive physically based models in computer graphics. *Computer Graphics Forum 36*, 6 (2017), 312–337. 2

[NOB16] NARAIN R., OVERBY M., BROWN G. E.: ADMM ⊇ projective dynamics: Fast simulation of general constitutive models. In *Proceedings of SCA* (2016), pp. 21–28. 2

[NSO12] NARAIN R., SAMII A., O'BRIEN J. F.: Adaptive anisotropic remeshing for cloth simulation. *ACM Trans. Graph. (SIGGRAPH Asia) 31*, 6 (Nov. 2012), 152:1–152:10. 2

[OGRG07] OTADUY M. A., GERMANN D., REDON S., GROSS M.: Adaptive deformations with fast tight bounds. In *Proceedings of SCA* (2007), pp. 181–190. 3, 4, 6

[ONW08] OH S., NOH J., WOHN K.: A physically faithful multigrid method for fast cloth simulation. *Computer Animation and Virtual Worlds 19*, 3 (2008). 3

[SLD09] SIMNETT T. J. R., LAYCOCK S. D., DAY A. M.: An Edge-based Approach to Adaptively Refining a Mesh for Cloth Deformation. In *Theory and Practice of Computer Graphics* (2009), The Eurographics Association, pp. 77–84. 2

[TJM15] TAMSTORF R., JONES T., MCCORMICK S. F.: Smoothed aggregation multigrid for cloth simulation. *ACM Trans. Graph. (SIGGRAPH Asia) 34*, 6 (Oct. 2015), 245:1–245:13. 2, 4

[TSIF05] TERAN J., SIFAKIS E., IRVING G., FEDKIW R.: Robust quasistatic finite elements and flesh simulation. In *Proceedings of SCA* (2005), pp. 181–190. 5

[TWT*16] TANG M., WANG H., TANG L., TONG R., MANOCHA D.: CAMA: Contact-aware matrix assembly with unified collision handling for GPU-based cloth simulation. *Computer Graphics Forum (Eurographics) 35*, 2 (2016), 511–521. 2

[VMT06] VOLINO P., MAGNENAT-THALMANN N.: Resolving surface collisions through intersection contour minimization. *ACM Trans. Graph. (SIGGRAPH) 25*, 3 (July 2006), 1154–1159. 7

[VMTF09] VOLINO P., MAGNENAT-THALMANN N., FAURE F.: A simple approach to nonlinear tensile stiffness for accurate cloth simulation. *ACM Trans. Graph. 28*, 4 (Sept. 2009), 105:1–105:16. 2

[Wan15] WANG H.: A Chebyshev semi-iterative approach for accelerating projective and position-based dynamics. *ACM Trans. Graph. (SIGGRAPH Asia) 34*, 6 (Oct. 2015), 246:1–246:9. 2, 4, 6

[WDGT01] WU X., DOWNES M. S., GOKTEKIN T., TENDICK F.: Adaptive nonlinear finite elements for deformable body simulation using dynamic progressive meshes. *Computer Graphics Forum (Eurographics) 20*, 3 (2001), 349–ÍC358. 2

[WOR10] WANG H., O'BRIEN J., RAMAMOORTHI R.: Multi-resolution isotropic strain limiting. *ACM Trans. Graph. (SIGGRAPH Asia) 29*, 6 (Dec. 2010), 156:1–156:10. 3

[WOR11] WANG H., O'BRIEN J. F., RAMAMOORTHI R.: Data-driven elastic models for cloth: Modeling and measurement. *ACM Trans. Graph. (SIGGRAPH) 30*, 4 (July 2011), 71:1–71:12. 2

[WRK*10] WICKE M., RITCHIE D., KLINGNER B. M., BURKE S., SHEWCHUK J. R., O'BRIEN J. F.: Dynamic local remeshing for elastoplastic simulation. *ACM Trans. Graph. 29* (July 2010), 49:1–49:11. 2

[WTTM15] WANG Z., TANG M., TONG R., MANOCHA D.: Tightcd: Efficient and robust continuous collision detection using tight error bounds. *Comput. Graph. Forum 34*, 7 (Oct. 2015), 289–298. 2

[WY16] WANG H., YANG Y.: Descent methods for elastic body simulation on the GPU. *ACM Trans. Graph. (SIGGRAPH Asia) 35*, 6 (Nov. 2016), 212:1–212:10. 2, 4, 8

[ZSTB10] ZHU Y., SIFAKIS E., TERAN J., BRANDT A.: An efficient multigrid method for the simulation of high-resolution elastic solids. *ACM Trans. Graph. 29*, 2 (Apr. 2010), 16:1–16:18. 3

© 2018 The Author(s)
Computer Graphics Forum © 2018 The Eurographics Association and John Wiley & Sons Ltd.

Appendix A

THEOREM 4.1. *Let* $\mathbf{Ax}=\mathbf{b}$ *be a linear system and* $\mathbf{A}$ *be positive definite. If* $\mathbf{x}^{(0)}=\mathbf{0}$ *and* $\mathbf{x}^{(K)}=\tilde{\mathbf{A}}^{-1}\mathbf{b}$ *is the result after K symmetric Gauss-Seidel iterations, then* $\tilde{\mathbf{A}}^{-1}$ *must be positive definite. If* $\mathbf{y}^{(K)}=\tilde{\mathbf{A}}^{-1}\mathbf{b}$ *is the result after K symmetric Gauss-Seidel iterations with Chebyshev acceleration, then* $\tilde{\mathbf{A}}^{-1}$ *must be also positive definite.*

PROOF. Let $\mathbf{A}=\mathbf{D}+\mathbf{L}+\mathbf{L}^{\mathsf{T}}$, where $\mathbf{D}$ is the diagonal part and $\mathbf{L}$ is the lower triangular part. By definition of symmetric Gauss-Seidel iteration, we know:

$$\begin{aligned}\mathbf{x}^{(k+1)} &= \mathbf{x}^{(k)}+\mathbf{M}^{-1}\left(\mathbf{b}-\mathbf{A}x^{(k)}\right)\\ &= \mathbf{M}^{-1}\mathbf{b}+\mathbf{M}^{-1}(\mathbf{M}-\mathbf{A})\mathbf{x}^{(k)}\\ &= \mathbf{M}^{-1}\mathbf{b}+\mathbf{M}^{-1}\mathbf{L}\mathbf{D}^{-1}\mathbf{L}^{\mathsf{T}}\mathbf{x}^{(k)}.\end{aligned} \tag{4}$$

in which $\mathbf{M}=(\mathbf{D}+\mathbf{L})\,\mathbf{D}^{-1}\,(\mathbf{D}+\mathbf{L})^{\mathsf{T}}$. From Equation 4, we get:

$$\tilde{\mathbf{A}}^{-1}=\sum_{k=0}^{K-1}\left(\mathbf{M}^{-1}\mathbf{L}\mathbf{D}^{-1}\mathbf{L}^{\mathsf{T}}\right)^{k}\mathbf{M}^{-1}. \tag{5}$$

Each term in $\tilde{\mathbf{A}}^{-1}$ is positive definite. Therefore, $\tilde{\mathbf{A}}^{-1}$ must also be positive definite.

By the definition of a stationary linear solver, we know the result of symmetric Gauss-Seidel satisfies:

$$\mathbf{x}^{(j)}=\mathbf{x}^{*}+\mathbf{G}^{j}(\mathbf{x}^{(0)}-\mathbf{x}^{*})=(\mathbf{I}-\mathbf{G}^{j})\mathbf{A}^{-1}\mathbf{b}, \tag{6}$$

in which $\mathbf{G}$ is the iteration matrix of symmetric Gauss-Seidel and $\mathbf{x}^{*}=\mathbf{A}^{-1}\mathbf{b}$ is the exact solution. By the definition of the Chebyshev semi-iterative method, we have:

$$\begin{aligned}\mathbf{y}^{(K)} &= \sum_{j=0}^{K}v_j(K)\left(\mathbf{x}^{*}+\mathbf{G}^{j}(\mathbf{x}^{(0)}-\mathbf{x}^{*})\right)\\ &= \sum_{j=0}^{K}v_j(K)(\mathbf{I}-\mathbf{G}^{j})\mathbf{A}^{-1}\mathbf{b},\end{aligned} \tag{7}$$

in which $v_j(K)$ is a set of coefficients calculated from Chebyshev polynomials, satisfying $\sum_{j=0}^{K}v_j(K)=1$. Since $(\mathbf{I}-\mathbf{G}^{j})\mathbf{A}^{-1}$ is the smoothing matrix of symmetric Gauss-Seidel and it is positive definite, the joint smoothing matrix given in Equation 7 must be positive definite as well. □

THEOREM 4.2. Suppose that the linear system is positive definite. If the matrices representing the pre-smoothing process and the post-smoothing process at the same level are identical and positive definite, then the matrix representing the whole smoothing process provided by linear multigrid is also positive definite.

PROOF. We prove this by induction. Let $\mathbf{B}^{-1}$ be the matrix representing pre-smoothing and post-smoothing at the h-th level, and $\mathbf{C}_{h+1}^{-1}$ be the matrix representing the smoothing process below the h-th level. After pre-smoothing, we get:

$$\mathbf{x}_h^{(1)}=\mathbf{x}_h^{(0)}+\mathbf{B}^{-1}\mathbf{r}_h^{(0)}, \tag{8}$$

in which $\mathbf{x}_h^{(0)}$ is the initial solution and $\mathbf{r}_h^{(0)}=\mathbf{b}-\mathbf{A}_h\mathbf{x}_h^{(0)}$ is the initial residual at the h-th level. After smoothing below the h-th level, we obtain:

$$\begin{aligned}\mathbf{x}_h^{(2)} &= \mathbf{x}_h^{(1)}+\mathbf{C}_{h+1}^{-1}\left(\mathbf{b}-\mathbf{A}_h\mathbf{x}_h^{(1)}\right)\\ &= \mathbf{x}_h^{(0)}+\left(\mathbf{B}^{-1}+\mathbf{C}_{h+1}^{-1}(\mathbf{B}-\mathbf{A}_h)\mathbf{B}^{-1}\right)\mathbf{r}_h^{(0)}.\end{aligned} \tag{9}$$

Finally after post-smoothing, we obtain:

$$\mathbf{x}_h^{(3)}=\mathbf{x}_h^{(2)}+\mathbf{B}^{-1}\left(\mathbf{b}-\mathbf{A}_h\mathbf{x}_h^{(2)}\right)=\mathbf{x}_h^{(0)}+\mathbf{C}_h^{-1}\mathbf{r}_h^{(0)}, \tag{10}$$

in which $\mathbf{C}_h^{-1}=\mathbf{B}^{-1}+\mathbf{B}^{-1}(\mathbf{B}-\mathbf{A}_h)\left(\mathbf{B}^{-1}+\mathbf{C}_{h+1}^{-1}(\mathbf{B}-\mathbf{A}_h)\mathbf{B}^{-1}\right)$. Since $\mathbf{B}$ and $\mathbf{A}_h$ are positive definite, if $\mathbf{C}_{h+1}^{-1}$ is positive definite, $\mathbf{C}_h^{-1}$ must be positive definite as well. Meanwhile, we know $\mathbf{C}_H^{-1}=\mathbf{A}_H^{-1}$ at the coarsest level, which is positive definite by its definition. Therefore, the statement is true. □

Appendix B: Mesh Hierarchy Generating

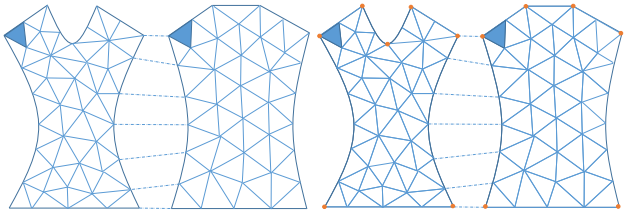
We generate the mesh hierarchy from fine to coarse in the 2D material space at initialization. As shown in the following figures, we take four steps to cosrsen a fine cloth mesh stitched by multiple patches (a):

1 (b): Choose all the critical corner vertices to keep the shape of boundary;

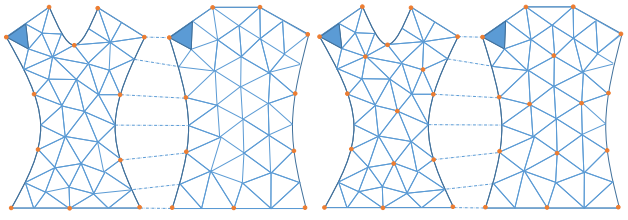
2 (c): Uniformly choose boundary vertices to coarsen the boundaries. If two vertices on different boundaries are stitched, they should be chosen together.

3 (d): Uniformly choose some inner vertices to coarsen each patch.

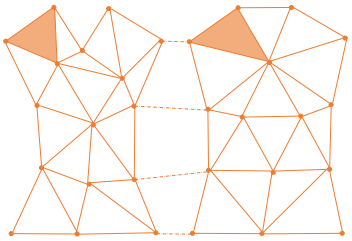
4 (e): Trianglulation with boundary constraints.



(a) (b) Choose Corner Vertices



(c) Choose Boundary Vertices (d) Choose Inner Vertices



(e) Triangulation

© 2018 The Author(s)
Computer Graphics Forum © 2018 The Eurographics Association and John Wiley & Sons Ltd.