Mixed Variational Finite Elements for Implicit, General-Purpose Simulation of Deformables

ANONYMOUS AUTHOR(S)[*]

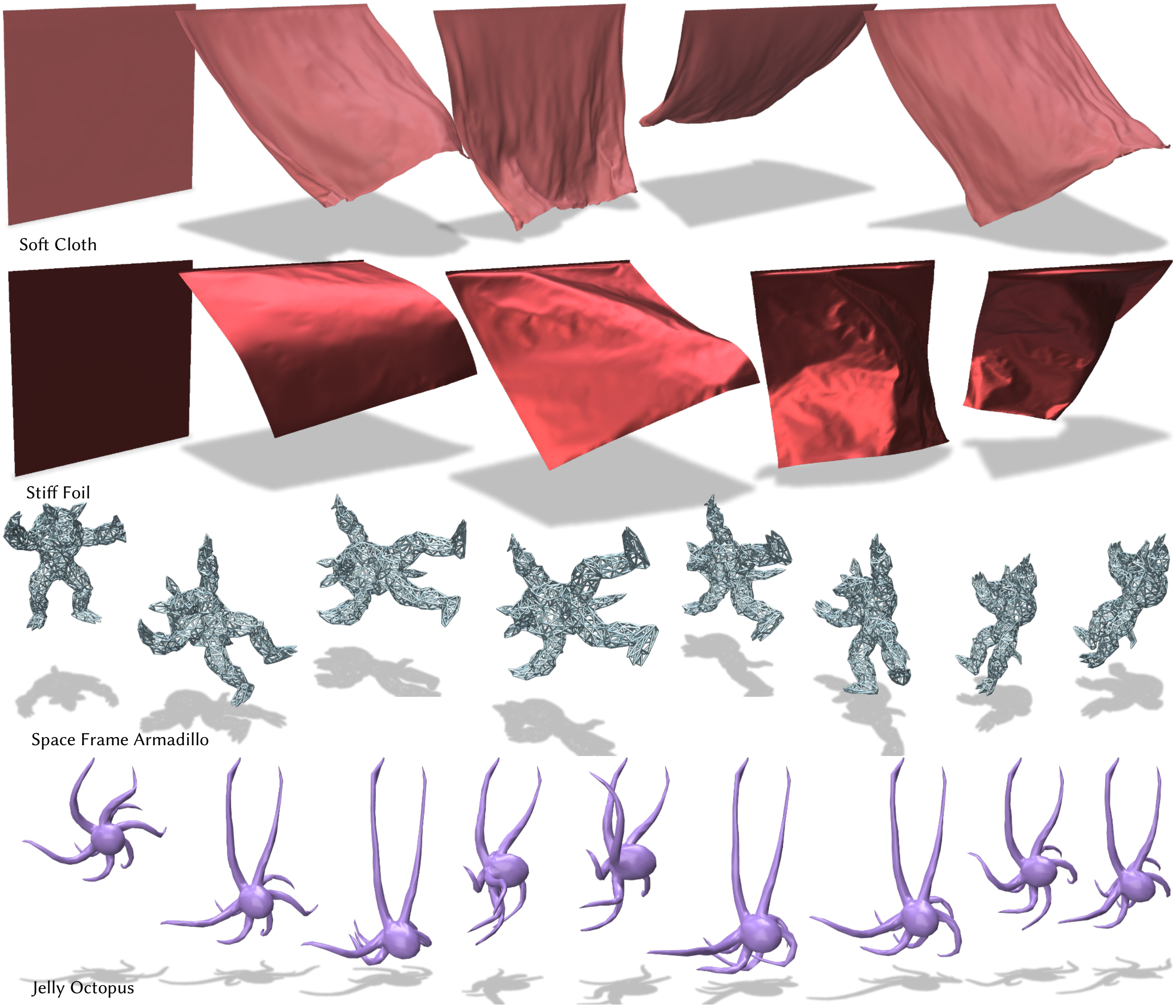

Fig. 1. Interactive dynamic simulations of elastic and rigid solid objects (volumes, shells and rods) using our unified mixed variational solver.

We propose and explore a new, general-purpose method for the implicit time integration of elastica. Key to our approach is the use of a mixed variational principle. In turn its finite element discretization leads to an efficient alternating projections solver with a superset of the desirable properties of many previous fast solution strategies. This framework fits a range of elastic constitutive models and remains stable across a wide span of timestep sizes, material parameters (including problems that are quasi-static and approximately rigid). It is efficient to evaluate and easily applicable to volume, surface, and rods models. We demonstrate the efficacy of our approach on a number of simulated examples across all three codomains.

CCS Concepts: • **Computer systems organization** → **Embedded systems**; *Redundancy*; Robotics; • **Networks** → Network reliability.

2022. XXXX-XXXX/2022/1-ART $15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

Additional Key Words and Phrases: physics-based animation, physics simulation

ACM Reference Format:
Anonymous Author(s). 2022. Mixed Variational Finite Elements for Implicit, General-Purpose Simulation of Deformables. 1, 1 (January 2022), 6 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

In this paper we explore the use of a mixed variational principle to build an efficient and general-purpose simulation algorithm for the physics-based animation of elastica.

Standard approaches for the implicit time integration of continua discretize with finite differences in time and finite elements in space. Recent methods often leverage the observation that, for these implicit time integration choices, each individual time step solve can then be cast as a minimization problem. In turn, the applied strategy for solving these optimization problems then leads to a wide range of well-known simulation algorithms [Li et al. 2019]. For example, a "standard" finite element approach involves minimizing an implicit integration energy via Newton's method while solving the bottleneck of inner linear-systems solves either via direct or iterative methods. Extended Position-Based Dynamics replaces standard direct or iterative solvers with iterations (e.g., GS, Jacobi, and/or SOR) acting on the dual variables (constraint forces) while Projective Dynamics and its more recent generalizations apply various forms of ADMM-type solver to split, augmented Lagrangian forms.

Despite their common variational origin, implicit solvers for elastica exhibit a wide range of features and limitations, and so tradeoffs. Standard Newton-type approaches and ADMM-based methods (including Projective Dynamics) exhibit various difficulties simulating stiff materials. On the other hand, Position-Based Dynamics, while fast and stable (even at high stiffness), lacks a direct correspondence to a discretization model and underlying PDE – it is then often unclear how to convert general material moduli and models to fit its constraint-based formulation.

Arguably an implicit time integrator for physics-based animation should

- support general elastic constitutive models;
- easily adapt to volumetric, surface and rod simulation;
- offer stability for a wide range of materials including those stiff enough to be effectively rigid;
- remain stable for large, frame-rate sized time steps; and
- provide efficient solutions of every time step.

Existing popular methods, as covered briefly here and below, are often custom-specialized to a small subset of these properties. Here, we explore a mixed variational finite-element model and an efficient solver for it, that covers the full spread of the above target properties.

2 RELATED WORK

Implicit integrators, especially Backward Euler, are ubiquitous in computer graphics. They were originally derived by introducing a velocity variable into Newton's second law, resulting in a nonlinear system of coupled first-order differential equations. This system of equations was solved via Newton's Method [Terzopoulos and Qin 1994] or single-step approximation [Baraff and Witkin 1998]. Forces were derived variationally from either distortion metrics [Terzopoulos and Qin 1994] or mechanical conditions on the simulated objects [Baraff and Witkin 1998]. Modern approaches derive and implement Backward Euler time integration from a variational perspective [Hairer and Lubich 2014; Martin et al. 2011] which, once discretized in time using finite differences, and in space using finite elements, yields a nonlinear minimization that must be solved once per simulation step. Efficient, robust and accurate solution of this optimization problem lies at the heart of physics-based animation research.

The "standard" solution approach is to apply Newton's method and use a direct solver to solve the resulting sequence of linear systems or to apply iterative solvers [Smith et al. 2018] for better runtime performance and memory usage. A highly efficient alternative approach is to apply a primal dual solver wherein the dual optimization problem is solved to compute physical forces from constraint equations. The most successful application of this approach is the Position Based Dynamics (PBD) algorithm [Müller et al. 2007] which acts on compliant constraints (similar to the condition energies from Baraff and Witkin [1998]), applying fast local updates (Gauss-Seidel or Successive-Over-Relaxation) to generate an approximate solution. Extended Position Based Dynamics (XPBD) uses a quadratic compliant dynamics formulation [Servin et al. 2006] to establish a relationship between PBD constraints and certain material models [Macklin et al. 2016] and allows the simulation of rigid and quasi-rigid bodies [Müller et al. 2020]. However the local nature of the PBD updates, necessitates additional complexity in achieving good convergence for high resolution meshes and large objects [Müller et al. 2017]. XPBD type solves also arise as intermediate steps in non-smooth Newton's Methods for contact and friction [Macklin et al. 2019].

Projective Dynamics [Bouaziz et al. 2014] applies the Alternating Direction Method of Multipliers algorithm [Overby et al. 2017] to minimize the integration energy. This leads to the incorporation of a global projection step which helps propagate forces across the mesh. Standard ADMM approaches suffer from convergence issues [Li et al. 2019], parameter tuning difficulties and difficulties simulating high stiffness materials. Impressive recent work addresses parameter tuning and improves convergence at the cost of more expensive global and local update steps [Brown and Narain 2021].

While the work described above has explored modifications at the solver level, one thing remains constant – the initial optimization form. Mixed variational principles in optimization have previously been explored for implicit integration. The Hamilton-Pontryagin princple [Kharevych et al. 2006], which treats position and displacement as independent variables coupled by constraints, has been used to derive discrete variational integrators. We are motivated by a different approach, that of Reissner [1985], which decouples deformation and displacements. In what follows we show that embarking from a Reissner-like mixed variational principle, and applying well-motivated design choices from previous work and optimization literature, naturally leads us to an efficient elastodynamics solver that, while superficially resembling fast solvers in geometry processing [Jacobson et al. 2012], exhibits a superset of features taken from the above algorithms. These include compatibility with general elastic constitutive models, stability for a wide range of timestep sizes and material parameters (including problems that are quasi-static and approximately rigid), as well as being easily applicable to volume, surface and rod simulation. We demonstrate the efficacy of our method on a number of examples simulated at interactive rates.

3 METHODS

Note: We use tensor notation in our derivations. Basic tensors and operations are described in our supplemental material. It's highly recommended to have it handy upon first reading.

Given an input finite element discretization with $|T|$ elements and $|V|$ vertices, the position-level optimization form of Backward Euler time integration [Hairer and Lubich 2014] is

$$\mathbf{q}^{t+1} = \arg\min_{\mathbf{q}} \frac{1}{2h^2}\mathbf{a}\left(\mathbf{q}\right)^T M\mathbf{a}\left(\mathbf{q}\right) + U\left(\mathbf{q}\right), \tag{1}$$

where $M \in \mathbb{R}^{3|V|\times 3|V|}$ is a mass matrix, $\mathbf{a}\left(\mathbf{q}\right) = \mathbf{q} - 2\mathbf{q}^t + \mathbf{q}^{t-1}$, all $\mathbf{q}^k \in \mathbb{R}^{3|V|}$ are stacked vectors of deformed vertex positions with t and $t-1$ denoting the current and previous time steps, and U is the elastic potential energy.

We immediately depart from standard derivations by replacing U with the mixed variational potential [Reissner 1985] which in its continuous form is given by

$$\tilde{U} = \int_\Omega \psi\left(S\left(\mathbf{X}\right)\right) - \lambda\left(\mathbf{X}\right) : \left(R\left(\mathbf{X}\right)S\left(\mathbf{X}\right) - F\left(\mathbf{X}\right)\right) d\Omega, \tag{2}$$

where Ω is the undeformed domain of the simulated object, ψ is any hyperelastic strain energy density, $\lambda \in \mathbb{R}^{3\times 3}$ are the Lagrange multipliers that enforce the deformation gradient constraint, $R \in SO^3$ is a local rotation matrix and $S \in \mathbb{R}^{3\times 3}$, a *symmetric* local deformation. $F \in \mathbb{R}^{3\times 3}$ is the deformation gradient at position $\mathbf{X} \in \mathbb{R}^3$ in the underformed domain. The inclusion of explicit rotations will allow our solver to rigidly rotate objects, which improves convergence, especially for stiff materials [Brown and Narain 2021]. Standard potential energies for body forces can be added to this potential, though we omit them here for the sake of brevity.

Finally, we form a discrete mixed potential by discretizing using a tetrahedral mesh with piecewise linear finite elements:

$$\begin{aligned}\tilde{U}^D = \sum_{i=1}^{|T|} &\left(\psi\left(\mathbf{C}\cdot\bar{P}_i\mathbf{s}\right) \cdots\right.\\ &\left. - \left(\mathbf{B}\cdot P_i\mathbf{l}\right) : \left(R_i\left(\mathbf{C}\cdot\bar{P}_i\mathbf{s}\right) - \left(\mathbf{D}^4\cdot N_i\mathbf{q}\right)G_i^3\right)\right) dv_i.\end{aligned} \tag{3}$$

Here $|T|$ is the number of tetrahedra in the mesh and subscript i denotes terms associated with the i^{th} tetrahedron. The variables $\mathbf{q} \in \mathbb{R}^{3|T|}$, $\mathbf{s} \in \mathbb{R}^{6|T|}$ and $\mathbf{l} \in \mathbb{R}^{9|T|}$ are stacked vectors of deformed vertex positions, entries of S and entries of λ respectively, and R_i is a local rotation matrix. Finally, $\bar{P}_i$ (resp. P_i and N_i) are sparse selection matrices that select from $\mathbf{s}$ (resp $\mathbf{l}$ and $\mathbf{q}$) the subset of the vectors associated with tetrahedron i. The remaining quantities, $\mathbf{B} \in \mathbb{R}^{3\times 3\times 9}$, $\mathbf{C} \in \mathbb{R}^{3\times 3\times 6}$, $\mathbf{D}^4 \in \mathbb{R}^{3\times 4\times 12}$, along with the tensor operations $\cdot$ and $:$ are used to convert from vector variables to matrix variables via tensor contraction. $G_i^3 \in \mathbb{R}^{4\times 3}$ is the gradient operator for tetrahedron i. See our supplemental material for their detailed definitions.

We compute the updated position of our deforming mesh by extremizing the mixed finite element integration energy

$$\mathbf{q}^{t+1}, \mathbf{s}^{t+1}, \mathbf{l} = \arg\max_{\mathbf{l}} \arg\min_{\mathbf{q},\mathbf{s}} \frac{1}{2h^2}\mathbf{a}\left(\mathbf{q}\right)^T M\mathbf{a}\left(\mathbf{q}\right) + \tilde{U}^D\left(\mathbf{q},\mathbf{s},\mathbf{l}\right), \tag{4}$$

3.1 A Newton-Like Solver

Our fast dynamics solver (Algorithm 1) performs Newton iteration using a fast alternating projections solver to compute search directions. The solver is applied to a quadratic approximation of Equation 4 in $\mathbf{q}$ and $\mathbf{s}$, given by

$$\begin{aligned}\Delta E|_{\mathbf{q}^k,\mathbf{s}^k} = &\frac{1}{2h^2}\Delta\mathbf{q}^T M\Delta\mathbf{q} - \frac{1}{h^2}\Delta\mathbf{q}^T M\mathbf{a}\left(\mathbf{q}^k\right)\cdots\\ &+ \frac{1}{2}\Delta\mathbf{s}^T H\Delta\mathbf{s} + \Delta\mathbf{s}^T\mathbf{g}\cdots\\ &- \mathbf{l}^T\left(W\left(\mathbf{s}^k + \Delta\mathbf{s}\right) - J\left(\mathbf{q}^k + \Delta\mathbf{q}\right)\right).\end{aligned} \tag{5}$$

Here $\mathbf{q}^k$ and $\mathbf{s}^k$ are the current guesses for $\mathbf{q}$ and $\mathbf{s}$, $H \in \mathbb{R}^{6|T|\times 6|T|}$ is a block diagonal matrix storing $\frac{\partial^2\psi}{\partial\mathbf{s}^2}|_{\bar{P}_i\mathbf{s}^k}$ in each diagonal block and $\mathbf{g}$ is the stacked vector of gradients, $\frac{\partial\psi}{\partial\mathbf{s}}|_{\bar{P}_i\mathbf{s}^k}$. We compact our tensors from Equation 3 into matrices; $J = \sum_{i=1}^{|T|} P_i^T\left(\mathbf{B}^T : \left(G_i^3\mathbf{D}^{4T}\right)\right)N_i$, $W = \sum_{i=1}^{|T|}\mathbf{Z} : R_i$ and $Z \in \mathbb{R}^{9\times 6\times 3\times 3} = \mathbf{B}\cdot\mathbf{C}$.

Importantly J is a constant matrix that can be assembled at initialization time and W is block diagonal which depends linearly on R_i, and can be updated quickly, in parallel.

Our method uses the search directions computed by minimizing Equation 5 in conjunction with a backtracking linesearch procedure. The crucial missing piece to minimize Equation 5 efficiently.

Algorithm 1 Our fast elastodynamics solver

```
Input: $\mathbf{q}^t, \mathbf{q}^{t-1}, \mathbf{s}^t$
Input: $n$                                  ▹ maximum number of outer iterations
Input: $m$                                  ▹ maximum number of inner iterations
  $ii \leftarrow 0$
  $jj \leftarrow 0$
  $\mathbf{l} \leftarrow 0$
  $\mathbf{q}^{\mathbf{t+1}} \leftarrow 2\mathbf{q}^{\mathbf{t}} - \mathbf{q}^{\mathbf{t-1}} + h^2M^{-1}\mathbf{f}_{ext}$   ▹ initalize with Forward Euler
  $\mathbf{s}^{\mathbf{t+1}} \leftarrow \mathbf{s}^{\mathbf{t}}$
  while Not Converged and $ii < n$ do
    Update $H, g, \mathbf{b}$
    while Not Converged and $jj<m$ do                                  ▹ §3.2
      $\forall i \in |T|, R_i \leftarrow$ procrustes $\left(N_i\mathbf{q}^{t+1}, \bar{P}_i\mathbf{s}^{t+1}, P_i\mathbf{l}\right)$   ▹ §3.2.2
      Update $W$
      $\mathbf{q}^{t+1}, \mathbf{l} \leftarrow$ global $\left(\mathbf{q}^{t+1}, \mathbf{q}^t, \mathbf{q}^{t-1}, \mathbf{s}^{t+1}R_1...R_{|T|}\right)$   ▹ §3.2.1
      $\mathbf{s}^{t+1} \leftarrow W^T\mathbf{l} - H^{-1}\mathbf{g}$
    end while
    $\mathbf{q}^{t+1}, \mathbf{s}^{t+1} \leftarrow$ backtracking linesearch
  end while
```

3.2 Alternating Projections Solver

Our incremental potential is quadratic in $\Delta\mathbf{q}$ and $\Delta\mathbf{s}$, but linear in our rotational variables. We apply alternating projections to minimize this energy, motivated by its prior successes when applied to similar objectives [Jacobson et al. 2012; Kugelstadt et al. 2018; Liu et al. 2013]. Each step of the alternating projections algorithm is consists of a global substep which updates $\Delta\mathbf{q}$ and $\Delta\mathbf{s}$ and a local substep which updates all per-element rotations.

3.2.1 Global Solve. Minimizing Equation 5 with respect to $\Delta\mathbf{q}$ and $\Delta\mathbf{s}$ leads to a standard indefinite KKT system [Wright and Nocedal 1999], which we further modify by eliminating $\mathbf{\Delta s}$. This recovers a generalized form of the compliant dynamic formulation of Servin et al. [2006]:

$$\begin{pmatrix} \frac{1}{h^2}M & J^T \\ J & -WH^{-1}W^T \end{pmatrix} \begin{pmatrix} \Delta\mathbf{q} \\ \mathbf{l} \end{pmatrix} = \begin{pmatrix} \frac{1}{h^2}M\mathbf{a} + \mathbf{f}_{ext} \\ W\mathbf{s}^k - J\mathbf{q}^k + WH^{-1}\mathbf{g} \end{pmatrix} \tag{6}$$

Preconditioned conjugate gradient can be applied directly to solve Equation 6 [Durazzi and Ruggiero 2003]. Updating the left-hand side of the system only requires recomputing $-WH^{-1}W^T$ which is block diagonal and can be done in parallel. The right-hand side is similarly quick to update each iteration. Our preconditioner is a constant form of Equation 6 with $W^T H^{-1} W = \frac{1}{\mu} I$ where μ is the first Lamé parameter of the simulated material. This preconditioner is constant and so we prefactorize once for the entire simulation. Like the fluid pressure solve, elastic finite element stresses have a nullspace. We find a small amount of Tikhonov Regularization (10^{-6}) added to the lower right-hand block of Equation 6 resolves all convergence issues. With $\mathbf{l}$ in hand we can compute $\Delta\mathbf{s} = H^{-1}\left(W^T\mathbf{l} - \mathbf{g}\right)$ in parallel for each tetrahedron.

3.2.2 Local Rotation Update. Minimizing Equation 5 directly for rotation variables leads to an unstable algorithm so instead our local rotation update minimizes the augmented Lagrangian form with quadratic penalty. This yields a fast, stabilized rotation update which is computed by solving

$$R_i = \arg\max_R \left\langle R, \left(\frac{1}{\beta}\lambda_i + F_i\right) S_i^T \right\rangle_F, \tag{7}$$

where $\langle\cdot,\cdot\rangle_F$ is the Frobenius inner product, $\lambda_i = \mathbf{B}\cdot(P_i\mathbf{l})$, $F_i = \mathbf{B}\cdot\left(P_i J\left(\mathbf{q}^{\mathbf{k}} + \mathbf{\Delta q}\right)\right)$ and $S_i = \mathbf{C}\cdot\bar{P}_i\left(\mathbf{s}^k + \Delta s\right)$. This orthogonal Procrustes problem [Eggert et al. 1997] can be solved efficiently via fast Singular Value Decomposition [McAdams et al. 2011], parallelized over each tetrahedron. Here β is the quadratic penalty parameter. Following guidance from Wright and Nocedal [1999] that β should be greater than $|P_i\mathbf{l}\|_\infty$ to guarantee convergence, we set $\beta = \alpha\cdot|P_i\mathbf{l}\|$, where $\alpha = 10$ initially and increases by 1.5× each substep.

3.3 Order of Local-Global Solves and Initial Guess

Our choice of ordering and initial guess is taken from previous work. Specifically both Müller et al. [2005] and Li et al. [2019] advise warm-starting simulations with a Forward Euler predictor while the Shape Matching algorithm [Müller et al. 2005] finds an initial, best-fit rigid rotation to this initial guess. Forward Euler initialization, combined with performing the local rotation first in the alternating projections sequence, effectively generalizes the Shape Matching predictor-corrector approach.

3.4 Constraints

Pinned vertices can be handled in the standard manner, by projecting them out of the global position vector $\mathbf{q}$. In our current proof-of-concept implementation we use penalty springs [Bridson et al. 2002] with hand tuned parameters for each example. We do not handle self-collisions in the current implementation.

3.5 Cloth and Rods

Our mixed formulation is trivially adaptable to cloth and rod simulation, requiring only four minor updates to the above algorithm:

(1) Rebuild J using $\mathbf{D}^3$ (resp. $\mathbf{D}^2$) and $\mathbf{G}^2$ (resp. $\mathbf{G}^1$) for cloth (rods).
(2) For cloth (resp. rods), add $\sum_{i=1}^{i=|T|} P_i^T\mathbf{B} : (R_i\mathbf{n}_i\mathbf{n}_i^T)$ (additionally $\sum_{i=1}^{i=|T|} P_i^T\mathbf{B} : (R_i\mathbf{n}_i'\mathbf{n}_i'^T)$ for rods) to the Lagrange multiplier right-hand side in Equation 6. Here $\mathbf{n}_i$ is the i^{th}, per-facet reference space normal, and $\mathbf{n}_i'$ the binormal.
(3) Replace S_i with $S_i - \mathbf{n}_i\mathbf{n}_i^T$ (resp. $S_i - \mathbf{n}\mathbf{n}^T - \mathbf{n}'\mathbf{n}'^T$) in the local step for cloth (rods).
(4) Replace integration volumes (Equation 3) with appropriate values.

Crucially we do not need to alter the strain energy density calculation in any way when simulating rods or cloth. No matter what the input discretization, material properties are specified using standard, volumetric models, which avoids complications when moving between different representations. This also enables interesting generalizations (ie fast simulation of nonlinear, Neohookean springs).

4 RESULTS AND DISCUSSION

All tests of our solver were performed on a MacBook Pro 13.3" (Apple M1 8-core CPU, 8-core GPU, 16GB Memory, 512GB SSD) using the opensource tools Eigen [Guennebaud et al. 2010] for linear algebra, SuiteSparse for direct solvers [Davis 2006], libigl [Jacobson et al. 2018] for geometry processing, Bartels [Levin 2020] for physics utility code and Polyscope [Sharp et al. 2019] for display. Due to time constraints optimizations were limited to obvious precomputation and multithreading opportunities. For all examples we run using an interactive configuration with a fixed number of outer and inner iterations (Algorithm 1, $n = 1$, m given in Table 1). We use a residual tolerance of $1e-7$ for our Conjugate Gradient solver, which is currently not assembly free. Despite the relatively low iteration counts our method generates a myriad of visually plausible results quickly (Table 1) and robustly.

Our solver is cable of simulating volumetric objects, surface only objects and objects made of rods/springs (Figure 2). We show the same Stanford Bunny mesh simulated using tetrahedral finite elements, triangular finite elements on the surface and springs placed along the edges of a coarse tetrahedral mesh. The volumetric FEM simulation has been skinned with a high resolution surface mesh. A Neohookean material model is used for all three simulations.

Table 1. Performance and parameters for all examples. Wall-clock timings are reported in milliseconds (ms), and physical parameters are reported with appropriate units. ρ is the density, **E** is the Young's Modulus, ν is Poisson's ratio. **Model** is the material model, which may be Neohookean (NH), Corotational (Corot), or ARAP. **Substeps** is the number of global substeps taken per timestep. The provided times represent the time taken in a single substep. **Assembly** is the time to assemble the left-hand side KKT system and the corresponding right-hand side. **KKT Solve** is the time to solve the KKT system, and **Rotation Solve** is the time taken for the local rotation update plus the time to compute and apply Δs.

Example	$\lvert q \rvert$	$\lvert T \rvert$	**Model**	ρ(kg/m^3)	**E(Pa)**	ν	**Substeps**	**Assembly (ms)**	**KKT Solve (ms)**	**Rotation Solve (ms)**
Square Cloth (soft)	5929	11552	NH	$1e^2$	$1e^5$	0.40	5	5.40	3.69	1.95
Square Cloth (rigid)	5929	11552	NH	$1e^3$	$1e^9$	0.40	5	3.47	3.37	2.03
Rod Armadillo	675	2019	NH	$1e^1$	$1e^7$	0.45	10	1.01	0.48	0.73
Jelly Octopus	452	1140	Corot	$1e^3$	$5e^5$	0.45	3	0.36	0.71	0.29
Beam ARAP	6000	29142	ARAP	$1e^3$	$1e^5$	0.45	5	8.24	14.9	3.06
Beam Corot	6000	29142	Corot	$1e^3$	$1e^5$	0.45	5	4.74	82.6	3.24
Bean NH	6000	29142	NH	$1e^3$	$1e^5$	0.45	5	8.88	169	3.21
Tet Bunny	699	2274	NH	$1e^3$	$1e^5$	0.45	5	0.90	1.47	0.33
Cloth Bunny	34834	69664	NH	$1e^2$	$1e^5$	0.40	5	17.38	16.3	7.18
Rods Bunny	500	2434	NH	$1e^2$	$2e^5$	0.45	5	0.95	1.19	0.44
Spot Drop (soft)	4079	15555	Corot	$1e^3$	$1e^6$	0.45	15	2.29	25.0	1.80
Spot Drop (rigid)	4079	15555	Corot	$1e^3$	$1e^9$	0.45	15	1.28	4.02	1.83

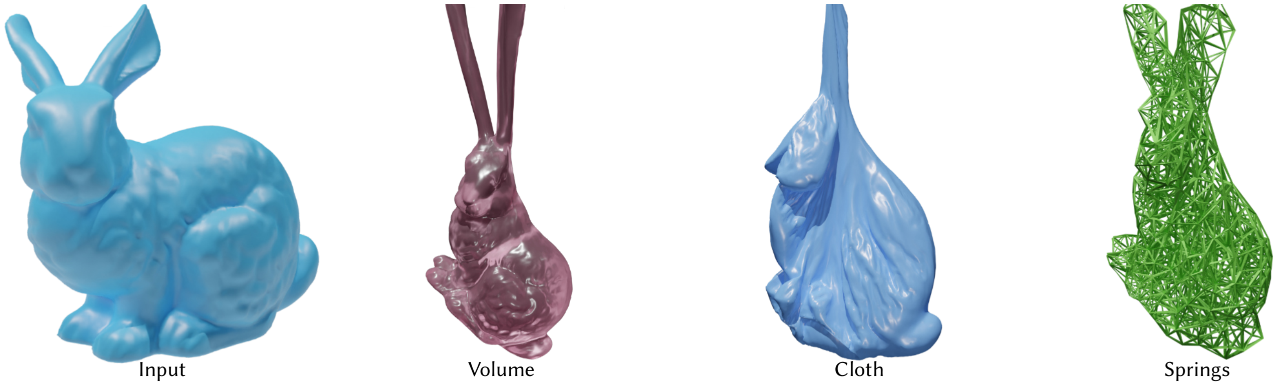

Fig. 2. Stanford Bunny Three Ways: The Stanford Bunny simulated as a volume, a shell and springs using the Neohookean material model.

We apply three different materials (ARAP, Corotational elasticity and Neohookean elasticity) to identical squares, which are then stretched via moving symmetric boundary conditions on the left and right sides (Figure 3). Because the ARAP model omits a volume preservation term, its deformation is purely orthogonal while ARAP and Neohookean material models exhibit different degrees of necking.

Next we demonstrate our solvers robustness in the face of changing material properties (Figure 4). These two cows are both simulated using the Neohookean material model but with vastly different stiffnesses(1GPa vs 1MPa). In both cases we observe stable, plausible simulations.

5 CONCLUSION AND FUTURE WORK

In this paper we have shown how judicious discretization of a mixed variational potential leads to an efficient, robust, flexible solver for elastodynamics. In order to promote further exploration of our approach we will release an open source version of our solver, under a permissive license along with this paper. While we think this new simulation framework is exceedingly exciting, we also enjoy how the mixed variational approach elucidates connections between existing simulation algorithms such as Position-Based Dynamics approaches and more standard solvers. While the solver presented here has both compelling features and performance, it is by no means the end of the mixed variational journey. One immediately attractive option would be to replace our preconditioned conjugate gradient solver with a multigrid approach. This would amount to using a multi-resolution position-based dynamics solver for the global step of our method. Integrating more robust contact handling such as incremental potential [Li et al. 2020] approaches would go a long way to broadening the practical applications of our current implementation. On the theoretical side, further exploring the connections between our solver and variants of ADMM, such as Douglas-Rachford Splitting, could draw further connections between methods and help improve convergence of the method.

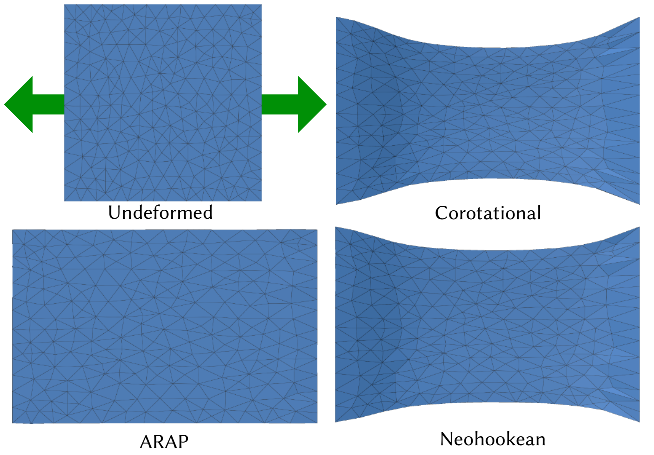

Fig. 3. Simulating stretching a square using ARAP, Corotational and Neohookean hyperelastic models.

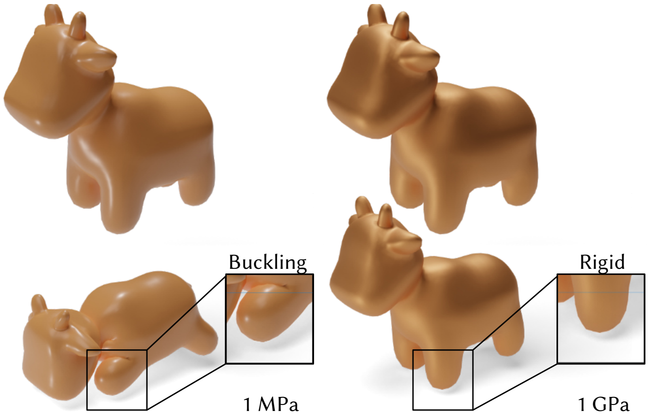

Fig. 4. Spot Drop Test: We simulate two cows being dropped from identical heights onto a ground plane. Our method is equally effective on relatively soft (1MPa) objects as it is on relatively stiff (1GPa) objects

.

REFERENCES

David Baraff and Andrew Witkin. 1998. Large Steps in Cloth Simulation. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '98)*. Association for Computing Machinery, New York, NY, USA, 43–54. https://doi.org/10.1145/280814.280821

Sofien Bouaziz, Sebastian Martin, Tiantian Liu, Ladislav Kavan, and Mark Pauly. 2014. Projective Dynamics: Fusing Constraint Projections for Fast Simulation. *ACM Trans. Graph.* 33, 4, Article 154 (July 2014), 11 pages.

Robert Bridson, Ronald Fedkiw, and John Anderson. 2002. Robust Treatment of Collisions, Contact and Friction for Cloth Animation. *ACM Trans. Graph.* 21, 3 (July 2002), 594–603.

George E. Brown and Rahul Narain. 2021. WRAPD: Weighted Rotation-aware ADMM for Parameterization and Deformation. *ACM Transactions on Graphics (Proc. SIGGRAPH)* 40, 4 (8 2021).

Timothy A Davis. 2006. *Direct methods for sparse linear systems.* SIAM.

Carla Durazzi and Valeria Ruggiero. 2003. Indefinitely preconditioned conjugate gradient method for large sparse equality and inequality constrained quadratic problems. *Numerical linear algebra with applications* 10, 8 (2003), 673–688.

David W Eggert, Adele Lorusso, and Robert B Fisher. 1997. Estimating 3-D rigid body transformations: a comparison of four major algorithms. *Machine vision and applications* 9, 5 (1997), 272–290.

Gaël Guennebaud, Benoît Jacob, et al. 2010. Eigen v3. http://eigen.tuxfamily.org.

Ernst Hairer and Christian Lubich. 2014. Energy-diminishing integration of gradient systems. *IMA J. Numer. Anal.* 34, 2 (2014), 452–461.

Alec Jacobson, Ilya Baran, Ladislav Kavan, Jovan Popović, and Olga Sorkine. 2012. Fast Automatic Skinning Transformations. *ACM Trans. Graph.* 31, 4 (2012), to appear.

Alec Jacobson, Daniele Panozzo, et al. 2018. libigl: A simple C++ geometry processing library. https://libigl.github.io/.

L. Kharevych, Weiwei Yang, Y. Tong, E. Kanso, J. E. Marsden, P. Schröder, and M. Desbrun. 2006. Geometric, Variational Integrators for Computer Animation. In *Proceedings of the 2006 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Vienna, Austria) *(SCA '06)*. Eurographics Association, Goslar, DEU, 43–51.

Tassilo Kugelstadt, Dan Koschier, and Jan Bender. 2018. Fast Corotated FEM using Operator Splitting. *Computer Graphics Forum (SCA)* 37, 8 (2018).

David I.W. Levin. 2020. Bartels: A lightweight collection of routines for physics simulation. https://github.com/dilevin/Bartels.

Minchen Li, Zachary Ferguson, Teseo Schneider, Timothy Langlois, Denis Zorin, Daniele Panozzo, Chenfanfu Jiang, and Danny M. Kaufman. 2020. Incremental Potential Contact: Intersection- and Inversion-free Large Deformation Dynamics. *ACM Trans. Graph. (SIGGRAPH)* 39, 4, Article 49 (2020).

Minchen Li, Ming Gao, Timothy Langlois, Chenfanfu Jiang, and Danny M. Kaufman. 2019. Decomposed Optimization Time Integrator for Large-Step Elastodynamics. *ACM Transactions on Graphics* 38, 4 (2019).

Tiantian Liu, Adam W. Bargteil, James F. O'Brien, and Ladislav Kavan. 2013. Fast Simulation of Mass-Spring Systems. *ACM Transactions on Graphics* 32, 6 (Nov. 2013), 209:1–7. http://cg.cis.upenn.edu/publications/Liu-FMS Proceedings of ACM SIGGRAPH Asia 2013, Hong Kong.

Miles Macklin, Kenny Erleben, Matthias Müller, Nuttapong Chentanez, Stefan Jeschke, and Viktor Makoviychuk. 2019. Non-Smooth Newton Methods for Deformable Multi-Body Dynamics. *CoRR* abs/1907.04587 (2019). arXiv:1907.04587 http://arxiv.org/abs/1907.04587

Miles Macklin, Matthias Müller, and Nuttapong Chentanez. 2016. XPBD: Position-Based Simulation of Compliant Constrained Dynamics. In *Proceedings of the 9th International Conference on Motion in Games* (Burlingame, California) *(MIG '16)*. Association for Computing Machinery, New York, NY, USA, 49–54. https://doi.org/10.1145/2994258.2994272

Sebastian Martin, Bernhard Thomaszewski, Eitan Grinspun, and Markus Gross. 2011. Example-Based Elastic Materials. In *ACM SIGGRAPH 2011 Papers* (Vancouver, British Columbia, Canada) *(SIGGRAPH '11)*. Association for Computing Machinery, New York, NY, USA, Article 72, 8 pages. https://doi.org/10.1145/1964921.1964967

Aleka McAdams, Yongning Zhu, Andrew Selle, Mark Empey, Rasmus Tamstorf, Joseph Teran, and Eftychios Sifakis. 2011. *Efficient Elasticity for Character Skinning with Contact and Collisions.* ACM, New York, NY, USA.

Matthias Müller, Nuttapong Chentanez, Miles Macklin, and Stefan Jeschke. 2017. Long Range Constraints for Rigid Body Simulations. In *Proceedings of the ACM SIGGRAPH / Eurographics Symposium on Computer Animation* (Los Angeles, California) *(SCA '17)*. Association for Computing Machinery, New York, NY, USA, Article 14, 10 pages. https://doi.org/10.1145/3099564.3099574

Matthias Müller, Bruno Heidelberger, Matthias Teschner, and Markus Gross. 2005. Meshless Deformations Based on Shape Matching. *ACM Trans. Graph.* 24, 3 (July 2005), 471–478.

Matthias Müller, Bruno Heidelberger, Marcus Hennix, and John Ratcliff. 2007. Position based dynamics. *Journal of Visual Communication and Image Representation* 18, 2 (2007), 109 – 118.

Matthias Müller, Miles Macklin, Nuttapong Chentanez, Stefan Jeschke, and Tae-Yong Kim. 2020. Detailed Rigid Body Simulation with Extended Position Based Dynamics. *Computer Graphics Forum* (2020). https://doi.org/10.1111/cgf.14105

Matthew Overby, George E. Brown, Jie Li, and Rahul Narain. 2017. ADMM $\supseteq$ Projective Dynamics: Fast Simulation of Hyperelastic Models with Dynamic Constraints. *IEEE Transactions on Visualization and Computer Graphics* 23, 10 (Oct 2017), 2222–2234. https://doi.org/10.1109/TVCG.2017.2730875

E. Reissner. 1985. On mixed variational formulations in finite elasticity. *Acta Mechanica* 56, 3-4 (1985), 117–125. https://doi.org/10.1007/bf01177113

Martin Servin, Claude Lacoursiere, and Niklas Melin. 2006. Interactive simulation of elastic deformable materials. In *SIGRAD 2006. The Annual SIGRAD Conference; Special Theme: Computer Games*. Citeseer.

Nicholas Sharp et al. 2019. Polyscope. www.polyscope.run.

Breannan Smith, Fernando De Goes, and Theodore Kim. 2018. Stable Neo-Hookean Flesh Simulation. *ACM Trans. Graph.* 37, 2, Article 12 (March 2018), 15 pages. https://doi.org/10.1145/3180491

Demetri Terzopoulos and Hong Qin. 1994. Dynamic NURBS with Geometric Constraints for Interactive Sculpting. *ACM Trans. Graph.* 13, 2 (April 1994), 103–136.

S J Wright and J Nocedal. 1999. Numerical optimization. (1999).