

Progressive Inverse Dynamics for Artist-Directed Cloth Simulation

Trey DuBose

CMU-CS-26-125

July 2026

Computer Science Department
School of Computer Science Department
Carnegie Mellon University
Pittsburgh PA 15213

Thesis Committee:

Minchen Li, Chair
James McCann

*Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Computer Science.*

Keywords: Cloth Simulation; Physically Based Animation; Inverse Dynamics; Progressive Dynamics; Numerical Optimization; Computer Graphics; Deformable Body Simulation

Dedicated to the memories of Susie, Stan, and Kali

Abstract

Physically based cloth simulation is widely used to generate realistic animations for films, games, and virtual environments. However, achieving artist-directed cloth motion often requires costly trial-and-error adjustment of simulation parameters, making the process increasingly impractical for high-resolution simulations. This thesis presents a coarse-to-fine framework for artist-directed cloth animation that combines differentiable inverse simulation with Progressive Dynamics. Artistic intent is formulated as a differentiable objective over the simulated cloth trajectory, enabling user-defined goals, such as target poses or motions, to be optimized directly through gradient-based methods.

To improve computational efficiency, inverse optimization is performed over a hierarchy of cloth discretizations. Optimization begins on a coarse representation of the cloth, where large-scale motion can be recovered efficiently, before the optimized state and control parameters are progressively transferred to finer resolutions to reconstruct high-frequency geometric detail. By integrating Progressive Dynamics with differentiable inverse simulation, the proposed framework reduces the computational cost of inverse optimization while preserving physically plausible cloth behavior. Experimental results demonstrate improved optimization efficiency compared to direct high-resolution inverse simulation, providing a more practical and intuitive workflow for artist-directed cloth animation.

Acknowledgments

I would like to express my sincere gratitude to Professor Minchen Li for his outstanding mentorship throughout my research. His guidance, encouragement, and expertise have been invaluable in shaping both this thesis and my development as a researcher. More importantly, he fostered my appreciation for physically based simulation and inspired my continued interest in computer graphics.

I would also like to thank Professor James McCann for his wisdom, thoughtful advice, and support throughout my academic career. His encouragement and perspective have greatly influenced my growth as both a student and researcher.

Finally, I would like to thank my family and friends, both near and far, for their unwavering love, encouragement, and support throughout this journey. Their belief in me, especially during the most challenging moments, gave me the strength and motivation to persevere. This achievement would not have been possible without them, and I am deeply grateful for their presence in my life.

Contents

1	Introduction	1
2	Background and Related Work	3
2.1	Physically-Based Cloth Simulation	3
2.2	Differentiable and Inverse Simulation	3
2.3	Artist-Controlled Physical Simulation	4
2.4	Progressive and Hierarchical Numerical Methods	4
2.5	Motivation	5
3	Method	7
3.1	Overview	7
3.2	Forward Simulation	7
3.2.1	Progressive Dynamics	8
3.2.2	Artist Objectives	10
3.2.3	Position-Based Objectives	11
3.3	Inverse Optimization	11
3.3.1	Gradient Computation	11
3.3.2	Parameter Dependent Sensitivity	12
3.3.3	Timestep Jacobian	13
3.4	Unified Pipeline	13
4	Implementation	15
4.1	Software Architecture	15
4.2	Differential Inverse Simulation	15
4.3	Progressive Optimization Pipeline	16
5	Results	17
5.1	Experimental Setup	17
5.2	Performance Comparison	18
5.3	Convergence Analysis	18
5.4	Effect of Hierarchy Depth	19
5.5	Visual Results	20
5.5.1	Swinging Cloth	20
5.5.2	Cloth Toss	21

6	Limitations & Future Work	23
6.1	Limitations	23
6.2	Future Work	23
7	Conclusion	25
A	Proof of Timestep Jacobian	27
	Bibliography	29

List of Figures

5.1	Swinging cloth optimization results. Gray colored cloth depicts intermediate frames and the blue cloth is the final frame reaching the target positions in red. . .	20
5.2	Cloth toss progression comparing initial and optimized results. Gray colored cloth depicts intermediate frames and the blue cloth is the final frame reaching the target positions in red.	21

List of Tables

- 5.1 Runtime comparison across benchmark scenarios, between direct inverse optimization and our proposed Progressive Dynamics framework. 18
- 5.2 Runtime comparison across benchmark scenarios, between direct inverse optimization and our proposed Progressive Dynamics framework. 18
- 5.3 Comparison of optimization algorithms. 19
- 5.4 Effect of hierarchy depth on optimization performance. 19

Chapter 1

Introduction

Cloth simulation is a fundamental component of modern computer graphics, enabling realistic digital garments and dynamic secondary motion in animation. However, traditional forward simulation provides limited artistic control, requiring artists to manually tune physical parameters or rely on keyframing to achieve desired motion. While inverse simulation addresses this limitation by formulating animation as an optimization problem over user-defined objectives, its computational cost remains a significant barrier due to repeated physics solves over high-dimensional cloth states.

This thesis presents a coarse-to-fine framework for artist-directed cloth animation that combines differentiable inverse simulation with progressive dynamics. By optimizing high-level artistic objectives at reduced resolutions and progressively refining the resulting motion, the proposed framework reduces the computational burden of inverse optimization while preserving physically plausible cloth behavior. This approach enables artists to directly specify desired cloth trajectories and motions while leveraging physical simulation to produce realistic results.

The primary contributions of this thesis are:

- An inverse simulation formulation that represents artistic intent as differentiable objectives over cloth trajectories.
- A progressive dynamics framework that performs coarse-to-fine refinement to improve the efficiency of inverse optimization.
- A unified optimization pipeline that integrates low-resolution inverse simulation with progressive refinement to recover high-resolution detail.
- An experimental evaluation demonstrating improvements in optimization efficiency while maintaining the resulting quality of cloth.

Chapter 2

Background and Related Work

2.1 Physically-Based Cloth Simulation

Physically based cloth simulation seeks to reproduce the behavior of deformable fabrics by numerically solving the governing equations of motion. Early work by Baraff and Witkin in **Large steps in cloth simulation** [1] established many of the foundations of modern cloth simulation through implicit time integration, enabling significantly larger timesteps than explicit methods while maintaining numerical stability. Their formulation demonstrated that robust cloth simulation could be posed as a nonlinear optimization problem solved at every timestep.

Subsequent work has focused on improving both efficiency and robustness. **Projective Dynamics**[3] reformulates simulation as an energy minimization problem using alternating local and global optimization steps, allowing many elastic materials to be simulated efficiently while retaining unconditional stability. **Extended Position-Based Dynamics (XPBD)**[10] further improves robustness by introducing compliant constraints that remain independent of timestep size, making the method particularly attractive for interactive applications. More recently, **Incremental Potential Contact (IPC)**[6] along with the extension into elastodynamics (C-IPC)[7], incorporated collision handling directly into the optimization problem through barrier energies, enabling robust, intersection-free simulations for thin shells and cloth undergoing complex contact.

Together, these methods have significantly advanced forward cloth simulation by improving stability, computational efficiency, and collision handling. However, they assume that the physical parameters governing the simulation are known and therefore provide only limited support for directly specifying desired artistic outcomes.

2.2 Differentiable and Inverse Simulation

Many graphics applications require solving the inverse problem: determining the simulation parameters that produce a desired animation. Differentiable simulation addresses this challenge by exposing gradients of simulation outputs with respect to physical parameters, allowing gradient-based optimization to be applied directly to physical simulation.

Recent work such as **DiffCloth**[9] and **DiffPD**[5] has demonstrated that optimization-based

cloth simulators can be differentiated efficiently while preserving physically meaningful behavior. These methods enable applications including parameter estimation, trajectory optimization, and learning-based control by propagating gradients through the simulation process. More broadly, differentiable simulation has become a powerful framework for inverse problems involving deformable objects.

Despite these advances, inverse simulation remains computationally demanding. Each optimization iteration typically requires repeated forward simulations together with gradient evaluations, making high-resolution cloth optimization prohibitively expensive for many practical applications.

2.3 Artist-Controlled Physical Simulation

In animation and visual effects, artists are often more interested in achieving a desired visual result than reproducing exact physical behavior. Rather than specifying material parameters or external forces directly, artists typically describe target poses, silhouettes, or motion that the simulation should reproduce.

Early work addressed this problem through space-time optimization, which computes physically consistent trajectories satisfying user-defined objectives over an entire animation sequence. More recent approaches formulate artist constraints as optimization objectives, allowing sparse positional constraints or keyframes to guide physically based simulation[8][12]. These techniques provide intuitive control while preserving physical plausibility, but they continue to rely on repeated nonlinear optimization and therefore inherit the computational cost of inverse simulation.

2.4 Progressive and Hierarchical Numerical Methods

Hierarchical numerical methods[2][13][4] have long been used to accelerate large-scale optimization by solving simplified problems before progressively introducing additional degrees of freedom. Multigrid methods[11][14], for example, eliminate low-frequency error on coarse meshes before refining the solution on finer resolutions, substantially reducing computational cost.

Building on this idea, **Progressive Dynamics**[15][16] introduces a hierarchy of optimization problems for cloth and shell simulation. Instead of solving the complete simulation at the finest resolution, the method first computes coarse-scale motion and then progressively reconstructs higher-frequency geometric detail using prolongation operators. This coarse-to-fine optimization significantly reduces the number of expensive Newton iterations required at the finest mesh while maintaining high simulation accuracy.

Although Progressive Dynamics has proven highly effective for accelerating forward simulation, its application has thus far been limited to solving forward dynamics.

2.5 Motivation

The development of cloth simulation has progressed from robust implicit integration methods to optimization-based solvers capable of handling increasingly complex materials and contact. More recently, differentiable simulation has enabled gradient-based optimization for inverse problems, while artist-directed simulation has demonstrated the value of high-level control over physically based animation. At the same time, progressive numerical methods have shown that hierarchical optimization can substantially reduce computational cost for large simulation problems.

Despite these advances, these research directions have largely evolved independently. Modern cloth simulators emphasize physical robustness, differentiable simulators enable optimization, and artist-control techniques provide intuitive interfaces, yet inverse optimization remains computationally expensive for high-resolution simulations. This observation motivates the central question of this thesis: **can progressive dynamics be integrated with inverse cloth simulation to provide efficient, artist-directed control while preserving physical plausibility?**

The remainder of this thesis presents a unified framework that combines progressive dynamics with inverse optimization, enabling efficient artistically directed cloth simulation while maintaining the robustness of modern physically based simulation methods.

Chapter 3

Method

3.1 Overview

The proposed framework combines differentiable inverse simulation with Progressive Dynamics to enable efficient artist-directed cloth optimization. Given a set of user-defined control parameters, the framework first performs a differentiable forward simulation that computes the cloth trajectory together with the gradients of the simulated motion with respect to the optimization parameters. These sensitivities are then used within a gradient-based optimizer to minimize an artist-defined objective by iteratively updating the simulation parameters.

Rather than performing this optimization directly on the finest-resolution cloth mesh, the framework adopts a coarse-to-fine optimization strategy based on Progressive Dynamics. Optimization begins on a coarse simulation where the dominant, low-frequency cloth motion is efficiently recovered. Once convergence is achieved, both the optimized simulation state and control parameters are transferred to the next refinement level, where additional geometric detail is introduced. This process continues through the hierarchy until the finest-resolution simulation is reached, producing a physically plausible cloth animation that satisfies the desired artistic objectives while significantly reducing the computational cost of inverse optimization.

Throughout this chapter, let

$$x = [x_1, x_2, \dots, x_N] \in \mathbb{R}^{3N}$$

denote the vector of all cloth vertex positions, where $(x_i \in \mathbb{R}^3)$ is the position of the (i) -th vertex. The superscript (x^k) denotes the configuration at timestep (k) . We further denote the diagonal lumped mass matrix by (M) and the timestep size by (h) .

3.2 Forward Simulation

At each timestep, cloth motion is computed by minimizing a discrete energy functional that combines inertial, elastic, contact, frictional, and external force terms. Given the predicted inertial position

$$\hat{x}^k = x^k + hv^k + h^2M^{-1}f_{ext}, \quad (3.1)$$

the total energy is defined as

$$\frac{1}{2}\|x - \hat{x}^k\|_M^2 + B(x) + D(x) + S(x) + \Psi(x) + \Phi(x), \quad (3.2)$$

where:

- $(\frac{1}{2}\|x - \hat{x}^k\|_M^2)$ is the inertial energy,
- $(B(x))$ is the barrier contact energy,
- $(D(x))$ is the frictional dissipation term,
- $(S(x))$ is a strain-limiting energy used to prevent excessive stretching,
- $(\Psi(x))$ is the membrane or shell energy, and
- $(\Phi(x))$ is the bending energy.

For ease of notation we will denote the sum of all non-inertia energies as,

$$E(x) = B(x) + D(x) + S(x) + \Psi(x) + \Phi(x), \quad (3.3)$$

The next timestep configuration is then obtained by solving the nonlinear optimization problem

$$x^{k+1} = \arg \min_x \frac{1}{2}\|x - \hat{x}^k\|_M^2 + E(x). \quad (3.4)$$

In practice, this minimization is solved using Newton iterations. At iteration (k) , the search direction (p) is obtained from

$$H p = -g, \quad (3.5)$$

where g is the gradient (first derivative) and H is the Hessian (second derivative) of all energies with respect to x .

Then the configuration is updated as

$$x^{k+1} = x^k + \alpha p, \quad (3.6)$$

where (α) is determined by a line-search procedure. This process is repeated until convergence of $\frac{\|p\|}{h} < \epsilon$.

3.2.1 Progressive Dynamics

Although the forward simulation described above produces physically accurate cloth motion, solving the full optimization problem at the finest mesh resolution is computationally expensive. In particular, the membrane and bending energies dominate the cost of each Newton iteration due to their large number of degrees of freedom and highly nonlinear behavior. Progressive Dynamics[15] addresses this challenge by solving a hierarchy of progressively refined optimization problems, allowing low-frequency deformations to be resolved on coarse meshes before introducing higher-frequency geometric detail on finer levels.

Let L denote the finest resolution of the cloth mesh and let $l \in \{0, \dots, L\}$ denote a hierarchy of progressively finer meshes. The state of the cloth at level l is represented by the nodal position vector x_l . A prolongation operator projects the solution from a coarse level l to the next finer level $l + 1$ according to

$$P_{l+1}^l(x_l) = U_{l+1}^l x_l + a_{l+1}^l(x_l), \quad (3.7)$$

where U_{l+1}^l is a linear intrinsic interpolation operator and $a_{l+1}^l(\cdot)$ is a nonlinear extrinsic offset that reconstructs geometric detail lost during coarsening.

These prolongation operators satisfy a compositional property such that the projection from any level l to the finest level L can be written as

$$P_L^l = P_L^{L-1} \circ P_{L-1}^{L-2} \circ \dots \circ P_{l+1}^l, \quad (3.8)$$

where $\circ$ denotes function composition. For notational simplicity, the composite projection to the finest level is denoted by $P^l = P_L^l$ throughout the remainder of this thesis.

Unlike geometric multigrid methods, Progressive Dynamics does not simply solve identical optimization problems on multiple meshes. Instead, each level minimizes a *proxy energy* that separates coarse-scale global effects from fine-scale cloth deformation. Global energies such as inertia, contact, friction, and strain limiting are evaluated directly on the current level, while the membrane and bending energies are evaluated after prolongating the solution to the finest mesh. The resulting proxy energy is given by

$$F_l(x_l) = B_l(x_l) + D_l(x_l) + S_l(x_l) + \Psi(P^l(x_l)) + \Phi(P^l(x_l)), \quad (3.9)$$

where B_l , D_l , and S_l denote the barrier contact, friction, and strain-limiting energies evaluated on level l , while Ψ and Φ represent the membrane and bending energies evaluated on the reconstructed fine-resolution mesh.

The optimization problem at each level is therefore formulated as

$$x_l^{t+1} = \arg \min_x \frac{1}{2} \|x - \hat{x}_l^t\|_{M_l}^2 + F_l(x), \quad (3.10)$$

where h is the timestep size, M_l is the lumped mass matrix at level l , and $\hat{x}_l^t$ denotes the inertial prediction used to initialize the optimization.

Several strategies have been proposed for computing the inertial prediction during progressive refinement. Throughout this work we adopt the *velocity prolongation* strategy[16], which transfers the coarse-level velocity to initialize the next finer level. The predicted position is computed as

$$\hat{x}_{l+1}^t = x_{l+1}^t + P_{l+1}^l(x_l^t - x_l^{t-1}), \quad (3.11)$$

thereby preserving the large-scale motion already computed on the coarser mesh while allowing finer geometric detail to emerge during subsequent optimization.

For Newton's method, the gradient and Hessian of the proxy energy must be projected from the fine degrees of freedom to the corresponding coarse representation. Since the prolongation operator contains both linear and nonlinear components, its Jacobian is given by

$$J_P(x_l) = U + \frac{\partial a}{\partial x_l}, \quad (3.12)$$

where U denotes the linear interpolation operator and $a(x_l)$ is the nonlinear correction.

Applying the chain rule, the projected gradients of the proxy membrane and bending energies become

$$\nabla \Psi(x_l) = J_P(x_l)^T \nabla \Psi(P(x_l)), \quad \nabla \Phi(x_l) = J_P(x_l)^T \nabla \Phi(P(x_l)), \quad (3.13)$$

Approximating the projected Hessians by neglecting the second derivatives of the nonlinear correction yields

$$\nabla^2 \Psi(x_l) \approx J_P(x_l)^T \nabla^2 \Psi(P(x_l)) J_P(x_l), \quad \nabla^2 \Phi(x_l) \approx J_P(x_l)^T \nabla^2 \Phi(P(x_l)) J_P(x_l), \quad (3.14)$$

which corresponds to a Gauss-Newton approximation of the exact projected Hessian. When the prolongation operator is purely linear, these expressions reduce to the familiar multigrid projections using the interpolation matrix alone.

3.2.2 Artist Objectives

Traditional cloth simulation is designed to reproduce physically plausible motion given a set of material properties, external forces, and initial conditions. However, in an animation workflow, artists typically specify desired outcomes rather than physical parameters. For example, an artist may wish to create a specific silhouette, place a fold at a particular location, or match the motion of a keyframe. Directly controlling the physical parameters that produce these behaviors is often unintuitive and requires extensive manual tuning.

To bridge the gap between physical simulation and artistic intent, we formulate artist input as an objective function defined over the simulated cloth state. Rather than directly prescribing the cloth trajectory, the artist specifies constraints that describe the desired behavior of the simulation. The optimizer then determines the physical parameters required to satisfy these objectives while maintaining the physical constraints imposed by the forward simulation.

Let θ denote the set of controllable simulation parameters, including initial velocities, external forces, material parameters, and other user defined controls. The resulting cloth trajectory is generated through the forward simulation process described in Section 3.2, such that

$$\{x^0, x^1, \dots, x^T\} = \mathcal{S}(\theta), \quad (3.15)$$

where $\mathcal{S}$ represents the complete cloth simulation pipeline.

The artist objective is defined as a loss function that measures the difference between the simulated result and the desired behavior. The general optimization objective is

$$L(\theta) = L_{\text{artist}}(\mathcal{S}(\theta)) + \lambda R(\theta), \quad (3.16)$$

where L_{artist} measures the discrepancy between the simulated cloth and the artist-defined target, and $R(\theta)$ is a regularization term that discourages physically implausible parameter values. The weighting parameter λ controls the tradeoff between satisfying the artistic objective and maintaining physically reasonable simulation parameters.

Parameters controlled by $R(\theta)$ include factors such as: velocity to prevent contact jumping, strengthening or lessening membrane constraints, and preventing unrealistic parameter values.

3.2.3 Position-Based Objectives

One common form of artist control is specifying desired cloth positions at a particular timestep. Let x^* denote the target configuration provided by the artist. The positional objective is defined as

$$L_{\text{position}} = \frac{1}{2} \|x^n - x^*\|_W^2, \quad (3.17)$$

where W is a weighting matrix that allows the artist to specify the relative importance of different vertices. This formulation allows partial constraints, where only a subset of vertices or regions of the cloth are required to match the target.

By expressing artistic intent as differentiable objectives, the inverse simulation framework (defined below) can automatically determine the physical parameters required to produce the desired behavior. The following section describes the optimization procedure used to minimize these objectives.

3.3 Inverse Optimization

While forward simulation computes cloth motion from known parameters, the goal of inverse simulation is to determine parameters (θ) that produce a desired animation. The parameter vector may include initial velocities, wind parameters, material properties, or other control variables.

To mimic the traditional keyframing workflow, the initial configuration and any prescribed initial conditions are fixed. A target objective is then specified. We define the loss function like Equation 3.17. Although the loss can be evaluated over multiple timesteps, we found that evaluating it at a small number of keyframes provides a good balance between controllability and computational cost.

The desired cloth behavior is formulated as a nonlinear optimization problem in which the simulation parameters are chosen to minimize an artist-defined objective while respecting the governing physical dynamics.

$$\theta^* = \arg \min_{\theta} L(\theta). \quad (3.18)$$

3.3.1 Gradient Computation

To optimize the parameters, gradients of the loss with respect to (θ) are required. Since the cloth trajectory is generated through a sequence of simulation steps, the gradients are computed by

propagating sensitivities backward through time.

For parameters that affect only the initial state, such as the initial velocity, the chain rule gives

$$\frac{dL}{d\theta} = \frac{dL}{dx^n} \frac{dx^n}{dx^{n-1}} \cdots \frac{dx^1}{dx^0} \frac{dx^0}{d\theta}. \quad (3.19)$$

This expression shows that the sensitivity of the loss at the final timestep is propagated backward through every simulation step, analogous to backpropagation in neural networks.

Define the trajectory sensitivity

$$G^k = \frac{dx^k}{d\theta}. \quad (3.20)$$

For parameters that influence only the initial state, the sensitivities satisfy the recurrence

$$G^{k+1} = \frac{\partial x^{k+1}}{\partial x^k} G^k. \quad (3.21)$$

For general parameters that directly affect each timestep, the recurrence becomes

$$G^{k+1} = \frac{\partial x^{k+1}}{\partial x^k} G^k + \frac{\partial x^{k+1}}{\partial \theta}. \quad (3.22)$$

The first term propagates sensitivities through the dynamics, while the second term accounts for the direct dependence of the timestep solve on the parameters.

Intuitively, this recurrence propagates the influence of each optimization parameter through the simulated trajectory. Each timestep contributes both an indirect effect through the evolving cloth state and a direct effect arising from the parameter itself.

3.3.2 Parameter Dependent Sensitivity

The trajectory sensitivity depends on the choice of optimization parameters. Different parameterizations influence the simulation at different points in time and therefore require different initial conditions for the sensitivity recurrence.

When optimizing the initial velocity, the initial cloth configuration (x^0) is fixed while the predicted inertial position for the first timestep depends directly on the optimization variables. Consequently, the initial state itself is independent of the parameters,

$$\frac{dx^0}{d\theta} = 0, \quad (3.23)$$

and the first nonzero sensitivity appears during the initial simulation step,

$$\frac{\partial x^1}{\partial \theta}, \quad (3.24)$$

since the initial velocity contributes directly to the predicted inertia position of the first time step $\hat{x}^0$ (from Equation 3.1). After this first timestep, the parameter influence is propagated through the remaining trajectory using the sensitivity recurrence described in Equation 3.22.

Algorithm 1 Differentiable Forward Simulation

Require: Initial state x^0 , parameters θ

Ensure: : Simulation trajectory $\{x^0, \dots, x^T\}$ and backwards sensitivity gradient G^T

- 1: Initialize sensitivity $G^0 = \frac{dx^0}{d\theta}$ (Eq. 3.23)
 - 2: **for** $t = 0$ to T **do**
 - 3: Initialize $\hat{x}^{t+1}$ (Eq. 3.11)
 - 4: Advance simulation step: $x^{t+1} = \arg \min_x \frac{1}{2} \|x - \hat{x}^{t+1}\|_M^2 + F(x)$ (Eq. 3.10)
 - 5: Update sensitivity: $G^{t+1} = \frac{\partial x^{t+1}}{\partial x^t} G^t + \frac{\partial x^{t+1}}{\partial \theta}$ (Eq. 3.22)
 - 6: **end for**
 - 7: **return** $\{x^0, \dots, x^T\}, G^T$
-

In contrast, parameters such as wind forces do not modify the initial cloth configuration or prescribed initial conditions. Instead, they act as external forces during the simulation and therefore contribute directly to every timestep through the term

$$\frac{\partial x^{k+1}}{\partial \theta}, \quad (3.25)$$

while the initial sensitivity remains unchanged from Equation 3.23.

More generally, optimization parameters that modify only the initial conditions produce a single direct sensitivity contribution during initialization, whereas parameters that influence the governing dynamics contribute throughout the entire simulated trajectory.

3.3.3 Timestep Jacobian

The remaining quantity required for gradient computation is the Jacobian of a single timestep with respect to the previous state.

$$\frac{\partial x^{k+1}}{\partial x^k} = [M + \nabla^2 E(x^{k+1})]^{-1} M, \quad (3.26)$$

The proof is provided in Appendix A. Intuitively, the Hessian $(M + \nabla^2 E(x^{k+1}))$ describes the local stiffness of the energy landscape, and its inverse determines how perturbations at one timestep influence the next.

Substituting this Jacobian into the sensitivity recurrences provides the complete gradient computation framework. The resulting gradients are then used within a gradient-based optimizer, such as gradient descent, ADAM, or L-BFGS, to iteratively update (θ) and minimize the target loss.

3.4 Unified Pipeline

The proposed framework treats simulation resolution as an additional optimization dimension. Rather than solving the inverse optimization directly on the finest-resolution simulation, we re-

Algorithm 2 Coarse-to-Fine Progressive Inverse Simulation

Require: Initial state x^0 , hierarchy $\{0, \dots, L\}$, initial parameters θ_0 , artist objective L

Ensure: Optimized parameters θ^*

```
1: Initialize  $\theta_0$ 
2: for  $l = 0, \dots, L$  do
3:   if  $l > 0$  then
4:     Initialize  $\theta_l \leftarrow \theta_{l-1}$ 
5:   end if
6:   repeat
7:     Simulate trajectory  $(\{x_l^0, \dots, x_l^T\}, G^T) = \mathcal{S}_l(x_l^0, \theta_l)$ 
8:     Evaluate objective  $\mathcal{L}_l = L(\theta_l)$ 
9:     Compute gradient using inverse simulation  $g = \frac{d\mathcal{L}_l}{d\theta_l} = \frac{d\mathcal{L}_l}{dx_l^T} G^T$ 
10:    Update parameters  $\theta_l \leftarrow \theta_l - \eta g$ 
11:   until  $\mathcal{L}_l < \varepsilon_l$ 
12: end for
13: return  $\theta_L$ 
```

formulate the optimization as a hierarchy of progressively refined problems. At the coarsest level, only the dominant, low-frequency cloth motion is optimized. Once convergence is achieved, the optimized parameters and cloth state are transferred to the next resolution using the Progressive Dynamics prolongation operator. Each subsequent level refines both the simulation and the optimization variables, allowing higher-frequency wrinkles and fine-scale deformation to emerge while requiring only a small number of additional optimization iterations. This coarse-to-fine strategy significantly reduces the cost of inverse simulation by avoiding expensive optimization on the full-resolution mesh until a near-optimal solution has already been obtained. By resolving global motion before local deformation, Progressive Dynamics significantly reduces the number of optimization iteration required at the finest resolution while producing results that closely match a direct optimization on the full mesh.

Chapter 4

Implementation

With the methods of this thesis clearly defined, this section will describe the engineering decisions, software architecture, optimization settings and experimental setup for the purpose of analyzing performance and reproducibility of results.

4.1 Software Architecture

The proposed framework was implemented by extending an existing C++ Incremental Potential Contact (IPC) simulator developed by the Simulation Intelligence Group (SIG) led by Minchen Li at Carnegie Mellon University. The original simulator performs implicit cloth simulation using Newton’s method with Incremental Potential Contact, barrier energies, membrane elasticity, and bending energies. This work extends the simulator by incorporating wind forces, differentiable inverse optimization, and a hierarchical coarse-to-fine optimization framework based on Progressive Dynamics.

The simulator employs sparse linear algebra throughout the optimization pipeline to efficiently solve the large systems arising from implicit integration. Sparse matrices are used for Hessians, Jacobians, and projection operators, while dense vectors store vertex positions, velocities, and optimization parameters. This representation substantially reduces both memory usage and computational cost compared to dense formulations.

4.2 Differential Inverse Simulation

The forward simulation loop was extended to compute trajectory sensitivities with respect to the optimization variables. During each implicit timestep, the Jacobian described in Equation (3.26) is computed from the timestep Hessian already assembled for the Newton solve. Rather than explicitly constructing dense Jacobian matrices, the implementation reuses the sparse Hessian factorization produced during forward simulation, significantly reducing the computational overhead of gradient evaluation.

The trajectory sensitivity (3.20) is stored as a dense matrix of dimension $|x| \times |\theta|$. Following each timestep, the sensitivity is updated using the recurrence defined in Equation (3.22), allowing

gradients to be accumulated throughout the simulated trajectory without storing intermediate Jacobians.

After the final timestep, the gradient of the artist objective is multiplied by the accumulated trajectory sensitivity to obtain the parameter gradient $\frac{dL}{d\theta}$. This gradient is then passed to a gradient-based optimizer. Although the framework is compatible with gradient descent and Adam, the optimizer used in each experiment is specified in Section 3.3.1.

4.3 Progressive Optimization Pipeline

Cloth representations are built at runtime for the finest level mesh and all lower level meshes. In addition to that, all projection matrices between neighboring levels l and $l + 1$ as well as between each level l and the finest level L creating both P_{l+1}^l and P_L^l which are necessary for all proxy representations and the velocity prolongation advancement. Projections are stored as a from-to index pair and a weight value.

Optimization begins on the coarsest mesh in the hierarchy. Once the optimization converges the optimized parameters (i.e. wind, initial velocity) are transferred directly to the next refinement level. The cloth state is initialized using the Progressive Dynamics prolongation operator, allowing the finer optimization to begin from an already physically plausible trajectory.

Since membrane and bending energies are always evaluated on the reconstructed finest-resolution mesh, geometric quantities such as the rest shape, element areas, and material parameters are computed once during initialization and reused throughout optimization. As a result, only the projected vertex positions change between optimization iterations, reducing the computational cost associated with repeatedly assembling material data.

To evaluate the proxy Hessian (3.14), the internally stored projection operators are converted into Eigen sparse matrices. This allows the coarse-space Hessian projection to be computed using Eigen’s optimized sparse matrix multiplication routines while maintaining compatibility with the remainder of the simulator.

Because velocity prolongation (3.11) depends on cloth positions from consecutive timesteps, the simulator stores the complete trajectory at every hierarchy level throughout optimization. Although this increases memory consumption, it eliminates redundant forward simulations when initializing finer-resolution levels.

Chapter 5

Results

This chapter evaluates the proposed Progressive Inverse Dynamics framework through a series of benchmark cloth simulation scenarios. The experiments are designed to measure the computational efficiency of the proposed coarse-to-fine optimization strategy, compare optimization algorithms, investigate the effect of hierarchy depth, and evaluate the visual quality of the resulting cloth animations.

5.1 Experimental Setup

The proposed framework was implemented by extending the simulator described in Chapter 4. All experiments were performed using identical cloth representations and simulation parameters (Table 5.1) for both the baseline inverse simulation and the proposed Progressive Dynamics approach in order to ensure a fair comparison.

Optimization was performed using gradient-based methods with differentiable objectives defined over the simulated cloth trajectory. Multiple optimization algorithms and learning rates were evaluated to study convergence behavior and optimization stability. The optimization terminated after a fixed number of iterations or once the objective function converged below a specified tolerance.

Three benchmark scenarios were selected to evaluate the framework.

- **Swinging Cloth.** A rectangular cloth pinned along one edge is subjected to optimized wind forces. The objective is to determine the wind parameters that produce a desired resting configuration corresponding to a target angle.
- **Cloth Toss.** The initial velocity of the cloth is optimized so that the simulated trajectory reaches a prescribed target position while maintaining physically plausible dynamics.
- **Cloth Flip.** A challenging nonlinear benchmark in which the cloth must rise, rotate about a single axis, and return to its initial location while completing a full flip.

Performance is evaluated using the following metrics:

- Total optimization time
- Progressive refinement time
- Total runtime

Table 5.1: Runtime comparison across benchmark scenarios, between direct inverse optimization and our proposed Progressive Dynamics framework.

N	faces	Levels	density	Membrane Stretch	Membrane Compress	Membrane Bend
9216	0	3	0	10^4	100	10

Table 5.2: Runtime comparison across benchmark scenarios, between direct inverse optimization and our proposed Progressive Dynamics framework.

Scene	PD Opt(s)	PD Refine(s)	(Ours) PD Opt + Refine(s)	Direct Opt(s)	Speedup
Swinging Cloth	72.49	26.2	98.69	143.28	1.451
Cloth Toss	38.94	25.3	64.24	82.3	1.281
Cloth Flip	208.72	25.7	234.42	269.1	1.147

- Number of optimization iterations
- Visual agreement with the target motion

5.2 Performance Comparison

The primary objective of this experiment is to evaluate the computational efficiency of Progressive Inverse Dynamics relative to direct high-resolution inverse optimization.

For each benchmark scenario, optimization was performed both with and without Progressive Dynamics. The baseline method optimized directly on the highest resolution simulation mesh, whereas the proposed framework used 3 hierarchy levels and first optimized on the coarsest representation before progressively refining the solution across successively finer meshes.

Table 5.2 compares the runtime of both approaches, including optimization time, progressive refinement time, and total execution time.

The experimental results indicate that the proposed coarse-to-fine optimization strategy substantially reduces the computational cost of inverse simulation. Although Progressive Dynamics introduces additional projection and refinement operations, these costs remain relatively small compared to the optimization performed on high-resolution meshes.

Furthermore, optimization is largely completed on the coarsest level of the mesh hierarchy. Higher-resolution levels primarily perform forward simulation and velocity prolongation using the optimized parameters, eliminating the need for additional expensive optimization at every refinement level.

5.3 Convergence Analysis

To evaluate optimization robustness, several first-order optimization methods were compared using a range of learning rates. The objective of this experiment is to determine which optimization strategy provides the fastest and most stable convergence for inverse cloth simulation.

Table 5.3 demonstrates how the learning rate affects the optimization iterations required for each optimizer to converge.

Table 5.3: Comparison of optimization algorithms.

Optimizer	Learning Rate	Iterations till Convergence
GD	1.5	10
GD	1.25	6
GD	1	11
GD	0.5	21
GD	0.25	22
GD	0.1	28
ADAM	1.5	8
ADAM	1.25	5
ADAM	1	7
ADAM	0.5	11
ADAM	0.25	16
ADAM	0.1	29

Table 5.4: Effect of hierarchy depth on optimization performance.

Levels	Optimization Time (s)	PD Time (s)	Total Time (s)
0	73.89	0	73.89
1	48.95	8.556	57.506
2	28.32	20.89	49.21
3	19.12	25.3	44.42
4	21.021	28.34	49.361
5	20.42	31.23	51.65

Across all benchmark problems, both Gradient Descent and Adam successfully reduced the objective function. Larger learning rates generally accelerated early-stage convergence by taking larger optimization steps toward the target solution. However, aggressive learning rates occasionally produced oscillatory behavior or overshoot the optimum, particularly during later optimization stages.

Overall, the experiments demonstrate the trade-off between convergence speed and optimization stability. The most appropriate optimizer depends on the complexity of the objective function and the desired balance between robustness and computational efficiency.

5.4 Effect of Hierarchy Depth

The proposed framework introduces an additional design parameter through the number of refinement levels used during optimization. To investigate the impact of hierarchy depth, experiments were conducted using varying numbers of progressive refinement levels.

Table 5.4 compares total runtime as a function of the number of hierarchy levels.

The results suggest that increasing the hierarchy depth initially improves optimization efficiency by reducing the computational complexity of the coarse optimization stage. However,

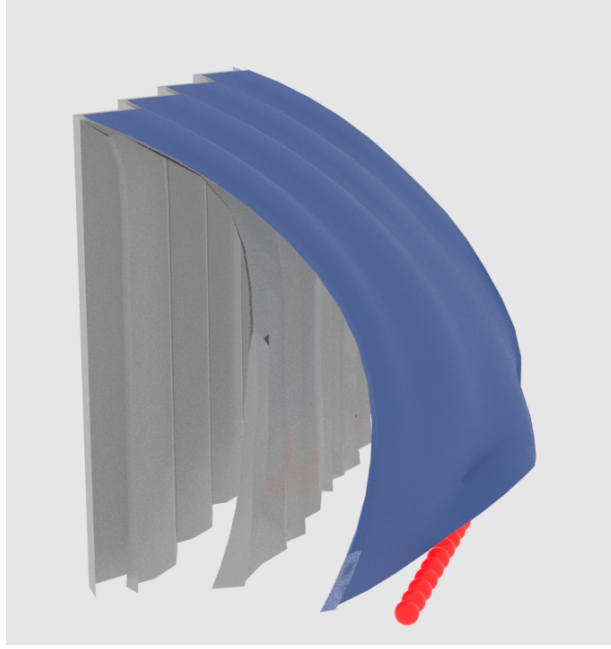


Figure 5.1: Swinging cloth optimization results. Gray colored cloth depicts intermediate frames and the blue cloth is the final frame reaching the target positions in red.

beyond a certain number of refinement levels, additional performance gains diminish as the cost of projection and proxy energy evaluation begins to dominate the total runtime.

These observations indicate that an intermediate hierarchy depth provides the best compromise between optimization efficiency and refinement overhead.

5.5 Visual Results

In addition to computational performance, the proposed framework was evaluated qualitatively by comparing the resulting cloth trajectories with the desired artist-defined objectives.

5.5.1 Swinging Cloth

The swinging cloth experiment demonstrates optimization of wind parameters to achieve a prescribed equilibrium configuration. Figure 5.1 illustrates the optimized wind force that pushes the cloth to the target state.

The optimized coarse solution successfully captures the desired large-scale motion, while Progressive Dynamics reconstructs the fine geometric detail without requiring additional optimization.

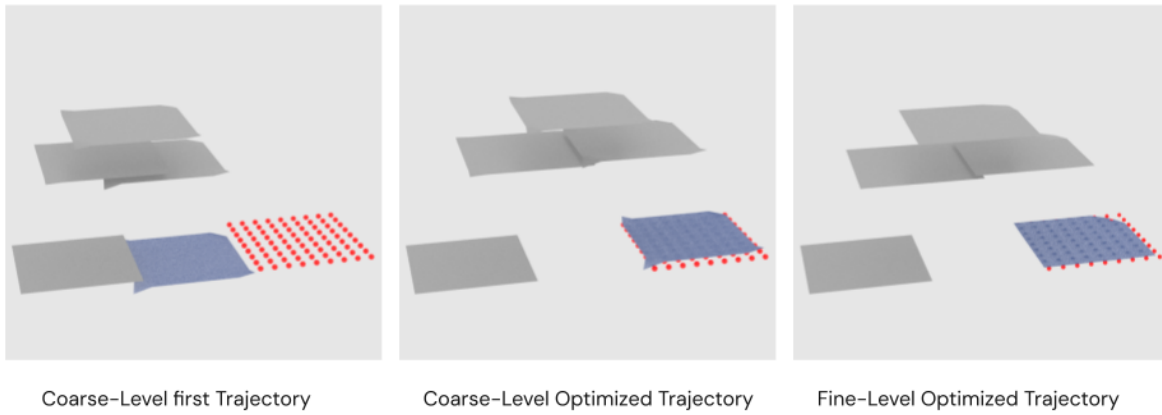


Figure 5.2: Cloth toss progression comparing initial and optimized results. Gray colored cloth depicts intermediate frames and the blue cloth is the final frame reaching the target positions in red.

5.5.2 Cloth Toss

The cloth toss benchmark evaluates optimization of the initial velocity required to achieve a desired trajectory.

Figure 5.2 presents representative simulation frames. Gray meshes indicate intermediate trajectory states, while the blue mesh represents the final optimized configuration.

Although optimization is performed on a coarse simulation mesh, the transferred solution closely matches the desired trajectory after refinement, demonstrating that large-scale motion can be effectively optimized at reduced resolution.

Chapter 6

Limitations & Future Work

6.1 Limitations

Although the proposed Progressive Inverse Dynamics framework demonstrates significant improvements in optimization efficiency, several limitations remain.

The first limitation arises from the hierarchical approximation used throughout the optimization process. While optimizing on coarse simulation meshes greatly reduces computational cost, the reduced representation cannot fully capture the high-frequency wrinkles and fine geometric detail present in the final high-resolution cloth. Although Progressive Dynamics successfully reconstructs much of this detail during refinement, approximation error introduced at coarse levels may occasionally lead to small deviations from the optimal high-resolution solution.

A second limitation stems from the optimization problem itself. Artist-directed cloth simulation is inherently nonlinear and nonconvex, causing the optimization landscape to contain numerous local minima. Consequently, the quality of the resulting animation depends on both the initial parameter estimates and the selected optimization algorithm. Gradient-based methods may converge to suboptimal solutions or require careful tuning of learning rates to maintain stable convergence.

Finally, the current framework assumes that all artist objectives are differentiable. This restriction enables efficient gradient computation but limits the range of objectives that can be optimized directly. More complex animation goals involving discrete events, logical constraints, or non-differentiable artistic objectives cannot currently be represented within the optimization framework without additional approximations.

Despite these limitations, the proposed framework demonstrates that combining inverse simulation with Progressive Dynamics provides a practical and efficient approach for artist-directed cloth animation.

6.2 Future Work

The results presented in this thesis suggest several promising directions for future research.

One important extension is the development of an interactive user interface that allows artists to define target motions, manipulate optimization objectives, and visualize the resulting simu-

lations in real time. Such a system would enable practical evaluation of the framework within professional animation pipelines while improving accessibility for artists with limited technical expertise.

Another promising direction involves improving the robustness of the framework for complex contact-rich simulations. Cloth interactions involving dense self-collisions, multiple deformable objects, or rapidly changing contact conditions remain computationally challenging and would benefit from additional algorithmic improvements.

Future work could also investigate extending the framework to support topological changes such as cloth tearing, cutting, or stitching. Incorporating these operations into a differentiable optimization framework would significantly expand the range of simulations available to artists and improve the realism of interactive cloth design.

Finally, adaptive refinement represents a particularly promising research direction. Rather than uniformly refining the entire simulation mesh, adaptive methods could selectively increase resolution only within regions exhibiting large deformation or complex contact. Such an approach has the potential to further reduce computational cost while maintaining the visual fidelity of the final animation.

Chapter 7

Conclusion

This thesis presented a coarse-to-fine framework for artist-directed cloth simulation that combines differentiable inverse simulation with Progressive Dynamics. By formulating artistic intent as a differentiable optimization objective, the proposed method enables artists to directly specify desired cloth behavior while preserving the physically based nature of the underlying simulation.

To improve computational efficiency, inverse optimization is performed on a hierarchy of progressively refined simulation meshes. Rather than optimizing high-resolution cloth directly, the proposed framework first recovers large-scale motion on a coarse representation before transferring the optimized state and control parameters to increasingly finer resolutions through Progressive Dynamics. This strategy significantly reduces the computational cost associated with inverse simulation while maintaining visually plausible cloth behavior.

Experimental evaluation demonstrates that the proposed framework accelerates optimization compared to direct high-resolution inverse simulation while maintaining the quality of cloth animations across multiple benchmark scenarios. Furthermore, the framework maintains compatibility with existing physics-based simulation techniques, making it a practical addition to modern animation pipelines.

Overall, this work demonstrates that integrating Progressive Dynamics with differentiable inverse simulation provides an effective solution for efficient artist-directed cloth animation. The proposed framework represents a step toward more intuitive and computationally efficient physically based animation systems, opening new opportunities for future research in inverse simulation and interactive computer graphics.

Appendix A

Proof of Timestep Jacobian

Proof of Equation 3.26.

At the end of an implicit timestep, the stationarity condition for the step potential function yields:

$$M (x^{n+1} - (x^n + hv^n)) + \nabla E(x^{n+1}) = 0$$

Differentiating both sides with respect to x^n :

$$M \left(\frac{\partial x^{n+1}}{\partial x^n} - I \right) + \nabla^2 E(x^{n+1}) \frac{\partial x^{n+1}}{\partial x^n} = 0$$

Expanding and grouping terms:

$$M \frac{\partial x^{n+1}}{\partial x^n} - M + \nabla^2 E(x^{n+1}) \frac{\partial x^{n+1}}{\partial x^n} = 0$$

$$(M + \nabla^2 E(x^{n+1})) \frac{\partial x^{n+1}}{\partial x^n} = M$$

Multiplying from the left by $(M + \nabla^2 E(x^{n+1}))^{-1}$:

$$\frac{\partial x^{n+1}}{\partial x^n} = [M + \nabla^2 E(x^{n+1})]^{-1} M$$

For progressive projection, replace the energy $E(x)$ with the proxy energy $F(x)$.

Bibliography

- [1] David Baraff and Andrew Witkin. Large steps in cloth simulation. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '98*, page 43–54, New York, NY, USA, 1998. Association for Computing Machinery. ISBN 0897919998. doi: 10.1145/280814.280821. URL <https://doi.org/10.1145/280814.280821>. 2.1
- [2] Jan Bender, Daniel Weber, and Raphael Diziol. Fast and stable cloth simulation based on multi-resolution shape matching. *Computers Graphics*, 37(8):945–954, 2013. ISSN 0097-8493. doi: <https://doi.org/10.1016/j.cag.2013.08.003>. URL <https://www.sciencedirect.com/science/article/pii/S0097849313001283>. 2.4
- [3] Sofien Bouaziz, Sebastian Martin, Tiantian Liu, Ladislav Kavan, and Mark Pauly. Projective dynamics: fusing constraint projections for fast simulation. *ACM Trans. Graph.*, 33(4), July 2014. ISSN 0730-0301. doi: 10.1145/2601097.2601116. URL <https://doi.org/10.1145/2601097.2601116>. 2.1
- [4] Kwang-Jin Choi and Hyeong-Seok Ko. Stable but responsive cloth. *ACM Trans. Graph.*, 21(3):604–611, July 2002. ISSN 0730-0301. doi: 10.1145/566654.566624. URL <https://doi.org/10.1145/566654.566624>. 2.4
- [5] Tao Du, Kui Wu, Pingchuan Ma, Sebastien Wah, Andrew Spielberg, Daniela Rus, and Wojciech Matusik. Diffpd: Differentiable projective dynamics. *ACM Trans. Graph.*, 41(2), November 2021. ISSN 0730-0301. doi: 10.1145/3490168. URL <https://doi.org/10.1145/3490168>. 2.2
- [6] Minchen Li, Zachary Ferguson, Teseo Schneider, Timothy Langlois, Denis Zorin, Daniele Panozzo, Chenfanfu Jiang, and Danny M. Kaufman. Incremental potential contact: intersection-and inversion-free, large-deformation dynamics. *ACM Trans. Graph.*, 39(4), August 2020. ISSN 0730-0301. doi: 10.1145/3386569.3392425. URL <https://doi.org/10.1145/3386569.3392425>. 2.1
- [7] Minchen Li, Danny M. Kaufman, and Chenfanfu Jiang. Codimensional incremental potential contact. *ACM Trans. Graph. (SIGGRAPH)*, 40(4), 2021. 2.1
- [8] Siwang Li, Jin Huang, Fernando de Goes, Xiaogang Jin, Hujun Bao, and Mathieu Desbrun. Space-time editing of elastic motion through material optimization and reduction. *ACM Trans. Graph.*, 33(4), July 2014. ISSN 0730-0301. doi: 10.1145/2601097.2601217. URL <https://doi.org/10.1145/2601097.2601217>. 2.3
- [9] Yifei Li, Tao Du, Kui Wu, Jie Xu, and Wojciech Matusik. Diffcloth: Differentiable cloth

simulation with dry frictional contact. *ACM Trans. Graph.*, 42(1), October 2022. ISSN 0730-0301. doi: 10.1145/3527660. URL <https://doi.org/10.1145/3527660>. 2.2

- [10] Miles Macklin, Matthias Müller, and Nuttapong Chentanez. Xpbd: position-based simulation of compliant constrained dynamics. In *Proceedings of the 9th International Conference on Motion in Games*, MIG '16, page 49–54, New York, NY, USA, 2016. Association for Computing Machinery. ISBN 9781450345927. doi: 10.1145/2994258.2994272. URL <https://doi.org/10.1145/2994258.2994272>. 2.1
- [11] {Seung Woo} Oh, Junyong Noh, and Kwangyun Wohn. A physically faithful multigrid method for fast cloth simulation. *Computer Animation and Virtual Worlds*, 19(3-4):479–492, August 2008. ISSN 1546-4261. doi: 10.1002/cav.255. 2.4
- [12] Christian Schulz, Christoph von Tycowicz, Hans-Peter Seidel, and Klaus Hildebrandt. Animating deformable objects using sparse spacetime constraints. *ACM Trans. Graph.*, 33(4), July 2014. ISSN 0730-0301. doi: 10.1145/2601097.2601156. URL <https://doi.org/10.1145/2601097.2601156>. 2.3
- [13] Nur Saadah Mohd Shapri, Abdullah Bade, and Daut Daman. Hierarchy techniques in self-collision detection for cloth simulation. In *2009 Second International Conference on Machine Vision*, pages 325–329, 2009. doi: 10.1109/ICMV.2009.23. 2.4
- [14] Rasmus Tamstorf, Toby Jones, and Stephen F. McCormick. Smoothed aggregation multigrid for cloth simulation. *ACM Trans. Graph.*, 34(6), November 2015. ISSN 0730-0301. doi: 10.1145/2816795.2818081. URL <https://doi.org/10.1145/2816795.2818081>. 2.4
- [15] Jiayi Eris Zhang, Doug James, and Danny M. Kaufman. Progressive dynamics for cloth and shell animation. *ACM Trans. Graph.*, 43(4), July 2024. ISSN 0730-0301. doi: 10.1145/3658214. URL <https://doi.org/10.1145/3658214>. 2.4, 3.2.1
- [16] Jiayi Eris Zhang, Doug L. James, and Danny M. Kaufman. Progressive dynamics++: A framework for stable, continuous, and consistent animation across resolution and time. *ACM Transactions on Graphics (TOG)*, 44(4):1–20, 2025. 2.4, 3.2.1