Vision, Modeling, and Visualization (2025)
B. Egger and T. Günther (Eds.)

Differentiable XPBD for Gradient-Based Learning of Physical Parameters from Motion

S. Drysch and D. Stotko and R. Klein

University of Bonn, Germany

Abstract

Accurate cloth simulation is a vital component in computer graphics, virtual reality, and fashion design. Position-Based Dynamics (PBD) and its extension (XPBD) offer robust and efficient methods for simulating deformable objects like cloth. This paper details the evaluation and comparison of cloth simulations based on XPBD, including its "small steps" variant and an Energy-Aware (EA) modification. The XPBD variants are evaluated for their physical plausibility and energy conservation to analyze their suitability for inverse problems. Furthermore, we explore the implementation of a differentiable XPBD simulator, enabling the estimation of material properties and external forces. The differentiable simulator is assessed for its capability to estimate parameters in scenarios of increasing complexity. Results indicate that small time steps with single iterations in XPBD offer good energy behavior, while the EA modification exhibits undesired characteristics. The differentiable simulator successfully estimates single parameters but identifies challenges with multi-parameter optimization due to compensatory effects.

CCS Concepts

• ***Computing Methodologies*** → *Modeling and simulation;* • ***Applied computing*** → *Physical sciences and engineering;*

1. Introduction

The simulation of clothing, textiles, and fabrics is integral to various fields, from fashion manufacturing and presentation to the video game and movie industry as well as virtual reality applications. Over the years, research has enabled applications like virtual try-ons [GRH*12,SOC19] and clothing design [Wan18,MTMP20]. A central challenge remains: generating cloth simulations that are not only visually realistic but also performant, robust, and accurate in predicting arbitrary deformations. The only way to obtain consistent dynamics of a real cloth in a virtual world is a simulation-based prediction using the correct physical material parameters.

Among the many simulation techniques, Position-Based Dynamics (PBD) and its extended formulation XPBD has become a popular choice due to its simplicity and real-time performance. XPBD improves upon PBD by introducing constraint compliance and better time integration, thereby enabling more physically plausible behavior, especially for stiff materials. Despite these advances, leveraging such simulation frameworks in inverse problems, where physical parameters are inferred from observed motion, remains non-trivial. Differentiable simulations allow for the computation of gradients of output variables with respect to input parameters, opening avenues for various optimization tasks. While analytical formulation of differentiable models is complex and error-prone, automatic differentiation frameworks offer an alternative. Besides general-purpose autodifferentiation frameworks like PyTorch [PGM*19], specialized frameworks, such as Warp [Mac22] and DiffTaichi [HAL*19], are designed to provide high-performance computations with automatic differentiation specialized for physical simulations. This enables direct optimization of material properties and external forces from motion data without manual computation and implementation of gradient functions.

This paper details a framework developed for cloth simulation based on these principles. The contributions are:

- We systematically analyze the physical accuracy and numerical behavior of multiple XPBD variants including standard XPBD [MMC16], a small-step approach [MSL*19], and an energy-aware (EA) modification [Cet19]. All variants are evaluated with respect to energy conservation and physical plausibility.
- We demonstrate the ability of our differentiable simulator to reconstruct physical parameters such as material stiffness, mass density, external forces, and collision parameters using gradient descent, showing both strengths and limitations.
- We identify and analyze the difficulties of multi-parameter optimization, particularly parameter compensation effects and ambiguity in inverse problems. These can hinder convergence to ground truth values despite low errors in the objective function.

2. Related Work

2.1. Cloth Simulation

Simulating deformable objects efficiently and stably has been a longstanding challenge in computer graphics. The foundational

© 2025 The Author(s).
Proceedings published by Eurographics - The European Association for Computer Graphics.
This is an open access article under the terms of the Creative Commons Attribution License, which permits use, distribution and reproduction in any medium, provided the original work is properly cited.

work of Baraff and Witkin [BW98] used mass-spring and implicit integration methods, which offer accurate physics at the cost of moderate speed. These techniques can be extended to reach even yarn-level accuracy [CLMMO14, CLMO15]. With the rapid advancement of neural networks and their integration into physical modeling, data-driven simulation has emerged as a compelling alternative to classical numerical methods [RPK19, CRBD18]. As with traditional simulation frameworks, these approaches cover general soft deformable bodies [PFSGB20] or cloth dynamics [SOC22, GBH23, SLD23] specifically among many other fields.

2.2. Position-Based Simulation

Position-Based Dynamics (PBD), introduced by Müller et al. [MHHR07], provides an iterative method for real-time simulation of deformable bodies, widely used in games and interactive applications due to its simplicity and unconditionally stable solver. Unlike many methods that adjust velocity, PBD directly modifies positions and then corrects velocities. A known limitation of PBD is that object stiffness is directly tied to the iteration count and time step size. The eXtended Position-Based Dynamics (XPBD) method [MMC16] addresses this by introducing compliance parameters, allowing stiffness to be controlled independently. Further work by Macklin et al. [MSL*19] demonstrated that using many small time steps with fewer solver iterations per step can yield more stable results than large time steps with many solver iterations. Cetinaslan [Cet19] proposed an Energy-Aware (EA) modification to XPBD that introduces feedback from kinetic and potential energy discrepancies to improve the stiffness behavior.

2.3. Differentiable Simulation and Parameters Estimation

Several recent works have explored learning-based estimation of physical parameters from diverse input. Yang et al. [YLL17] recover cloth material properties from video using a data-driven approach, while Feng et al. [FHXW22] estimate bending stiffness with a supervised method using a drape tester. Gong et al. [GZBW22] introduce a fine-grained differentiable yarn-level fabric model, offering high accuracy at increased computational cost. In the domain of shape reconstruction, physics-based models are incorporated in shape-from-template pipelines [KTE*22, SWK24], enhancing realism and stability. While Stuyck and Chen [SC23] already provided an analytical formulation of a differentiable XPBD solver, our work provides three distinct contributions compared to their and other work: (1) we perform the first systematic evaluation of multiple XPBD variants (including small-steps and the EA method) to assess their numerical accuracy and suitability specifically for inverse problems; (2) we provide an analysis of the optimization landscape, identifying practical strategies like log-scaling for stiffness and (3) we explicitly identify and analyze the parameter ambiguities and compensatory effects that arise in multi-parameter estimation, a critical challenge not detailed in prior work.

3. Methodology

This section presents the core simulation strategy based on eXtended Position-Based Dynamics (XPBD), along with two key modifications: the small-step XPBD approach and the energy-aware (EA) variant [MMC16, MSL*19, Cet19]. In this paper, we focus on soft body and especially cloth simulation.

3.1. eXtended Position-Based Dynamics (XPBD)

XPBD was introduced to address the limitations of conventional position-based methods, particularly the dependence of material stiffness on time step size and the number of solver iterations. We shortly repeat the derivation of the XPBD update formulas and highlight important parts for our findings.

We start with Newton's equation of motion

$$M\ddot{\mathbf{x}}(t) = \mathbf{F}(t, \mathbf{x}(t)) = -\nabla U(\mathbf{x}) \tag{1}$$

in which we combine all particle quantities into single vectors for their position $\mathbf{x}(t) = (x_1(t), \dots, x_n(t))^T$ and the forces $\mathbf{F} = (F_1, \dots, F_n)^T$ acting on them. The forces here are assumed to be conservative such that we can replace them with the gradient of an energy potential $U(\mathbf{x})$. Contributions of non-conservative external forces will be added later in a prediction step. All particle masses are combined in a diagonal mass matrix M which is either set directly or computed from a mass density ρ.

In XPBD, the energy potential

$$U(\mathbf{x}) = \frac{1}{2}\mathbf{C}(\mathbf{x})^T \alpha^{-1} \mathbf{C}(\mathbf{x}) = \frac{1}{2}\sum_{j=1}^{m} \alpha_j^{-1} (C_j(\mathbf{x}))^2 \tag{2}$$

is modeled using constraint functions $\mathbf{C}(\mathbf{x}) = (C_1(\mathbf{x}), \dots, C_m(\mathbf{x}))^T$ and a diagonal compliance matrix α. Note that the constraint functions themselves are therefore closely related to the energies in the simulated system. By discretizing Equation (1) in time as

$$M\left(\frac{\mathbf{x}(t+\Delta t) - 2\mathbf{x}(t) + \mathbf{x}(t-\Delta t)}{(\Delta t)^2}\right) = -\nabla U(\mathbf{x}(t+\Delta t)) \tag{3}$$

and introducing a Lagrange multiplier $\lambda = -\frac{1}{(\Delta t)^2}\alpha^{-1}\mathbf{C}(\mathbf{x})$, we arrive at the non-linear system of equations

$$\begin{aligned} M\left(\mathbf{x}(t+\Delta t) - \tilde{\mathbf{x}}(t)\right) - \nabla \mathbf{C}(\mathbf{x}(t+\Delta t))^T \lambda(t+\Delta t) = 0 \\ \mathbf{C}(\mathbf{x}(t+\Delta t)) + (\Delta t)^2 \alpha \lambda(t+\Delta t) = 0 \end{aligned} \tag{4}$$

where $\tilde{\mathbf{x}}(t) = 2\mathbf{x}(t) - \mathbf{x}(t-\Delta t) = \mathbf{x}(t) + \mathbf{v}(t)\Delta t$ is the predicted position. We now solve Equation (4) by linearizing it and using a few approximations that are detailed in [MMC16] to arrive at

$$\left(\nabla \mathbf{C}(\mathbf{x}(t)) M^{-1} \nabla \mathbf{C}(\mathbf{x}(t))^T\right) \Delta\lambda = -\mathbf{C}(\mathbf{x}(t)) - \frac{1}{(\Delta t)^2}\alpha\lambda(t) \tag{5}$$

and update the value $\lambda \leftarrow \lambda + \Delta\lambda$. The position update is computed similarly with the displacement

$$\Delta\mathbf{x} = M^{-1} \nabla \mathbf{C}(\mathbf{x}(t))^T \Delta\lambda \tag{6}$$

We solve Equation (5) via a Jacobi iteration such that we solve several constraints $C_j(\mathbf{x}(t))$ independently in parallel. This is faster than other methods like a Gauss-Seidel iteration and independent of any ordering within a solver iteration. Thus, Equation (5) reduces to the scalar equation

$$\Delta\lambda_j = \frac{-C_j(\mathbf{x}(t)) - \frac{1}{(\Delta t)^2}\alpha_j \lambda_j(t))}{\nabla C_j(\mathbf{x}(t)) M^{-1} \nabla C_j^T(\mathbf{x}(t)) + \frac{1}{(\Delta t)^2}\alpha_j} \tag{7}$$

© 2025 The Author(s).
Proceedings published by Eurographics - The European Association for Computer Graphics.

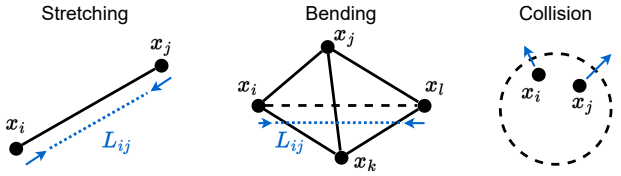

Figure 1: *Schematic visualization of all three constraint types.*

This formulation not only ensures stability at large time steps but also allows to model the stiffness of these constraints using the compliance parameters α_j.

The entire XPBD loop follows a predictor-corrector scheme. For each simulation step, particle positions are first predicted based on current velocities and accelerations from external forces:

$$\tilde{\mathbf{x}}(t+\Delta t) = \mathbf{x}(t) + \Delta t \mathbf{v}(t) + (\Delta t)^2 \mathbf{a}_{\text{ext}}(t) \tag{8}$$

Subsequently, the iterative solution of Equations (6) and (7) refines these predicted positions $\mathbf{x}(t+\Delta t) = \tilde{\mathbf{x}}(t+\Delta t) + \Delta \mathbf{x}$ by enforcing constraints that are related to physical energy terms such as stretching and bending energies. Finally, particle velocities are updated from the corrected positions

$$\mathbf{v}(t+\Delta t) = \frac{1}{\Delta t}\left(\mathbf{x}(t+\Delta t) - \mathbf{x}(t)\right) \tag{9}$$

such that all quantities are related to positions only.

3.2. Constraints

To ensure the simulated system remains concise and tractable, we intentionally restrict our set of constraints to stretching (distance), bending (angle) and collisions with spheres. This will simplify the interpretation and analysis of the results we find while still allowing for diverse and rich dynamics. All of these constraints are visualized in Figure 1.

3.2.1. Stretching

Distance constraints between two particles are often defined using a standard Hookean energy potential

$$E_Y = \frac{1}{2} Y \left(|x_i - x_j| - L_{ij}\right)^2 \tag{10}$$

where Y is the stretching stiffness parameter or Young's modulus and L_{ij} is the rest length or target distance of this edge. Comparing this expression with Equation (2) immediately reveals the constraint function to be defined as

$$C_{\text{stretch}}(x_i, x_j) = |x_i - x_j| - L_{ij} \tag{11}$$

with a compliance value of $\alpha_Y = \frac{1}{Y}$. We will keep the same compliance value for all stretching constraints to obtain a consistent behavior over the entire object and a manageable parameter optimization task. Nonetheless, individual values per constraint are a trivial generalization to these and following constraints.

3.2.2. Bending

We realize angle constraints between faces in a triangle mesh in the following way: each pair of triangles is connected via a shared edge between two vertices x_j, x_k. The relative bending of two faces along this edge changes the distance between the remaining two vertices x_i, x_l and therefore motivates a similar constraint definition as for stretching. In particular, we make use of the exact same constraint function as before but employ a different bending stiffness B such that the potential energy for a single bending constraint reads

$$E_B = \frac{1}{2} B \left(|x_i - x_l| - L_{il}\right)^2 \tag{12}$$

There are many other bending models for soft bodies like cloth but we decided to use this one due to its simplicity regarding the comparison of different XPBD variations and the gradient-based optimization of physical parameters.

3.2.3. Collision with Spheres

The collision of a vertex x_i with a static sphere is again modeled by the distance between both objects. For a sphere with radius r at position p we can use

$$C(x_i) = \begin{cases} |x_i - p| - r & \text{if } |x_i - p| < r \\ 0 & \text{else} \end{cases} \tag{13}$$

to define a constraint function. When modeling a perfectly rigid collision, the corresponding compliance value α has to be equal to zero. The resulting update formula then simply describes a projection of the vertex position x_i onto the sphere surface. We therefore include the collision directly in the prediction step by performing this projection P on the predicted positions:

$$P\left(\tilde{x}_i'(t+\Delta t)\right) = P\left((1-\mu)\cdot \tilde{x}_i(t+\Delta t) + \mu \cdot x_i(t)\right)$$

We additionally include a simple friction effect by interpolating the current positions $x_i(t)$ and the newly predicted position $\tilde{x}_i(t+\Delta t)$ with a friction parameter $0 < \mu \leq 1$. In all of our experiments we keep the friction parameter constant at $\mu = 0.5$. This friction effect and the projection are obviously applied only if the originally predicted position $\tilde{x}_i(t+\Delta t)$ was inside the sphere.

3.3. Small-Step XPBD

An important modification to the XPBD algorithm is the replacement of many solver iterations for Equation (7) at large time steps Δt with just one iteration at smaller steps Δt [MSL*19]. This reduces the equation to

$$\Delta\lambda_j = \frac{-C_j(\mathbf{x}(t))}{\nabla C_j(\mathbf{x}(t)) M^{-1} \nabla C_j^T(\mathbf{x}(t)) + \frac{1}{(\Delta t)^2}\alpha_j} \tag{14}$$

The authors point out that the main error comes from the external forces or accelerations in Equation (8) that scale with $(\Delta t)^2$. We will treat XPBD with and without this modification as two separate variants and compare them with each other.

3.4. Energy-aware (EA) XPBD

While XPBD improves stability and stiffness control, it can suffer from numerical energy dissipation over time. To address this, an

© 2025 The Author(s).
Proceedings published by Eurographics - The European Association for Computer Graphics.

energy-aware (EA) modification, proposed by Cetinaslan [Cet19], was implemented. This variant monitors kinetic and potential energy differences before and after applying constraints in each solver iteration. Here, the kinetic and potential energy differences are calculated as

$$\Delta\text{KE} = \sum_{i=1}^{N} \frac{1}{2} m_i \left((v_i(t+\Delta t))^2 - (v_i(t))^2 \right) \tag{15}$$

$$\Delta\text{PE} = \sum_{j=1}^{M} C_j(\mathbf{x}(t+\Delta t)) - C_j(\mathbf{x}(t)) \tag{16}$$

Note that the proposed potential energy only takes constraint functions into account without other sources like a gravitational field. Any energy imbalance quantified as an energy residual $\epsilon = \Delta\text{KE} + \Delta\text{PE}$ is added to the constraint function such that the XPBD algorithm is employed with

$$\mathbf{C}'(\mathbf{x}) = \mathbf{C}(\mathbf{x}) + \epsilon \tag{17}$$

This feedback mechanism reinforces or relaxes constraints to maintain energy balance.

Moreover, the author uses spring potentials directly as constraint functions

$$C_{\text{Hookean}}(x_i, x_j) = \frac{1}{2} Y' \left(|x_i - x_j| - L_{ij} \right)^2 \tag{18}$$

with Y' being the stretching stiffness. Note, that a stiffness or compliance value is usually not part of the constraint function and we will investigate its impact in Section 4.2.2. Thus, the updated constraint function $\mathbf{C}'(\mathbf{x})$ is a sum of two energy terms with compatible physical units.

3.5. Differentiable Simulation

Our main goal is the reconstruction of physical parameters like material parameters (ρ, Y, B), external accelerations $\mathbf{a}_{\text{ext}}$, and position and radius of a colliding sphere. For that, we follow standard gradient-based optimization schemes employing auto-differentiation frameworks. The loss function will be the mean-square error (MSE) that compares the position of all simulated vertices $\mathbf{x}(t_{\text{final}})$ at a specific point in time t_{final} with reference vertex positions $\mathbf{y}$:

$$\mathcal{L} = \frac{1}{N} \sum_{i=1}^{N} \| x_i(t_{\text{final}}) - y_i \|^2$$

The reference positions will be also be computed via our simulation procedure such that we obtain an idealized and controlled setup for the reconstruction of physical parameters with known ground truth values. Real-world tasks usually involve further difficulties such as noisy data from depth sensors or implicit information from using RGB images only.

4. Evaluation

4.1. Implementation

All XPBD variants are implemented using DiffTaichi [HLA*19, HAL*19] as it is designed and optimized to serve the needs of differentiable physics simulations. The results of the gradient computation are converted to PyTorch [PGM*19] to make use of the Adam Optimizer [KB14]. Although it is an adaptive optimizer, it is not directly suitable for all of our parameters. Meaningful parameters for stretching and bending stiffness Y, B and corresponding compliance values α_Y, α_B range over several orders of magnitude and therefore often experience either unstable optimization for too large learning rates or slow convergence for too low learning rates. However, suitable learning rates heavily depend on the initial and target values with the latter being unknown in actual reconstruction tasks. Hence, we optimize the logarithms $\log_{10}(\alpha_Y)$ and $\log_{10}(\alpha_B)$ using a constant learning rate. The benefit of using log-scaled stiffness parameters is evaluated in Section 4.4.1. All learning rates are summarized in Table 1.

Table 1: *Learning rates for all physical parameters.*

Parameter	ρ	$\log_{10}(\alpha_Y)$	$\log_{10}(\alpha_B)$	$\mathbf{F}_{\text{ext}}$	p_{shpere}	r_{shpere}
Learning rate	0.01	0.4	0.2	0.05	0.03	0.03

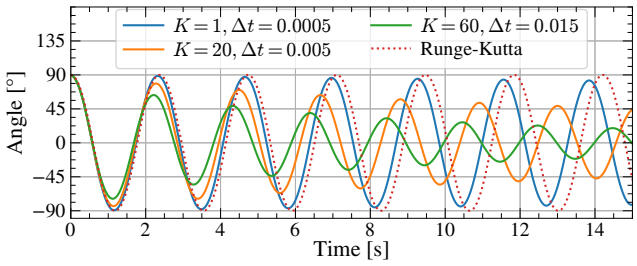

Figure 2: *Simulation of a pendulum with different time steps Δt.*

4.2. Comparison of XPBD Variants

We begin by comparing XPBD variants to determine the most appropriate for optimization. Each variant is analyzed with respect to its computational behavior and physical accuracy in simulating different mass-spring systems.

4.2.1. Pendulum

First, we want to reproduce the findings of Macklin et. al. [MSL*19] when taking smaller time steps with fewer iterations. To do so, we simulate a regular pendulum without damping released at a 90° angle. The pendulum has a length of 1 m and is pulled down by gravity with a gravitational acceleration of $g = 9.81 \frac{\text{m}}{\text{s}^2}$. As we want to simulate a perfectly stiff pendulum string, we set the compliance value for the stretching constraint to $\alpha = 0$ corresponding to a stiffness parameter of $Y = \infty$. We compare different numbers of solver iterations K as well as time steps Δt and show the resulting oscillations in Figure 2. A reference simulation is computed by solving the differential equation of that pendulum with a high-quality Runge-Kutta integration.

We indeed find improved simulation results with reduced time step size Δt. The improvements not only affect the amplitude, which indicates better energy conservation, but also the oscillation frequency. Similar results are also observable in other experiments but we will directly focus on the small-step XPBD and compare it against the energy-aware XPBD algorithm.

© 2025 The Author(s).
Proceedings published by Eurographics - The European Association for Computer Graphics.

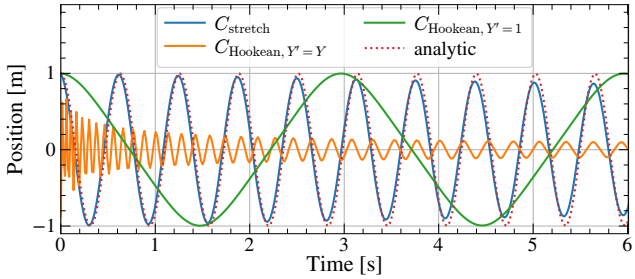

Figure 3: *Simulation of a harmonic oscillator using different constraint functions C.*

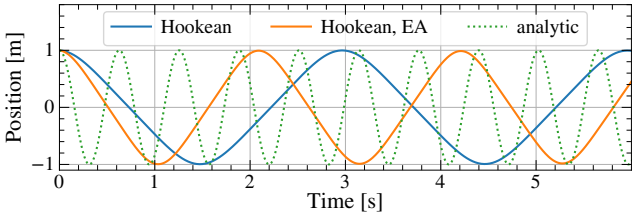

Figure 4: *Effect of energy-awareness on the harmonic oscillator.*

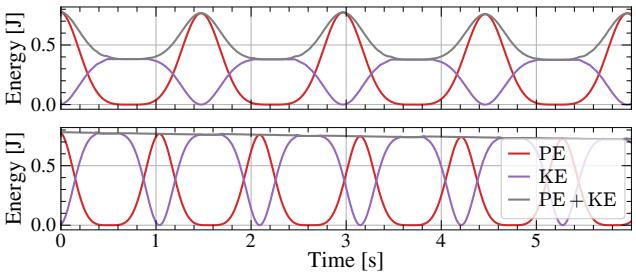

Figure 5: *Energy over time without (top) and with (bottom) energy-awareness. The potential energy is computed via Equation (2).*

4.2.2. Harmonic Oscillator

A 1D harmonic oscillator is the simplest system of just one particle being pulled towards its rest position at $x = 0$ without friction. On one hand, this simplicity allows us to compare the simulation results to the analytical solution

$$y(t) = y_0 \cos(\omega t)$$

with an initial displacement y_0 and without any initial velocity $v_0 = 0$. On the other hand, the forces acting on the particle follow Hooke's law and can be used to model a wide variety of mechanical interactions. Thus, the results of this analysis will give important insights for objects that can be described via a mass-spring model.

We first compare the behavior of the Hookean constraint with the regular stretching constraint and the analytic solution as a reference in Figure 3. We use the small time step $\Delta t = 0.0005\,\text{s}$ and $K = 1$ solver iteration which are similar to the best setup in the previous experiment. The Hookean constraint with stretching stiffness $Y' = Y = 100 \Rightarrow \alpha_Y = \frac{1}{100}$ produces unstable and very high-frequent oscillations in the beginning that are not close to the expected analytical solution. We therefore remove the parameter from the Hookean constraint function by setting $Y' = 1, Y = 100 \Rightarrow \alpha_Y = \frac{1}{100}$. This time, the simulation is stable but yields a low-frequent oscillation that also does not match the reference.

Now, we add the energy-awareness (EA) to the Hookean constraint with $Y' = 1$ and get the results in Figure 4. While the frequency is slightly altered, it remains inconsistent with the analytical solution. Moreover, the Hookean constraint (both with and without EA) modifies the dynamics in a way that breaks the expected sinusoidal motion. This deviation is more apparent in the energy diagram in Figure 5, where the phases of high potential energy are noticeably shorter than those of high kinetic energy.

Despite that, without the energy-awareness, the Hookean constraint produces varying total energies. All dissimilarities follow from the constraint itself being inconsistent with the XPBD theory. When using the Hookean constraint the theory assumes an energy potential of

$$U(x) = \frac{1}{2}\alpha^{-1}\left(C_{\text{Hookean}}(x,0)\right)^2 = \frac{1}{8}\alpha^{-1}Y'^2x^4$$

which creates different dynamics than a harmonic oscillator. Including EA in the constraint does produce nearly constant total energies but still suffers from the inconsistent constraint definition. However, adding EA to the regular stretching constraint C_{stretch} will not yield an ideal solution either. As mentioned in Section 3.4, the energy imbalance ϵ does not take external energy sources like a gravitational potential into account. This is not a problem for the harmonic oscillator but affects most other real-world simulations. Lastly, as the constraint functions only need to be fulfilled "softly", quantified by the compliance α, the energy conservation will also not strictly be enforced when $\alpha > 0$.

4.3. Cloth

We now focus on the simulation of a square piece of cloth that is modeled as a mass-spring system and held only at its two upper corners. The mass per vertex is computed by adding up a third of the weight of each adjacent triangle which is determined by an area mass density ρ. Besides one type of distance constraint (either stretching or Hookean), this simulation also includes the bending constraint C_B as well as gravitational forces pulling the cloth downwards. Note that the bending compliance α_B usually is several orders of magnitude weaker than the stretching compliance α_Y and therefore not able to support the weight of the cloth.

A comparison of all simulation variants is given in Figure 6. All simulations use the same compliance values $\alpha_Y = 10^{-4}, \alpha_B = 0.1$ and the Hookean constraints additionally use $Y' = 1$. The parameters are chosen such that the cloth is supposed to undergo only minimal deformations. Similar to previous analyses, we observe the small-steps variant (left) to have the best simulation results. Using more solver iterations at the cost of larger time steps while keeping the runtime constant (middle left) results in a visibly softer appearance. The Hookean constraint is not able to sufficiently support the cloth under its own weight independent of employing the EA update (middle right) or not (right).

© 2025 The Author(s).
Proceedings published by Eurographics - The European Association for Computer Graphics.



Figure 6: *Cloth simulation with different stretching constraints. C_{stretch} with small (left) and large (middle-left) Δt, C_{Hookean} at small Δt with EA (middle right) and without EA (right).*

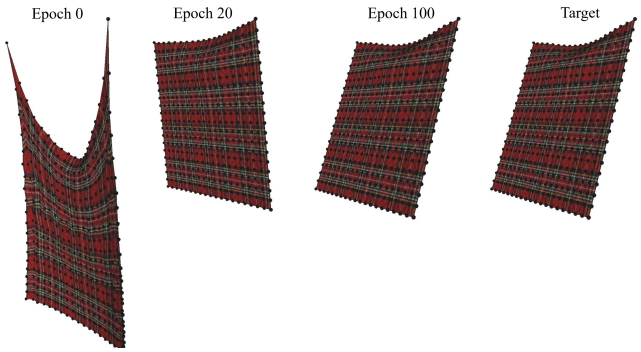

Figure 7: *The parameters of a hanging cloth are fitted by employing gradient-based optimization. The loss function is defined with the positions which are marked with black points.*

4.4. Evaluation of Physical Parameter Optimization

The analysis in Section 4.2 indicates that the small-steps variant [MSL*19] is superior regarding accuracy and consistency. Thus, we perform the physical parameter optimization using this simulation technique. The scenario will always be similar to Figure 7 in which the cloth starts with some predefined location and orientation. We always simulate 2000 time steps with $\Delta t = 0.001\,\text{s}$ and compare the vertex positions after these 2 seconds with the target positions. Note that the target positions are also computed by the simulation using specified physical parameters. Our goal is to recover the physical parameters as precisely as possible by only observing the 3D vertex positions.

4.4.1. Few parameters

Only Stretching:

First, we start with optimizing single parameters. All other physical parameters are constant during the optimization process and set to their ground truth values. We compare the optimization with and without logarithmic scaling on two examples, both involving the optimization of the stretching compliance $\alpha_Y = \frac{1}{Y}$. First, the parameter is initialized at 0.1 with a target of 10^{-4} ($0.1 \rightarrow 10^{-4}$), and in the second case optimized with $10 \rightarrow 0.1$ (see Figure 8). We clearly observe that the log-scaled parameter converges quickly and independent of the target value. Meaningful values of the scaled parameter only range from about -7 to 2 instead of 10^{-7} to 10^{2} for the unscaled parameter. Thus, a single learning rate is sufficient to

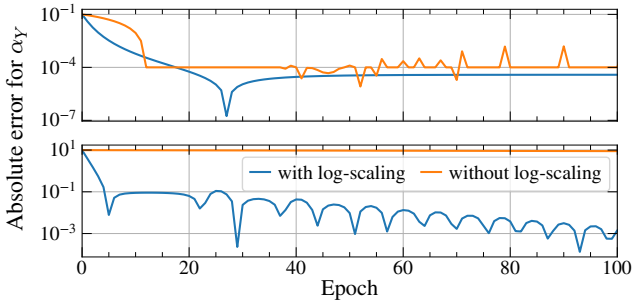

Figure 8: *Optimization of the stretching compliance $\alpha_Y = \frac{1}{Y}$ with and without logarithmic scaling using learning rates 0.4 and 10^{-4} respectively. The values are optimized from 0.1 to 10^{-4} (top) and from 10 to 0.1 (bottom). With logarithmic scaling the optimization converges quickly independent of the target value.*

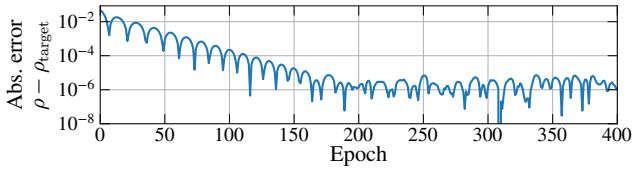

Figure 9: *Optimization of the area density ρ ($0.3 \rightarrow 0.25$).*

get quick convergence. Moreover, in both cases the optimization is able to reconstruct the correct value after 100 epochs up to absolute errors of $3.5 \cdot 10^{-5}$ and 10^{-3}. At $\alpha_y = 10^{-4}$ there are only negligible differences in the simulation result and the compliance may be influenced by the time step size Δt. Due to the finite time steps it is not possible to simulate arbitrarily stiff material with this approach such that the nuances of the motion can not be captured. The results for $\alpha_y = 0.1$ is much more accurate as the time steps size is sufficient to predict the dynamics.

Only Density:

Next, we optimize the area density ρ while keeping the other parameters fixed. Parameters other than the compliances α_Y and α_B are not scaled logarithmically because their values do not range over several orders of magnitude. In Figure 9 we see that we need about 170 epochs to converge but the target value is reach with very high accuracy of 10^{-6} close to floating point precision.

Stretching and Bending:

When optimizing streching and bending compliance α_Y, α_B simultaneously, we observe very intricate behavior. The diagram in Figure 10 shows that both parameters improve for the first 230 epochs and approach the target values but do not converge properly even after 400 epochs. The similarities between stretching and bending constraints introduce ambiguities that can not be resolved from the information given in the task. An important quantity therefore is the Loss function measuring the MSE. Despite both physical parameters changing constantly, the loss function reaches low values between 10^{-4} and 10^{-7}. Hence, the measured vertices exhibit only minimal displacement, strongly suggesting the presence of an ambiguity.

© 2025 The Author(s).
Proceedings published by Eurographics - The European Association for Computer Graphics.

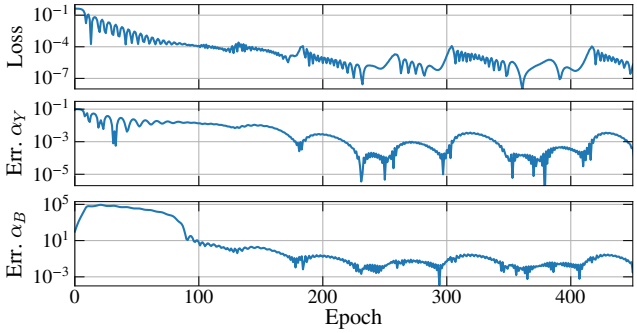

Figure 10: *Simultaneous optimization of stretching compliance* α_Y *(*$10^{-4} \rightarrow 0.1$*) and bending compliance* $\alpha_B = \frac{1}{B}$ *(*$100 \rightarrow 1$*).*

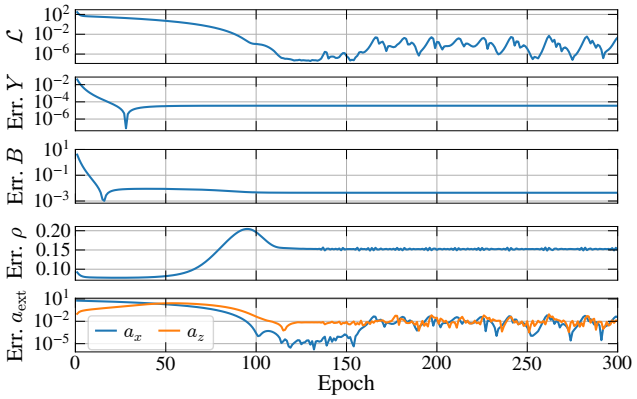

Figure 11: *Optimization of multiple parameters:* $\alpha_Y(0.1 \rightarrow 10^{-4})$, $\alpha_B(10 \rightarrow 0.01)$, $\rho(0.3 \rightarrow 0.2)$, $a_{\text{ext}}((3,0,0) \rightarrow (-3,0,0))$.



Figure 12: *Optimization scenario inducing collision with a ball. Here, we optimize for the stretching stiffness* Y*, the ball's position* p *and radius* r*.*

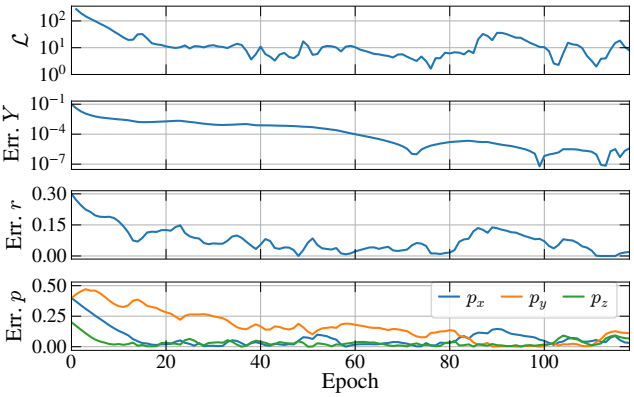

Figure 13: *Optimization of multiple parameters:* $\alpha_Y(0.1 \rightarrow 10^{-4})$, $r(0.2 \rightarrow 0.5)$, $p(0.2,-0.7,-0.1) \rightarrow (-0.2,-1.1,0.1))$.

4.4.2. Multiple parameters

Without Collision:

We now perform optimization tasks in several different variables. First, we still look at the hanging cloth as before but optimize stretching and bending compliances α_Y, α_B, area density ρ and external wind $\mathbf{a}_{\text{ext}}$ simultaneously. We assume the external accelerations to be constant in time and equal for all vertices. Without this assumption, the accelerations would introduce thousands of degrees of freedom. Moreover, we only change the horizontal components a_x and a_z as the vertical component a_y (not including gravity) is negligible in most cases.

The visual results of this experiment are shown in Figure 7 while the evolution of all physical parameters is depicted in Figure 11. We again observe complex behavior: while stretching and bending compliance converge this time to values comparable to their targets, the density reaches a completely different value than its target. On the other hand, the external acceleration vector is recovered with good precision. That indicates that some parameters influence each other while others do not. This also make sense intuitively: a heavier cloth needs to be stiffer to counteract the gravitational pull affecting density and compliance values. Our definition via external accelerations is independent of the mass while the use of forces is not. The visual results in Figure 7 and the loss in Figure 11 show again that the resulting vertex positions match the target position despite some physical parameters do not match their target values.

With Collision:

Lastly, we optimize a scenario with stretching compliance α_Y, radius r and position p of a colliding sphere. The visual results are presented in Figure 12 with the evolution of the parameters depicted in Figure 13. In this case, the stretching compliance α_Y is fitted well to the target value because no interfering parameters (bending compliance or density) are changed at the same time. Slight variations may appear since the values for the sphere's radius and position do not converge well. They approach the correct values at first but keep varying throughout the optimization process although they seem directly observable and independent of each other.

4.5. Limitations

While our differentiable XPBD framework enables effective gradient-based parameter estimation, several limitations remain. First, the optimization struggles with multiple interdependent parameters due to compensatory effects, leading to ambiguities even when the simulated motion closely matches the target (see Figures 7 and 11). Second, our evaluation relies on synthetic 3D data

© 2025 The Author(s).
Proceedings published by Eurographics - The European Association for Computer Graphics.

generated by the same simulator, limiting immediate applicability to real-world scenarios with noisy or indirect observations from cameras. Finally, although XPBD supports efficient simulation, the need for small time steps to ensure physical accuracy may increase computational cost in large-scale scenes.

5. Conclusion and Future Work

We presented a differentiable XPBD framework for learning physical parameters from motion through gradient-based optimization. By evaluating different XPBD variants, we identified the small-step formulation as the most physically consistent and stable for inverse tasks. Our experiments demonstrate that while single-parameter estimation performs reliably, multi-parameter scenarios reveal challenges due to compensatory effects and ambiguities even under idealized experimental conditions.

We aim to extend our approach by employing alternative bending constraint functions, as well as adding constraints for collisions and material effects such as shear resistance. This includes evaluating the correlation of all constraint functions regarding physical parameter optimization tasks. Moreover, we want to incorporate sensor noise from real-world data and deal with indirect observations. Optimizing over the entire motion sequence, rather than only the final positions, may offer additional information to help resolve existing ambiguities and provide consistency when using imperfect data.

Acknowledgements

This work has been funded by the Federal Ministry of Education and Research of Germany and the state of North-Rhine Westphalia as part of the Lamarr-Institute for Machine Learning and Artificial Intelligence and the InVirtuo 4.0 project.

References

[BW98] BARAFF D., WITKIN A.: Large steps in cloth simulation. In *Annual Conference on Computer Graphics and Interactive Techniques* (1998), SIGGRAPH. 2

[Cet19] CETINASLAN O.: Position-based simulation of elastic models on the gpu with energy aware gauss-seidel algorithm. *Computer Graphics Forum* (2019). 1, 2, 4

[CLMMO14] CIRIO G., LOPEZ-MORENO J., MIRAUT D., OTADUY M. A.: Yarn-level simulation of woven cloth. *ACM Trans. Graph.* (2014). 2

[CLMO15] CIRIO G., LOPEZ-MORENO J., OTADUY M. A.: Efficient simulation of knitted cloth using persistent contacts. In *Proceedings of the 14th ACM SIGGRAPH / Eurographics Symposium on Computer Animation* (2015), Association for Computing Machinery. 2

[CRBD18] CHEN R. T., RUBANOVA Y., BETTENCOURT J., DUVENAUD D. K.: Neural ordinary differential equations. *Advances in neural information processing systems* (2018). 2

[FHXW22] FENG X., HUANG W., XU W., WANG H.: Learning-based bending stiffness parameter estimation by a drape tester. *ACM Trans. Graph.* (2022). 2

[GBH23] GRIGOREV A., BLACK M. J., HILLIGES O.: Hood: Hierarchical graphs for generalized modelling of clothing dynamics. In *CVPR* (2023). 2

[GRH*12] GUAN P., REISS L., HIRSHBERG D. A., WEISS A., BLACK M. J.: Drape: Dressing any person. *ACM Trans. Graph.* (2012). 1

[GZBW22] GONG D., ZHU Z., BULPITT A., WANG H.: Fine-grained differentiable physics: a yarn-level model for fabrics. In *ICLR* (2022). 2

[HAL*19] HU Y., ANDERSON L., LI T., SUN Q., CARR N., RAGAN-KELLEY J., DURAND F.: Difftaichi: Differentiable programming for physical simulation. *CoRR* (2019). 1, 4

[HLA*19] HU Y., LI T.-M., ANDERSON L., RAGAN-KELLEY J., DURAND F.: Taichi: A language for high-performance computation on spatially sparse data structures. *ACM Trans. Graph.* (2019). 4

[KB14] KINGMA D. P., BA J.: Adam: A method for stochastic optimization. In *ICLR* (2014). 4

[KTE*22] KAIRANDA N., TRETSCHK E., ELGHARIB M., THEOBALT C., GOLYANIK V.: ϕ-SfT: Shape-from-Template with a Physics-Based Deformation Model. In *CVPR* (2022). 2

[Mac22] MACKLIN M.: Warp: A high-performance python framework for gpu simulation and graphics. https://github.com/nvidia/warp, 2022. NVIDIA GPU Technology Conference (GTC). 1

[MHHR07] MÜLLER M., HEIDELBERGER B., HENNIX M., RATCLIFF J.: Position based dynamics. *Journal of Visual Communication and Image Representation* (2007). 2

[MMC16] MACKLIN M., MÜLLER M., CHENTANEZ N.: Xpbd: position-based simulation of compliant constrained dynamics. In *Proceedings of the 9th International Conference on Motion in Games* (2016), Association for Computing Machinery. 1, 2

[MSL*19] MACKLIN M., STOREY K., LU M., TERDIMAN P., CHENTANEZ N., JESCHKE S., MÜLLER M.: Small steps in physics simulation. In *Proceedings of the 18th Annual ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2019), Association for Computing Machinery. 1, 2, 3, 4, 6

[MTMP20] MONTES J., THOMASZEWSKI B., MUDUR S., POPA T.: Computational design of skintight clothing. *ACM Trans. Graph.* (2020). 1

[PFSGB20] PFAFF T., FORTUNATO M., SANCHEZ-GONZALEZ A., BATTAGLIA P.: Learning mesh-based simulation with graph networks. In *ICLR* (2020). 2

[PGM*19] PASZKE A., GROSS S., MASSA F., LERER A., BRADBURY J., CHANAN G., KILLEEN T., LIN Z., GIMELSHEIN N., ANTIGA L., DESMAISON A., KOPF A., YANG E., DEVITO Z., RAISON M., TEJANI A., CHILAMKURTHY S., STEINER B., FANG L., BAI J., CHINTALA S.: Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems* (2019), Curran Associates, Inc. 1, 4

[RPK19] RAISSI M., PERDIKARIS P., KARNIADAKIS G. E.: Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational physics* (2019). 2

[SC23] STUYCK T., CHEN H.-Y.: Diffxpbd: Differentiable position-based simulation of compliant constraint dynamics. *Proc. ACM Comput. Graph. Interact. Tech.* (2023). 2

[SLD23] SHAO Y., LOY C. C., DAI B.: Towards multi-layered 3d garments animation. In *ICCV* (2023). 2

[SOC19] SANTESTEBAN I., OTADUY M. A., CASAS D.: Learning-based animation of clothing for virtual try-on. *Computer Graphics Forum* (2019). 1

[SOC22] SANTESTEBAN I., OTADUY M. A., CASAS D.: Snug: Self-supervised neural dynamic garments. In *CVPR* (2022). 2

[SWK24] STOTKO D., WANDEL N., KLEIN R.: Physics-guided Shape-from-Template: Monocular Video Perception through Neural Surrogate Models. In *CVPR* (2024). 2

[Wan18] WANG H.: Rule-free sewing pattern adjustment with precision and efficiency. *ACM Trans. Graph.* (2018). 1

[YLL17] YANG S., LIANG J., LIN M. C.: Learning-based cloth material recovery from video. In *ICCV* (Oct 2017). 2

© 2025 The Author(s).
Proceedings published by Eurographics - The European Association for Computer Graphics.