Fast Simulation of Deformable Characters with Articulated Skeletons in Projective Dynamics

Jing Li
University of Utah
Salt Lake City, UT, USA
jingli2070769@gmail.com

Tiantian Liu
Microsoft Research Asian
Beijing, China
ltt1598@gmail.com

Ladislav Kavan
University of Utah
Salt Lake City, UT, USA
ladislav.kavan@gmail.com

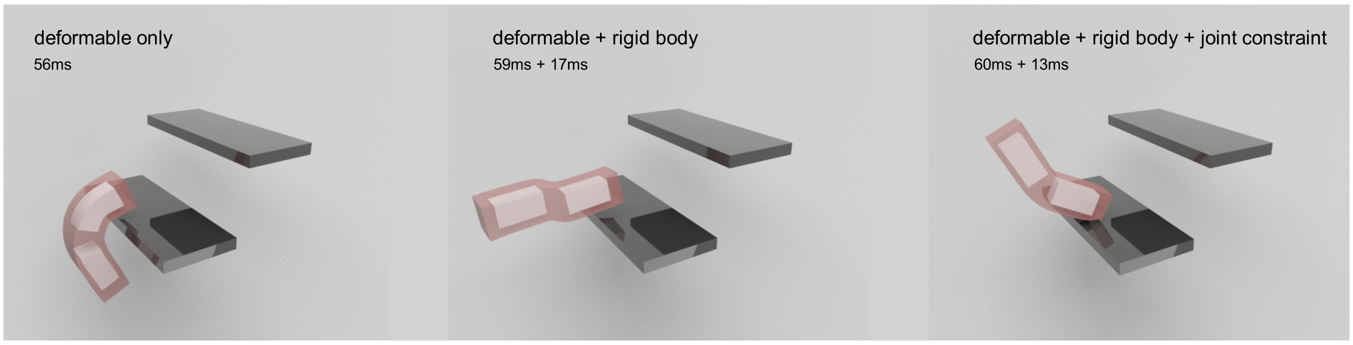

Figure 1: We propose a new constraint dynamics solver using backward Euler for both rigid and deformable degrees of freedom. The figure shows: a pure elastic body (left), elastic body with internal rigid bodies (middle), and using an articulated skeleton, i.e., rigid-body and joint constraints (right). Our method is fast and easy to implement; no explicit rigid body simulation or rigid-deformable coupling code is necessary.

ABSTRACT

We propose a fast and robust solver to simulate continuum-based deformable models with constraints, in particular, rigid-body and joint constraints useful for soft articulated characters. Our method embeds degrees of freedom of both articulated rigid bodies and deformable bodies in one unified optimization problem, thus coupling the deformable and rigid bodies. Our method can efficiently simulate character models, with rigid-body parts (bones) being correctly coupled with deformable parts (flesh). Our method is stable because backward Euler time integration is applied to rigid as well as deformable degrees of freedom. Our method is rigorously derived from constrained Newtonian mechanics. In an example simulation with rigid bodies only, we demonstrate that our method converges to the same motion as classical explicitly integrated rigid body simulator.

CCS CONCEPTS

• **Computing methodologies → Physical simulation**.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
SCA '19, July 26–28, 2019, Los Angeles, CA, USA
© 2019 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-1-4503-6677-9/19/07…$15.00
https://doi.org/10.1145/3309486.3340249

KEYWORDS

rigid bodies, deformable bodies, articulated, Projective Dynamics

ACM Reference Format:
Jing Li, Tiantian Liu, and Ladislav Kavan. 2019. Fast Simulation of Deformable Characters with Articulated Skeletons in Projective Dynamics. In *SCA '19:The ACM SIGGRAPH / Eurographics Symposium on Computer Animation (SCA '19), July 26–28, 2019, Los Angeles, CA, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3309486.3340249

1 INTRODUCTION

Realistic animation of articulated characters plays an important role in computer games, virtual reality and other interactive applications. The animation of articulated characters often models the body as a collection of rigid bodies ("Ragdoll physics") which can be computed quickly. However, virtual characters are usually based on real-world creatures, which often have deformable external parts (flesh) to interact with the environment (e.g., finger grasping). This external part is coupled with rigid internal structure (skeleton) that provide support. For example, believable animations of a human-like character, a snail, or a mermaid require simulation of bones or similar rigid structures, often further constrained with joints because two adjacent bones (e.g. upper and lower arm) usually cannot move arbitrarily. The mechanical interplay between rigid and deformable bodies results in numerically challenging simulation problems.

Projective Dynamics [Bouaziz et al. 2014] is an implicit Euler solver used in real-time deformable object simulation. It exploits a special potential energy structure which enables an efficient local/global solver, which often outperforms the classic Newton's method, making Projective Dynamics a suitable option for real-time simulation of deformable objects. However, Projective Dynamics (PD) assumes continuum-based elastic models. In theory, rigidity of the bones can be achieved by increasing the stiffness of the rigid parts. In practice however, this does not work very well, because with high stiffness ratios the convergence of PD deteriorates and more computationally expensive Newton-type solvers are recommended [Tournier et al. 2015]. The problem is that the global step of PD can only translate groups of vertices (e.g. four vertices in a tetrahedron), but not rotate them (the same problem is present also in Position Based Dynamics). Therefore, constraints with high stiffness effectively "lock" the orientation. In this paper, we avoid this problem by linearizing the $SE(3)$ manifolds, enabling exact (i.e. infinitely stiff) rigid-body constraints without locking. In Figure 6, we demonstrate an example of the PD locking problem and demonstrate how our proposed approach avoids it.

We propose a real-time simulation framework for soft articulated characters. Our method is derived from constrained formulation of Newtonian dynamics by using backward Euler time integration. In terms of the numerical solution, we show that we can combine the principles of Projective Dynamics and constrained dynamics into a unified optimization problem.

Contributions:

- We propose a unified method to simulate deformable characters with articulated skeletons in Projective Dynamics. Our method is fast, easy to implement and rigorously derived from Newton's laws via backward Euler time integration.
- We show that the special structure of potential energy from Projective Dynamics retains its numerical benefits when we reorganize the computations in a certain way. This allows us to benefit from the local/global solver in a similar way as the original, unconstrained Projective Dynamics.
- The vertices corresponding to rigid bodies are represented only via 12-dimensional affine subspaces, resulting in similar efficiency as classical rigid-body treatment with 6 DoF per rigid body. Our formulation is monolithic, i.e., there is no separate treatment of rigid and deformable degrees of freedom, transfer of momenta etc., simplifying implementation.
- Our method enforces the rigid-body and joint constraints exactly, while exhibiting fast convergence due to the linearization of the $SE(3)$ manifolds (Lie algebras).

2 RELATED WORK

2.1 Deformable body simulation

Deformable bodies can be simulated by mass-spring systems [Chen et al. 1998], finite difference method [Terzopoulos et al. 1987], finite element method (FEM) [Debunne et al. 2001; Müller et al. 2002], finite volume method (FVM) [Teran et al. 2003], boundary element method (BEM) [James and Pai 1999] etc. Position Based Dynamics (PBD) [Müller et al. 2007] is one of the popular methods to simulate deformable bodies in real-time applications. PBD assumes infinitely stiff energy potentials and the material stiffness depends on iteration count and time step, which is problematic in a scene with objects of varied stiffness, e.g.: soft bodies interacting with nearly rigid bodies [Macklin et al. 2016]. PBD is solved in a Gauss-Seidel fashion, which is stable and easy to implement, but does not converge rapidly. Extended position-based dynamics (XPBD) introduces a total Lagrange multiplier to PBD, which converges to the actual solution of backward Euler integration with physically correct stiffness. But similar to PBD, XPBD suffers from slow convergence speed of Gauss-Seidel iteration and low accuracy of only first order approximation of backward Euler integrator. Those shortcomings of PBD and XPBD are addressed by Projective Dynamics (PD) [Bouaziz et al. 2014] with rigor and higher accuracy from continuum mechanics. Projective Dynamics was derived as an extension of ShapeUp [Bouaziz et al. 2012] to dynamics. ShapeUp is an optimization system for geometry processing which also supports rigid-body constraints, but these are only soft constraints and dynamics is not considered. Increasing the stiffness of the rigid-body constraints would result in similar deterioration of convergence as in Projective Dynamics [Tournier et al. 2015].

Projective Dynamics can be seen as a quasi-Newton method [Liu et al. 2017] or an alternating direction method of multipliers (ADMM) [Narain et al. 2016], and therefore support more general materials. Fast GPU implementations of Projective Dynamics are also possible. Those methods can be further accelerated using Chebyshev semi-iterative approach [Wang 2015], a colored Gauss-Seidel method [Fratarcangeli et al. 2016], or a hyper-reduce scheme [Brandt et al. 2018]. [Kugelstadt et al. 2018] follow a similar strategy of Projective Dynamics by using an operator splitting approach to speed up the stretching resistance part. Moreover, it formulates the resistance to volume change as compliant constraints and introduces analytic polar decomposition(APD) to compute the rotational part of the deformation gradient. [Soler et al. 2018] simulate Cosserate rods with Projective Dynamics by incorporating body orientation in standard PD solver. [Peng et al. 2018] speeds up the convergence rate of Projective Dynamics for an accurate solution by applying Anderson acceleration, a well-established technique for fixed point iteration method.

2.2 Articulated rigid body simulation

There is a large body of classical work in computer graphics to simulate articulated rigid bodies [Baraff 1996; Bender et al. 2014; Featherstone 1987; Weinstein 2007]. [Armstrong and Green 1985] incorporates dynamics into the model of the human figure, given the forces and torques applied to joints at key points in the animation. [Witkin and Kass 1988] proposes a general tool to create articulated character animation by using spacetime constraints. [Weinstein et al. 2006] propose an iterative solution for handling joint and large numbers of unpredictable contact and collision events for rigid articulated bodies. [Zordan et al. 2005] incorporates motion capture data to make the articulated skeleton respond to unexpected impact.

2.3 Coupling between rigid and deformable bodies

Coupling rigid and deformable bodies is an interesting topic that has been explored by many authors [Baraff and Witkin 1997; Jansson and Vergeest 2003; Lenoir and Fonteneau 2004; O'Brien et al. 2000; Sifakis et al. 2007]. One common approach is to simulate each subsystem with specialized technique and then bridge the two together [Shinar 2008]. Specifically, [Shinar et al. 2008] designed a full two-way coupling of rigid and deformable bodies. It consists of 5 major steps to interleave the rigid and deformable body simulation in addition to the simulation of both the rigid and deformable degrees of freedom. ArtiSynth [Lloyd et al. 2012], an open source package used for modeling and simulating complex anatomical systems, is composed of both rigid and deformable bodies. It uses FEM to simulate deformable bodies and uses multibody techniques [Shabana 2005] to simulate rigid bodies separately, and couple these two parts together using attachment constraints. It is also common to simulate characters via physically-based skinning [Capell et al. 2005; Hahn et al. 2012; Kavan and Sorkine 2012; Liu et al. 2013; McAdams et al. 2011], however, with these methods the motion of the skeleton is specified kinematically, i.e., is not subject to physics-based simulation. [Galoppo et al. 2007a] combines articulated-body dynamics and skin deformation, which is expressed in pose space (rest configuration). It then applies displacement corrections from deformation to skinning. However, this method is not suitable for large global deformations, such as highly flexible characters. [Tournier et al. 2015] proposes an offline method for high stiffness material and constraints. [Tournier et al. 2015] also pointed out that Projective Dynamics results in artificial damping or even locking if the relative material stiffness is too high. Instead of using complete geometric stiffness[Andrews et al. 2017], Projective Dynamics only keeps the constant term in the geometric stiffness. [Kim and Pollard 2011] simulates skeleton-driven deformable body characters, which uses mesh embedding to reduce the DOFs of the deformable bodies. [Verschoor et al. 2018] defines the shape of the bone implicitly as capsule. The distance between the representative point on the bone and the interpolation point (from 4 vertices in a tetrahedron) is minimized as the energy term, whereas our method can simulate bones of various shape such as skulls.

3 METHOD

Continuous equations of motion with constraints can be written as [Lanczos 1986]:

$$\mathbf{M}\mathbf{a}(t) = -\nabla E(\mathbf{x}(t)) + \nabla C(\mathbf{x}(t))^{\mathsf{T}}\lambda(t) + \mathbf{f}_{\text{ext}} \tag{1a}$$

$$C(\mathbf{x}(t)) = 0 \tag{1b}$$

where $\mathbf{M}$ is a mass matrix, $\mathbf{a}(t)$ denotes acceleration and $\mathbf{C} : \mathbb{R}^{n\times 3} \to \mathbb{R}^{n_c\times 1}$ is an implicit function for the number of constraints n_c. $\nabla E(\mathbf{x}(t))$ and $\mathbf{f}_{\text{ext}} \in \mathbb{R}^{n\times 3}$ represent the elastic and external forces and $\lambda(t) \in \mathbb{R}^{n_c\times 1}$ is a vector of Lagrange multipliers.

Backward Euler time discretization corresponds to the following update rules:

$$\begin{aligned} \mathbf{x}_{n+1} - \mathbf{x}_n &= h\mathbf{v}_{n+1} \\ \mathbf{v}_{n+1} - \mathbf{v}_n &= h\mathbf{a}_{n+1} \end{aligned} \tag{2}$$

where $\mathbf{x}_{n+1}$ is the current state, $\mathbf{x}_n$ is the previous state, h is the time step, $\mathbf{v}_{n+1}$ is the current velocity, $\mathbf{v}_n$ is the previous velocity, and $\mathbf{a}_{n+1}$ is the current acceleration. From Eq. 2, we can write:

$$\mathbf{a}_{n+1} = \frac{1}{h^2}(\mathbf{x}_{n+1} - 2\mathbf{x}_n + \mathbf{x}_{n-1}) \tag{3}$$

Applying the backward Euler discretization to Eq. 1a we obtain:

$$\frac{\mathbf{M}}{h^2}(\mathbf{x} - \mathbf{y}) = -\nabla E(\mathbf{x}) + \nabla C(\mathbf{x})^{\mathsf{T}}\lambda \tag{4}$$

where $\mathbf{x}$ is a shorthand for $\mathbf{x}_{n+1}$ and the constant $\mathbf{y}$ is defined as: $\mathbf{y} := 2\mathbf{x}_n - \mathbf{x}_{n-1} + h^2\mathbf{M}^{-1}\mathbf{f}_{\text{ext}}$.
Together with $C(\mathbf{x}) = 0$, Eq. 4 can be interpreted as setting to zero the gradient of the Lagrangian of the following constrained optimization problem:

$$\min_{\mathbf{x}} \quad \frac{1}{2h^2}\|\mathbf{x} - \mathbf{y}\|_{\mathbf{M}}^2 + E(\mathbf{x}) \tag{5a}$$

$$\text{s.t.} \quad C(\mathbf{x}) = 0 \tag{5b}$$

This constrained optimization problem can be understood intuitively: Eq. 5a is nothing but the classical objective of unconstrained backward Euler [Martin et al. 2011], which is restricted to satisfy the constraints (Eq. 5b). Even though simply adding Eq. 5b to the original unconstrained problem might be seen as an ad-hoc way to enforce the constraints, we would like to emphasize that Eq. 5 is rigorously derived from constrained Newtonian dynamics (Eq. 1) with backward Euler time integration.

Previous work [Bouaziz et al. 2014] demonstrated that fast and robust optimizers can be derived by introducing auxiliary "projection" variables. The key contribution of this paper consists in generalizing these methods to the case of non-trivial hard constraints $C(\mathbf{x}) = 0$. Specifically, our method includes exact rigid-body constraints (in contrast to approximately-rigid bodies [Macklin et al. 2014; Müller et al. 2005]) and exact joint constraints. This is not a trivial extension because the rigid-body constraints are non-convex and joint-constraints between rigid bodies (e.g. bones) introduce coupling between the individual rigid-body constraints.

Our method is essentially a fast and robust solver for the constrained optimization problem in Eq. 5a and Eq. 5b, exploiting special structure of the manifolds of rigid body constraints $SE(3)$. In the following, we derive our final algorithm in two steps. First, we restrict vertices belonging to one rigid body to a 12-dimensional affine subspace (Section 3.1). Second, we further restrict the affine degrees of freedom to the 6-dimensional non-linear manifold $SE(3)$ while taking joint-coupling into account (Section 3.2). The final algorithm is summarized in Alg. 1.

3.1 Step 1: Affine Reduction

Projective Dynamics [Bouaziz et al. 2014] is an efficient numerical solver for deformable body simulation. It treats backward Euler integration as an unconstrained optimization problem formulated in Eq. 5a. Projective Dynamics solves this unconstrained optimization problem using a local-global strategy, where the global step can be described as follows:

$$\min \quad \frac{1}{2h^2}\|\mathbf{x} - \mathbf{y}\|_{\mathbf{M}}^2 + \frac{1}{2}\mathbf{x}^{\mathsf{T}}\mathbf{L}\mathbf{x} - \mathbf{x}^{\mathsf{T}}\mathbf{J}\mathbf{p} \tag{6}$$

where $\mathbf{L}$ and $\mathbf{J}$ are two constant matrices determined by the mesh topology and material stiffness, $\mathbf{p}$ is a projection vector computed by the local step. Projective Dynamics is a fast solver but can not handle nonlinear hard constraints in Eq. 5b. In order to take advantage of Projective Dynamics, we can first group the rigid body vertices together using affine constraints.

The purpose of this first (preparatory) step is to remove redundant degrees of freedom. Specifically, an arbitrary number of vertices corresponding to a rigid body is reduced to 12 degrees of freedom. This corresponds to classical linear model reduction and we do not claim any technical innovations in this step. The purpose is to set the stage for the second step which tackles the main problem of exactly enforcing coupled rigid-body constraints (Section 3.2). We believe that this decoupling improves the exposition of our method as well as structure of our code (implementation).

Our system is composed of $n = n_b + n_f$ vertices where n_f is the number of flesh vertices and n_b is the number of bone vertices. We reorder the vertices such that all of the flesh vertices come first and bone vertices are grouped together according to the bones they correspond to:

$$\mathbf{x} = \begin{bmatrix} \mathbf{x}_f \\ \mathbf{x}_b \end{bmatrix} \tag{7}$$

where $\mathbf{x} \in \mathbb{R}^{n\times 3}$ denotes the positions of all vertices, $\mathbf{x}_f \in \mathbb{R}^{n_f\times 3}$ the positions of flesh vertices, and $\mathbf{x}_b \in \mathbb{R}^{n_b\times 3}$ the positions of bone vertices.

Next, we restrict vertices belonging to one rigid body to an affine transformation of their rest pose positions:

$$\mathbf{x}_b = \mathbf{V}\mathbf{T} \tag{8}$$

$$\mathbf{V} = \left[\begin{array}{c|c|c|c} \mathbf{V}_1 & 0 & 0 & 0 \\ \hline 0 & \mathbf{V}_2 & 0 & 0 \\ \hline & & \ddots & \\ 0 & 0 & & 0 \\ \hline 0 & 0 & 0 & \mathbf{V}_m \end{array}\right] \tag{9}$$

where $\mathbf{V}_k$ is the rest-pose vertex positions of bone k, and $\mathbf{V} \in \mathbb{R}^{n_b\times 4m}$ consists of $\mathbf{V}_k$ shown in Eq. 9, m is the number of bones. For every bone k, there is an associated affine transformation matrix $\mathbf{T}_k \in \mathbb{R}^{3\times 4}$ (which will be eventually restricted to $SE(3)$ in Section 3.2). We stack all of the transformation matrices into a single matrix $\mathbf{T} = \begin{bmatrix}\mathbf{T}_1 & \mathbf{T}_2 & \dots & \mathbf{T}_m\end{bmatrix}^\mathsf{T} \in \mathbb{R}^{4m\times 3}$.

Substituting Eq. 8 into Eq. 7 can be written as:

$$\mathbf{x} = \begin{bmatrix} \mathbf{x}_f \\ \mathbf{V}\mathbf{T} \end{bmatrix} = \underbrace{\begin{bmatrix} \mathbf{I}_{n_f} & 0 \\ 0 & \mathbf{V} \end{bmatrix}}_{\mathbf{B}} \underbrace{\begin{bmatrix} \mathbf{x}_f \\ \mathbf{T} \end{bmatrix}}_{\hat{x}} \tag{10}$$

where $\mathbf{I}_{n_f}$ is the identity matrix, $\mathbf{B} \in \mathbb{R}^{(n_f+n_b)\times(n_f+4m)}$ is a constant matrix, and $\hat{\mathbf{x}} \in \mathbb{R}^{(n_f+4m)\times 3}$ becomes our new reduced variable. We can substitute $\hat{\mathbf{x}}$ into Eq. 6 with auxiliary (projection) variables $\mathbf{p}$ to get our new objective function:

$$\min \frac{1}{2h^2}||(\mathbf{B}\hat{\mathbf{x}} - \mathbf{y})||^2_\mathbf{M} + \frac{1}{2}\hat{\mathbf{x}}^\mathsf{T}\mathbf{B}^\mathsf{T}\mathbf{L}\mathbf{B}\hat{\mathbf{x}} - \hat{\mathbf{x}}^\mathsf{T}\mathbf{B}^\mathsf{T}\mathbf{J}\mathbf{p} \tag{11}$$

This new objective function is essentially a subspace version of the global step of Projective Dynamics [Brandt et al. 2018]. The constant $\mathbf{y}$ is computed as $\mathbf{y} := \mathbf{x}_n + h\mathbf{v}_n$ where $\mathbf{x}_n$ is the current positions and $\mathbf{v}_n$ is the current velocity. We compute $\mathbf{x}$ by using Eq. 10 after $\hat{\mathbf{x}}$ is solved for by Eq. 11, thus rigid body dynamics is not needed when determining $\mathbf{y}$ in the rigid part. With $\mathbf{p}$ fixed, the optimal $\hat{\mathbf{x}}$ is computed by setting the derivative of Eq. 11 to zero:

$$\underbrace{\mathbf{B}^\mathsf{T}(\frac{\mathbf{M}}{h^2} + \mathbf{L})\mathbf{B}}_{\text{prefactor}}\,\hat{\mathbf{x}} = \mathbf{B}^\mathsf{T}\left(\frac{\mathbf{M}\mathbf{y}}{h^2} + \mathbf{J}\mathbf{p}\right) \tag{12}$$

where pre-factorization is a possible acceleration strategy because $\mathbf{B}$, $\mathbf{M}$ and $\mathbf{L}$ are all constant matrices. We do not claim any technical novelty in this step, but this preparation will be useful in the next section.

3.2 Step 2: Non-linear Constraints

In this section we combine the affinely-reduced formulation from the previous section with rigid-body constraints. If desired, the rigid-body constraints can be also coupled with joint constraints (e.g., to model articulated skeletons). Writing out our constrained optimization problem in more detail:

$$\min_{\hat{\mathbf{x}}} \quad \frac{1}{2h^2}||\mathbf{B}\hat{\mathbf{x}} - \mathbf{y}||^2_\mathbf{M} + E(\hat{\mathbf{x}}, \mathbf{p}) \tag{13a}$$
$$\text{s.t.} \quad \mathbf{T}_k \in SE(3) \quad \text{for k = 1...m} \tag{13b}$$
$$\mathbf{T}_j\mathbf{q} = \mathbf{T}_k\mathbf{q} \quad \text{if the } j\text{- and } k\text{-th bones share a joint } \mathbf{q} \tag{13c}$$

where $\mathbf{T}_k$ is the transformation matrix for bone k, $\mathbf{q} \in \mathbb{R}^{4\times 1}$ Eq. 13c means that a joint transformed with bone k should be in the same position after being transformed with bone j if bone k and j share the joint $\mathbf{q}$.
For a given projection $\mathbf{p}$, we first minimize Eq. 13a using Eq. 12 and denote the result of this unconstrained minimization problem as $\mathbf{x}^0$. We then project the constraints to satisfy Eq. 13b and Eq. 13c while deviating from $\mathbf{x}^0$ as little as possible.

Enforcing Eq. 13b alone by itself is a well known problem called Procrustes problem that can be solved using singular value decomposition [Sorkine-Hornung and Rabinovich 2017]. The hard part is to *also* satisfy Eq. 13c without violating the $\mathbf{T}_k \in SE(3)$ constraints. In the first step (Section 3.1), we have relaxed the $SE(3)$ constraints to affine ones. These affine transformations $\mathbf{T}^0 = \begin{bmatrix}\mathbf{T}^0_1 & \mathbf{T}^0_2 \dots \mathbf{T}^0_m\end{bmatrix}^\mathsf{T}$ give us an initial guess for solving the non-convex optimization problem in Eq. 13.

When solving for Eq. 13b and Eq. 13c together, instead of restricting $\mathbf{T}_j$ and $\mathbf{T}_k$ to rigid body transformations directly, we restrict them to *linearized* rigid body transformations first and reparameterize the 12-DoF matrix $\mathbf{T}_k$ with two 3-DoF vectors: $\boldsymbol{\omega}_k$ for linearized rotation (angular velocity) and $\mathbf{l}_k$ for translation. The new variables $\boldsymbol{\omega}_k$ and $\mathbf{l}_k$ parametrize the tangent space (the Lie algebra) of $SE(3)$. Formally, at any given rigid-body transformation $\begin{bmatrix}\mathbf{R}^0_k & \mathbf{t}^0_k\end{bmatrix} \in SE(3)$, we can introduce the following substitution:

$$\mathrm{T}_k = \underbrace{\begin{bmatrix} R^0_k & \mathbf{t}^0_k \end{bmatrix}}_{3\times 4} \underbrace{\begin{bmatrix} \mathbf{I} + \boldsymbol{\omega}^*_k & \mathbf{l}_k \\ \mathbf{0} & 1 \end{bmatrix}}_{4\times 4} \tag{14}$$

where $\boldsymbol{\omega}^*_k \in \mathbb{R}^{3\times 3}$ denotes the cross-product matrix of $\boldsymbol{\omega}_k$. We compute $\mathbf{R}^0_k$ and $\mathbf{t}^0_k$ by solving the Procrustes problem using SVD [Sorkine-Hornung and Rabinovich 2017] to best fit the affinely transformed bone vertices, denoted as $\mathbf{x}^0_b = \mathbf{V}\mathbf{T}^\mathbf{0}$.

Next, we can solve the linearized version of Eq. 13b along with the (already linear) Eq. 13c as a convex quadratic minimization problem with linear equality constraints:

$$\min_{\mathbf{s}} \quad \mathbf{s}^\mathsf{T}\mathbf{Q}\mathbf{s} + \mathbf{c}^\mathsf{T}\mathbf{s} \tag{15a}$$

$$\text{s.t.} \quad \mathbf{C}\mathbf{s} = \mathbf{r} \tag{15b}$$

where $\mathbf{s} = \begin{bmatrix}\boldsymbol{\omega}_1^\mathsf{T} & \mathbf{l}_1^\mathsf{T} \dots & \boldsymbol{\omega}_m^\mathsf{T} & \mathbf{l}_m^\mathsf{T}\end{bmatrix}^\mathsf{T} \in \mathbb{R}^{6\times 1}$. Then $\mathbf{T}_k$ can be viewed as a function of $\mathbf{s}$ according to Eq. 14. Solving Eq. 15a and Eq. 15b exactly amounts to a small ($6m \times 6m$, where m is the number of bones) linear system solve and is thus very fast. The construction of the matrices $\mathbf{Q}$, $\mathbf{C}$ and the vectors $\mathbf{c}$ and $\mathbf{r}$ is explained in detail in the following subsections. Due to the linearization of $SE(3)$, this solve has to be iterated with the Procrustes projection back to $SE(3)$, which can be done independently for each bone. This entire process can be interpreted as generalization of Procrustes projection to multiple rigid constraints combined with joint constraints, i.e., articulated rigid bodies.

3.2.1 reparameterization of the linear constraints. First, we express Eq. 13c in terms of the new unknown $\mathbf{s}$ by using Eq. 14. We can rewrite the position of a joint $\mathbf{q}$ as:

$$\mathbf{T}_k\mathbf{q} = \boldsymbol{R}_k^0\left(\mathbf{q} + \mathbf{l}_k - (\mathbf{q}^*)\boldsymbol{\omega}_k\right) + \mathbf{t}_k^0 \tag{16}$$

where $\mathbf{q}^*$ denotes the cross-product matrix of $\mathbf{q}$. Assuming that $\boldsymbol{\omega}_k$ and $\mathbf{t}_k$ are small, we substitute Eq. 16 into Eq. 13c, obtaining:

$$\underbrace{(\boldsymbol{R}_j^0 - \boldsymbol{R}_k^0)\mathbf{q} + \mathbf{t}_j^0 - \mathbf{t}_k^0}_{\mathbf{r}} = \underbrace{\boldsymbol{R}_j^0(\mathbf{q}^*)\boldsymbol{\omega}_j - \boldsymbol{R}_k^0(\mathbf{q}^*)\boldsymbol{\omega}_k - \boldsymbol{R}_j^0\mathbf{l}_j + \boldsymbol{R}_k^0\mathbf{l}_k}_{\mathbf{C}\mathbf{s}} \tag{17}$$

The linearized constraints matrix $\mathbf{C}$ is assembled by collecting Eq. 17 over all joints, see Figure 3. Stacking the contribution of all joint constraints together will get us Eq. 15b.

3.2.2 reparameterization of the objective. Intuitively, we are solving for rigid bone transformations $\mathbf{T}$ while staying as close as possible to the initial guess $\mathbf{x}_b^0$. So we can write Eq. 15a as:

$$\min_{\mathbf{s}} \sum_k \|\mathbf{V}_k\mathbf{T}_k^\mathsf{T}(\mathbf{s}) - \mathbf{x}_{b,k}^0\|_\mathbf{M}^2 \tag{18}$$

where $\mathbf{V}_k$ is the rest-pose vertices for bone k, $\mathbf{x}_{b,k}^0$ stands for the initial guess we computed from Eq. 12. We can substitute Eq. 14 into $\mathbf{V}_k\mathbf{T}_k$ to get:

$$\|\mathbf{V}_k\mathbf{T}_k^\mathsf{T} - \mathbf{x}_{b,k}^0\|_\mathbf{M}^2 = \sum_i m_i\|\boldsymbol{R}_k^0(\mathbf{I} + \boldsymbol{\omega}_k^*)\mathbf{V}_{k,i} + \boldsymbol{R}_k^0\mathbf{l}_k + \mathbf{t}_k^0 - \mathbf{x}_{b,k,i}^0\|^2 \tag{19a}$$

$$= \sum_i m_i\|\boldsymbol{R}_k^0\mathbf{l}_k - \underbrace{\boldsymbol{R}_k^0(\mathbf{V}_{k,i}^*)}_{\mathbf{Q}_i^0}\boldsymbol{\omega}_k + \underbrace{\boldsymbol{R}_k^0\mathbf{V}_{k,i} + \mathbf{t}_k^0 - \mathbf{x}_{b,k,i}^0}_{\mathbf{e}_i}\|^2 \tag{19b}$$

$$= \begin{bmatrix}\boldsymbol{\omega}_k^\mathsf{T} & \mathbf{l}_k^\mathsf{T}\end{bmatrix}\begin{bmatrix}\sum_i m_i\mathbf{Q}_i^{0\mathsf{T}}\mathbf{Q}_i^0 & -\sum_i m_i\mathbf{Q}_i^{0\mathsf{T}}\boldsymbol{R}_k^0 \\ -\sum_i m_i\boldsymbol{R}_k^{0\mathsf{T}}\mathbf{Q}_i^0 & \sum_i m_i\boldsymbol{R}_k^{0\mathsf{T}}\boldsymbol{R}_k^0\end{bmatrix}\begin{bmatrix}\boldsymbol{\omega}_k \\ \mathbf{l}_k\end{bmatrix} \tag{19c}$$

$$+ 2\mathbf{l}_k^\mathsf{T}(\sum_i m_i\boldsymbol{R}_k^{0\mathsf{T}}\mathbf{e}_i) - 2\boldsymbol{\omega}_k^\mathsf{T}(\sum_i m_i\mathbf{Q}_i^{0\mathsf{T}}\mathbf{e}_i) + \sum_i m_i\|\mathbf{e}_i\|^2 \tag{19d}$$

From Eq. 19c, we can see that the objective is a quadratic function of new unknown $\boldsymbol{\omega}$ and $\mathbf{l}$. So we can expand Eq. 19c and extract $\boldsymbol{\omega}$ and $\mathbf{l}$, which leads to 19d. Eq. 15a is formed by summing Eq. 19 over $k = 1\dots m$. Each individual bone contributes a 6×6 block, visualized in Figure 2.

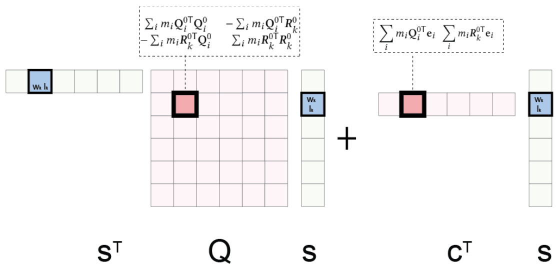
$\sum_i m_i\mathbf{Q}_i^{0\mathsf{T}}\mathbf{Q}_i^0 \quad -\sum_i m_i\mathbf{Q}_i^{0\mathsf{T}}\boldsymbol{R}_k^0$
$-\sum_i m_i\boldsymbol{R}_k^{0\mathsf{T}}\mathbf{Q}_i^0 \quad \sum_i m_i\boldsymbol{R}_k^{0\mathsf{T}}\boldsymbol{R}_k^0$
$\sum_i m_i\mathbf{Q}_i^{0\mathsf{T}}\mathbf{e}_i \quad \sum_i m_i\boldsymbol{R}_k^{0\mathsf{T}}\mathbf{e}_i$
Wk lk
+
sT
Q
s
cT
s

Figure 2: Block structure of Q and c

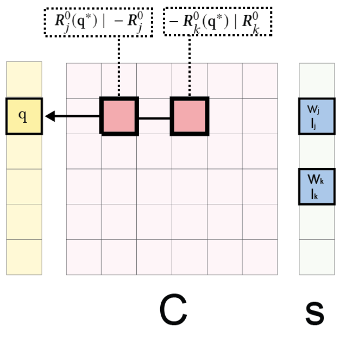
$\boldsymbol{R}_j^0(\mathbf{q}^*) \mid -\boldsymbol{R}_j^0$
$-\boldsymbol{R}_k^0(\mathbf{q}^*) \mid \boldsymbol{R}_k^0$
q
Wj lj
Wk lk
C
s

Figure 3: Block structure of C. The joint positions are shown in the left yellow column and the bones in the blue column along the right side. Each joint/bone pair corresponds to a 3×6 block in the matrix. A block is non-zero only if the bone ties to the joint. Here, we show a joint connecting two bones, such as the elbow joint.

We summarize our final algorithm in Alg. 1. The optimization problem in Eq. 15 amounts to a small linear system solve ($6m \times 6m$ where m is the number of bones), but we need to iterate (lines 5-12) because the $SE(3)$ manifold is non-linear. Note that our solver enforces rigidity constraints exactly within every iteration, and the joint error (i.e. the residual of Eq. 13c) typically converges to zero very quickly, as shown in Figure 8 in Section 5.

4 IMPLEMENTATION

The implementation of our method starts with constructing a conforming mesh of the rigid(bone) and deformable(flesh) bodies. We use one .obj file to define the closed surface of the exterior and anther .obj file to define the closed surface of the interior skeleton. Our framework takes the two .obj files as input and outputs a .smesh file which tetgen[Si 2015] can turn into conforming (Delaunay) tetrahedralized mesh with a different regional attribute for each enclosed region. From this .mesh file, we know exactly which vertices belong to the which bone region and which vertices belong to the flesh region. In addition, we created a .joint file specifically

ALGORITHM 1: Monolithic Solver for Constrained Dynamics

```
1  Init: $\hat{\mathbf{x}} := \begin{bmatrix}\mathbf{x}_{f0}\\ \mathbf{T}_0\end{bmatrix}$
2  for $i = 1, \ldots, n_{local/global\ iterations}$ do
3      Local step: fix $\hat{\mathbf{x}}$, compute $\mathbf{p}$;
4      Global step: fix $\mathbf{p}$, solve Eq. 12 for $\hat{\mathbf{x}}$;
5      for $j = 1, \ldots, n_{constraint\ iterations}$ do
6          For each bone $k$, compute $\begin{bmatrix}\mathbf{R}_k^0 & \mathbf{t}_k^0\end{bmatrix}$ (Procrustes);
7          if Residual of Eq. 13c is small then
8              Exit loop;
9          Iterate over bones, construct $\mathbf{Q}$ and $\mathbf{c}$;
10         Iterate over joints, construct $\mathbf{C}$ and $\mathbf{r}$;
11         Compute $\mathbf{s}$ by solving Eq. 15 (linear system);
12         Iterate over bones, update $\mathbf{T_k}$ using Eq. 14;
13     Update: $\hat{\mathbf{x}} := \begin{bmatrix}\mathbf{x}_f\\ \mathbf{T}\end{bmatrix}$
```

in our framework to keep track of the affiliation of all the rigid bones and joints. By combining the .joint and .mesh file, we have a complete model of the soft articulated character. This process of constructing the model of soft articulated character is illustrated in Figure 4. We show the model of a bar, a mermaid and a human in Figure 5, where the deformable parts and the rigid parts(bone) are colored pink and white respectively. The red dots represent the joint positions. The deformable parts are simulated using a linear co-rotational model in all the examples shown in Section 5.

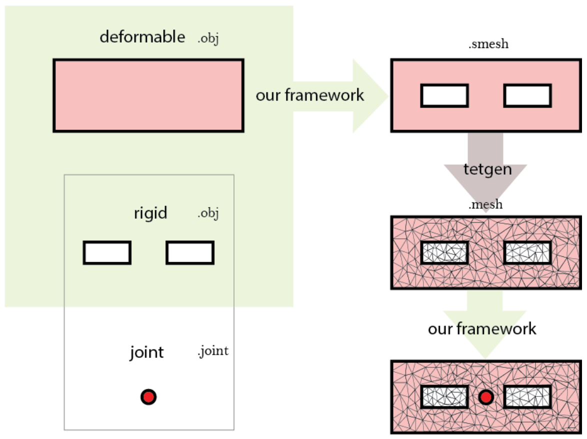

Figure 4: Schematic workflow of mesh construction

To handle contact, we used the collision method proposed in [Heidelberger 2007], which computes collision forces based on consistent n-body penetration depth information. We add the collision forces in $\mathbf{f}_{\text{ext}}$ in Eq. 1. We accelerate collision detection with spatial hashing [Heidelberger 2007] as well.

5 RESULTS

Table 1 summarizes all of our experiments including the run times of only using co-rotated linear elasticity. All of our simulations use 20 Projective Dynamics iterations ($n_{\text{local/global iterations}}$ in Alg. 1)

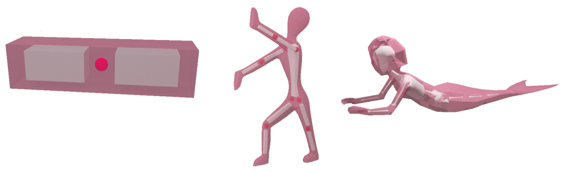

Figure 5: Visualization of the interior of our models. The deformable and rigid components are colored using pink and white respectively. The joints are illustrated using red dots.

and a time step of 0.033s. In this table, *Affine Time* is the cost for the local and global solve in Section 3.1, *Joint Time* is the time spent for joint constraints in Section 3.2. *Total Time* is the overall cost for our monolithic solver. *PD Time* is the reference time spent on Projective Dynamics as if there were no rigid or joint constraints at all. All timings were measured with an Intel®Core$^{\text{TM}}$ i7-8750H CPU.

We can see that the time spent on affine transformation is smaller or equivalent to Projective Dynamics, depending on how large the ratio between the number of rigid/elastic vertices are. The larger the number of rigid vertices, the smaller the size of the linear system is, resulting in faster solver computation times. We set the max number iterations for the joint constraint ($n_{\text{joint iterations}}$ in Alg. 1) optimization to 100 (under 10 iterations is usually taken). Figure 8 shows the convergence of the lines 6-13 in Alg. 1. Specifically, this is a log-plot of the joint error $\sum_j \|\mathbf{T}_j\mathbf{q} - \mathbf{T}_k\mathbf{q}\|^2$ for two example joints, where k is the index of the bone which connects to a joint and j is the indices of all the other bones which connect to the same joint.

In contrast, Projective Dynamics might fail to converge when coupling of rigid and deformable bodies is imitated by setting the stiffness of the two rigid parts higher than the deformable parts. As we can see in Figure 6 left, the rigid parts "lock" and fail to rotate. In the same scenario, our method quickly converges to the correct result (Figure 6). [Bouaziz et al. 2012] imposes the shape constraints by using penalty methods. The rigid constraints are not imposed exactly and controlled by the shape constraint weights. If the shape constraint weights are too high, it would still result in PD locking (Figure 7).

In Figure 9, we show the result of applying our method to a single rigid body (no deformable degrees of freedom) and compare to a state-of-the-art rigid body simulator [Bender 2007], which uses explicit RK4 integration. When both simulators use timestep 0.01, we can see small differences after simulating 3.0 seconds of motion. These differences can be explained by the numerical damping of our implicit integrator (backward Euler). Indeed, when we reduce the timestep to 0.001 in both simulators and simulate the same motion again, we obtain nearly identical results (Figure 9 right).

In Figure 10, we show a snail falling down under gravity and colliding with a sphere. In this example, there's only one rigid body and no joint constraints. We simply remove the Eq. 13c from Eq. 13 to obtain only the rigid body constraint. Almost half of the vertices of the whole system are rigid in this example, the time spent on solving the affine transformation is significantly lower

Table 1: Results on our example models.

Example	#Elastic Verts	#Verts	#Elems	#Bones	#Joints	Affine Time	Joint Time	Total Time	PD Time
`bar`	1010	1690	7661	2	1	59ms	17ms	76ms	56ms
`snail`	14318	28840	105694	1	0	998ms	16ms(rigid)	1014ms	1159ms
`washing machine`	11617	12683	45888	9	8	413ms	108ms	521ms	431ms
`aerial silk`	11617	12683	45888	9	8	407ms	262ms	669ms	439ms
`mermaid`	5249	7570	32270	9	0	276ms	8ms(rigid)	284ms	273ms

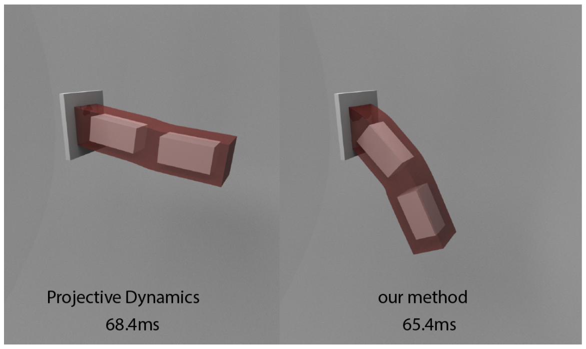

Figure 6: Imitating rigid bodies by higher stiffness in Projective Dynamics results in locking (left), Our method (right) produces the correct result.

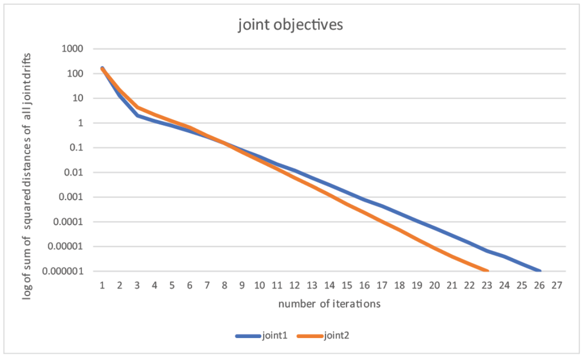

Figure 8: Reduction of joint error (i.e., the residual of Eq. 13c) with respect to the number of iterations of our generalized Procrustes solver (lines 5-12 in Algorithm 1).

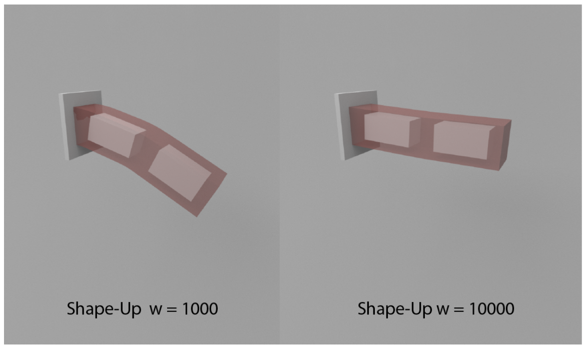

Figure 7: Imitating rigid bodies by shape constraints in [Bouaziz et al. 2012]. Higher weight of shape constraints still results in PD locking (right).

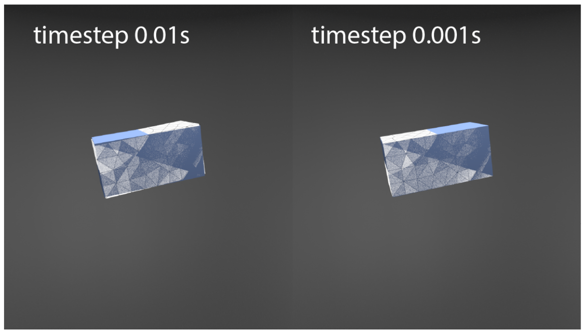

Figure 9: The blue bar is simulated by [Bender 2007], a library for dynamic simulation of multi-body systems. The white bar is simulated using our methods with all vertices rigidly constrained. The wireframe in the white bar shows our mesh, even though we use only 12 degrees of freedom. With time step 0.01s, there's a small deviation between the blue bar and the white bar. With time step 0.001s, the deviation between the blue bar and the white bar is significantly reduced.

than simulating the whole system as a deformable body using Projective Dynamics.

In Figure 11, we show that our method with joint constraints produces more realistic animation by tying the rigid bones to joints. The character is doing aerial silk performance like an acrobat, producing global deformation that skinning methods such as [Galoppo et al. 2007b] cannot produce. Note that there is no skull in the character as shown in Figure 5.

In Figure 12, the tail of the mermaid is pre-bent. During the animation, the tail moves as it tries to restore its rest-pose shape. And initial velocity moves the mermaid towards the cylindrical glass. The hands and arms collide with the glass, thus making the hands and arms move around.

6 LIMITATION AND FUTURE WORK

The joint constraints prevent bones from separating, but each bone can still rotate arbitrarily far, which is not true for most biological

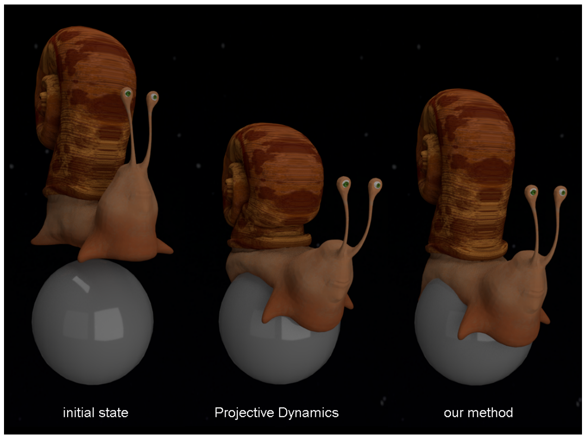

Figure 10: A snail collides with a sphere. Only highly deformable shell can be efficiently simulated with Projective Dynamics (middle). With our method (right), the shell stays rigid while the body elastically wraps around the sphere.

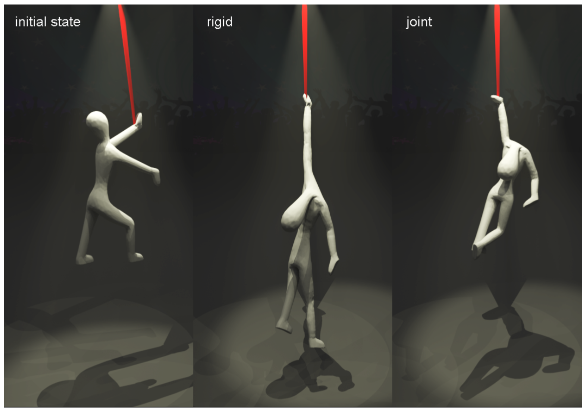

Figure 11: Simulation of an acrobat doing an aerial silk performance. The middle figure shows our method with rigid constraints where the bones are rigid bot not constrained to stay connected, resulting in unrealistic elongation of the character. Our method with joint constraints holds the bones together, resulting in a more realistic skeleton-like motion.

joints. An interesting avenue for future work would involve adding the support of joint limits. In all our examples, the characters are passive, i.e., not actuated. It would be possible to extend our method with artist directed control to our physics-based system. Spacetime constraints [Witkin and Kass 1988] offer control over the animation and add physical realism with desired secondary motion effects. State-of-the-art character control using spacetime constraints supports only articulated bodies [Witkin and Kass 1988] or purely deformable bodies [Schulz et al. 2014]. In the future it would be interesting to extend spacetime constraints to characters with soft flesh and rigid skeletons.

7 CONCLUSION

We introduced a new monolithic method to simulate articulated soft characters based on constrained dynamics. Our method has comparable computing performance to Projective Dynamics and is simple to implement; in particular, no additional code is required to explicitly handle the coupling between rigid and deformable bodies. Our method is suitable for simulating characters with deformable bodies, rigid bones and joints, which is often required for realistic creatures. We believe our new articulated/deformable simulating method will find use in computer games or training simulators.

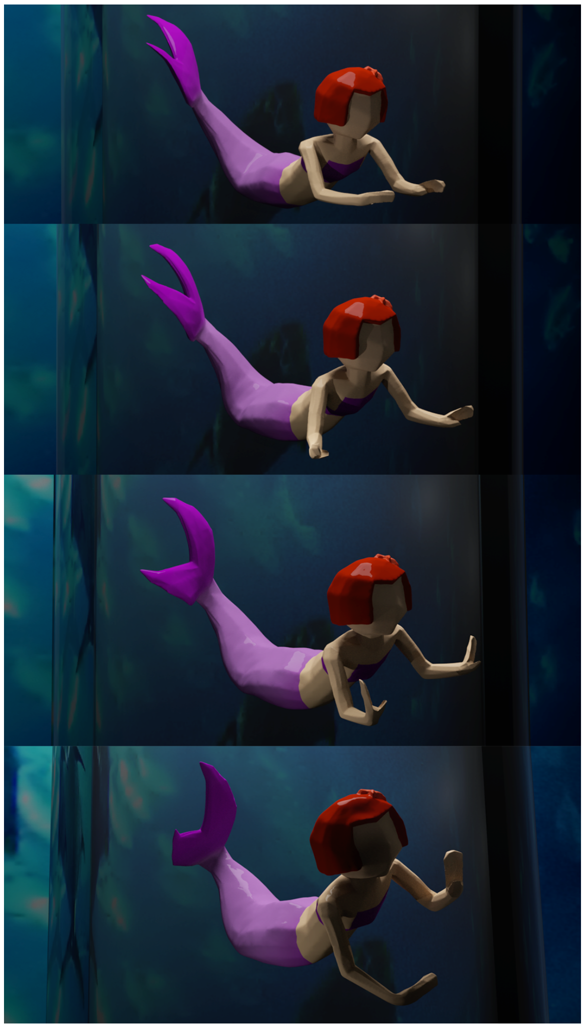

Figure 12: Results of a mermaid colliding with a cylindrical glass container. The tail is elastic and we can contrast its deformable nature to the relatively more rigid upper body which contains bones.

ACKNOWLEDGMENTS

We thank for Eftychios Sifakis many inspiring discussions. We also thank Pia Krensel for mermaid modelling, and Henry Rietra for snail modelling, Dimitar Dinev, Wenxian Guo for proofreading, Dimitar Dinev for narrating the accompanying video. This material is based upon work supported by the National Science Foundation under Grant Numbers IIS-1617172, IIS-1622360 and IIS-1764071. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation. We also gratefully acknowledge the support of Activision, Adobe, and hardware donation from NVIDIA Corporation.

REFERENCES

Sheldon Andrews, Marek Teichmann, and Paul G. Kry. 2017. Geometric Stiffness for Real-time Constrained Multibody Dynamics. *Computer Graphics Forum* 36, 2 (2017). https://doi.org/10.1111/cgf.13122

William W Armstrong and Mark W Green. 1985. The dynamics of articulated rigid bodies for purposes of animation. *The visual computer* 1, 4 (1985), 231–240.

David Baraff. 1996. Linear-time dynamics using Lagrange multipliers. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*. ACM, 137–146.

David Baraff and Andrew Witkin. 1997. *Partitioned dynamics*. Technical Report. CARNEGIE-MELLON UNIV PITTSBURGH PA ROBOTICS INST.

Jan Bender. 2007. IBDS: Impulse Based Dynamic Simulation. http://interactive-graphics.de/index.php/downloads/12-ibds.

Jan Bender, Kenny Erleben, and Jeff Trinkle. 2014. Interactive simulation of rigid body dynamics in computer graphics. In *Computer Graphics Forum*, Vol. 33. Wiley Online Library, 246–270.

Sofien Bouaziz, Mario Deuss, Yuliy Schwartzburg, Thibaut Weise, and Mark Pauly. 2012. Shape-up: Shaping discrete geometry with projections. In *Computer Graphics Forum*, Vol. 31. Wiley Online Library, 1657–1667.

Sofien Bouaziz, Sebastian Martin, Tiantian Liu, Ladislav Kavan, and Mark Pauly. 2014. Projective dynamics: fusing constraint projections for fast simulation. *TOG* 33, 4 (2014), 154.

Christopher Brandt, Elmar Eisemann, and Klaus Hildebrandt. 2018. Hyper-reduced Projective Dynamics. *ACM Trans. Graph.* 37, 4, Article 80 (2018), 80:1–80:13 pages.

Steve Capell, Matthew Burkhart, Brian Curless, Tom Duchamp, and Zoran Popović. 2005. Physically based rigging for deformable characters. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation*. ACM, 301–310.

Yan Chen, Qing hong Zhu, Arie E. Kaufman, and Shigeru Muraki. 1998. Physically-based Animation of Volumetric Objects. In *CA*.

Gilles Debunne, Mathieu Desbrun, Marie-Paule Cani, and Alan H Barr. 2001. Dynamic real-time deformations using space & time adaptive sampling. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*. ACM, 31–36.

Roy Featherstone. 1987. *Robot Dynamics Algorithm*. Kluwer Academic Publishers, Norwell, MA, USA.

Marco Fratarcangeli, Valentina Tibaldo, and Fabio Pellacini. 2016. Vivace: A practical gauss-seidel method for stable soft body dynamics. *ACM Transactions on Graphics (TOG)* 35, 6 (2016), 214.

Nico Galoppo, Miguel A Otaduy, Serhat Tekin, Markus Gross, and Ming C Lin. 2007a. Soft articulated characters with fast contact handling. In *Computer Graphics Forum*, Vol. 26. Wiley Online Library, 243–253.

Nico Galoppo, Miguel A. Otaduy, Serhat Tekin, Markus Gross, and Ming C. Lin. 2007b. Soft Articulated Characters with Fast Contact Handling. *Computer Graphics Forum* 26, 3 (2007), 243–253.

Fabian Hahn, Sebastian Martin, Bernhard Thomaszewski, Robert Sumner, Stelian Coros, and Markus Gross. 2012. Rig-space physics. *ACM transactions on graphics (TOG)* 31, 4 (2012), 72.

Bruno Heinz Heidelberger. 2007. *Consistent collision and self-collision handling for deformable objects*. Eidgenössische Technische Hochschule [ETH] Zürich.

Doug L James and Dinesh K Pai. 1999. ArtDefo: accurate real time deformable objects. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*. ACM Press/Addison-Wesley Publishing Co., 65–72.

Johan Jansson and Joris SM Vergeest. 2003. Combining deformable-and rigid-body mechanics simulation. *The Visual Computer* 19, 5 (2003), 280–290.

Ladislav Kavan and Olga Sorkine. 2012. Elasticity-inspired deformers for character articulation. *ACM Transactions on Graphics (TOG)* 31, 6 (2012), 196.

Junggon Kim and Nancy S Pollard. 2011. Fast simulation of skeleton-driven deformable body characters. *ACM Transactions on Graphics (TOG)* 30, 5 (2011), 121.

Tassilo Kugelstadt, Dan Koschier, and Jan Bender. 2018. Fast Corotated FEM using Operator Splitting. *Computer Graphics Forum* 37 (12 2018), 149–160. https://doi.org/10.1111/cgf.13520

C. Lanczos. 1986. *The Variational Principles of Mechanics*. Dover Publications. https://books.google.com/books?id=ZWoYYr8wk2IC

Julien Lenoir and Sylvere Fonteneau. 2004. Mixing deformable and rigid-body mechanics simulation. In *Proceedings Computer Graphics International, 2004*. IEEE, 327–334.

Libin Liu, KangKang Yin, Bin Wang, and Baining Guo. 2013. Simulation and control of skeleton-driven soft body characters. *ACM Transactions on Graphics (TOG)* 32, 6 (2013), 215.

Tiantian Liu, Sofien Bouaziz, and Ladislav Kavan. 2017. Quasi-newton methods for real-time simulation of hyperelastic materials. *ACM Transactions on Graphics (TOG)* 36, 4 (2017), 116a.

John E Lloyd, Ian Stavness, and Sidney Fels. 2012. ArtiSynth: A fast interactive biomechanical modeling toolkit combining multibody and finite element simulation. In *Soft tissue biomechanical modeling for computer assisted surgery*. Springer, 355–394.

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*. ACM, 49–54.

Miles Macklin, Matthias Müller, Nuttapong Chentanez, and Tae-Yong Kim. 2014. Unified particle physics for real-time applications. *ACM Transactions on Graphics (TOG)* 33, 4 (2014), 153.

Sebastian Martin, Bernhard Thomaszewski, Eitan Grinspun, and Markus Gross. 2011. Example-based elastic materials. In *ACM Transactions on Graphics (TOG)*, Vol. 30. ACM, 72.

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 Transactions on Graphics (TOG)* 30, 4 (2011), 37.

Matthias Müller, Julie Dorsey, Leonard McMillan, Robert Jagnow, and Barbara Cutler. 2002. Stable real-time deformations. In *Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation*. ACM, 49–54.

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, Bruno Heidelberger, Matthias Teschner, and Markus Gross. 2005. Meshless deformations based on shape matching. In *ACM transactions on graphics (TOG)*, Vol. 24. ACM, 471–478.

Rahul Narain, Matthew Overby, and George E. Brown. 2016. ADMM $\supseteq$ Projective Dynamics: Fast Simulation of General Constitutive Models. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA '16)*. Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 21–28. http://dl.acm.org/citation.cfm?id=2982818.2982822

James F O'Brien, Victor B Zordan, and Jessica K Hodgins. 2000. Combining active and passive simulations for secondary motion. *IEEE Computer Graphics and Applications* 20, 4 (2000), 86–96.

Yue Peng, Bailin Deng, Juyong Zhang, Fanyu Geng, Wenjie Qin, and Ligang Liu. 2018. Anderson acceleration for geometry optimization and physics simulation. *ACM Transactions on Graphics (TOG)* 37, 4 (2018), 42.

Christian Schulz, Christoph von Tycowicz, Hans-Peter Seidel, and Klaus Hildebrandt. 2014. Animating Deformable Objects Using Sparse Spacetime Constraints. *ACM Trans. Graph.* 33, 4, Article 109 (July 2014), 10 pages. https://doi.org/10.1145/2601097.2601156

Ahmed A. Shabana. 2005. *Dynamics of Multibody Systems* (3 ed.). Cambridge University Press. https://doi.org/10.1017/CBO9780511610523

Tamar Shinar. 2008. *Simulation of Coupled Rigid and Deformable Solids and Multiphase Fluids*. Ph.D. Dissertation. Stanford, CA, USA. Advisor(s) Fedkiw, Ronald. AAI3313660.

Tamar Shinar, Craig Schroeder, and Ronald Fedkiw. 2008. Two-way Coupling of Rigid and Deformable Bodies. In *Proceedings of the 2008 ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA '08)*. Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 95–103. http://dl.acm.org/citation.cfm?id=1632592.1632607

Hang Si. 2015. TetGen, a Delaunay-based quality tetrahedral mesh generator. *ACM Transactions on Mathematical Software (TOMS)* 41, 2 (2015), 11.

Eftychios Sifakis, Tamar Shinar, Geoffrey Irving, and Ronald Fedkiw. 2007. Hybrid simulation of deformable solids. In *Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation*. Eurographics Association, 81–90.

Carlota Soler, Tobias Martin, and Olga Sorkine-Hornung. 2018. Cosserat rods with projective dynamics. *Computer Graphics Forum (proceedings of SCA issue)* 37, 8 (2018).

Olga Sorkine-Hornung and Michael Rabinovich. 2017. Least-squares rigid motion using svd. *no* 3 (2017), 1–5.

Joseph Teran, Sylvia Blemker, V Hing, and Ronald Fedkiw. 2003. Finite volume methods for the simulation of skeletal muscle. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*. Eurographics Association, 68–74.

Demetri Terzopoulos, John Platt, Alan Barr, and Kurt Fleischer. 1987. Elastically deformable models. *ACM Siggraph Computer Graphics* 21, 4 (1987), 205–214.

Maxime Tournier, Matthieu Nesme, Benjamin Gilles, and Francois Faure. 2015. Stable constrained dynamics. *ACM Transactions on Graphics (TOG)* 34, 4 (2015), 132.

Mickeal Verschoor, Daniel Lobo, and Miguel A. Otaduy. 2018. Soft-Hand Simulation for Smooth and Robust Natural Interaction. http://gmrv.es/Publications/2018/VLO18

Huamin Wang. 2015. A chebyshev semi-iterative approach for accelerating projective and position-based dynamics. *ACM Transactions on Graphics (TOG)* 34, 6 (2015), 246.

Rachel Weinstein, Joseph Teran, and Ron Fedkiw. 2006. Dynamic Simulation of Articulated Rigid Bodies with Contact and Collision. *IEEE Transactions on Visualization and Computer Graphics* 12, 3 (May 2006), 365–374. https://doi.org/10.1109/TVCG.2006.48

Rachel Lara Weinstein. 2007. *Simulation and Control of Articulated Rigid Bodies.* Ph.D. Dissertation. STANFORD UNIVERSITY.

Andrew Witkin and Michael Kass. 1988. Spacetime Constraints. In *Proceedings of the 15th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '88)*. ACM, New York, NY, USA, 159–168. https://doi.org/10.1145/54852.378507

Victor Brian Zordan, Anna Majkowska, Bill Chiu, and Matthew Fast. 2005. Dynamic response for motion capture animation. In *ACM Transactions on Graphics (TOG)*, Vol. 24. ACM, 697–701.