Pacific Graphics 2020
E. Eisemann, A. Jacobson, and F.-L Zhang
(Guest Editors)

Volume 39 (2020), Number 7

Adjustable Constrained Soft-Tissue Dynamics

B. Wang[†], M. Zheng[‡], J. Barbič[§]

University of Southern California, United States

Abstract

Physically based simulation is often combined with geometric mesh animation to add realistic soft-body dynamics to virtual characters. This is commonly done using constraint-based simulation whereby a soft-tissue simulation is constrained to geometric animation of a subpart (or otherwise proxy representation) of the character. We observe that standard constraint-based simulation suffers from an important flaw that limits the expressiveness of soft-body dynamics. Namely, under correct physics, the frequency and amplitude of soft-tissue dynamics arising from constraints ("inertial amplitude") are coupled, and cannot be adjusted independently merely by adjusting the material properties of the model. This means that the space of physically based simulations is inherently limited and cannot capture all effects typically expected by computer animators. For example, animators need the ability to adjust the frequency, inertial amplitude, gravity sag and damping properties of the virtual character, independently from each other, as these are the primary visual characteristics of the soft-tissue dynamics. We demonstrate that independence can be achieved by transforming the equations of motion into a non-inertial reference coordinate frame, then scaling the resulting inertial forces, and then converting the equations of motion back to the inertial frame. Such scaling of inertia makes it possible for the animator to set the character's inertial amplitude independently from frequency. We also provide exact controls for the amount of character's gravity sag, and the damping properties. In our examples, we use linear blend skinning and pose-space deformation for geometric mesh animation, and the Finite Element Method for soft-body constrained simulation; but our idea of scaling inertial forces is general and applicable to other animation and simulation methods. We demonstrate our technique on several character examples.

CCS Concepts
• ***Computing methodologies*** $\rightarrow$ ***Physical simulation;***

1. Introduction

Digital characters generally follow a three-stage process in computer animation practice today. First, a modeler creates the geometric shape of the character (typically a textured triangle mesh) using 3D shape modeling tools. Next, a rigger equips the mesh with an animation-ready structure that permits one to deform the mesh using some low-dimensional mechanism such as a skeleton or inverse kinematics (IK) handles; usually by means of skeleton skinning weights, blendshapes, or some other geometric deformer. Finally, the animator uses the rig to create the output mesh animation, by keyframing the skeleton joint angles or IK handles over time. Such a "geometric animation" process produces static mesh shapes in each character's pose, devoid of dynamics or any time history whatsoever.

Characters can be made much more realistic and appealing by adding secondary soft-tissue dynamics. Such dynamics can model soft tissue vibrations due to the character's acceleration, or collision response against external objects. Dynamics is too tedious to be animated by hand. A common approach to add dynamics is to "attach" a physically based simulation to the geometrically animated mesh shape (or a subset thereof), using constraint-based dynamics. Such a process is very common and widely used both in academic publications and in industry [CBC*05, HMT*12, HTC*13, XB16, LXB17, KDGI19, Tis13], to name a few methods. The soft-tissue is often modeled as a tetrahedral mesh, and simulated using mass-spring systems, or, as in our case, using Finite Element Method simulation. Animators generally like the output quality of such systems, as long as the system is reasonably fast and robust, and very importantly, controllable. Namely, the animator needs to be able to easily adjust salient deformable dynamics characteristics, such as the deformation amplitude, frequency, sagging under gravity and damping properties. Commonly, animators today adjust these quantities by tweaking the material properties of the FEM mesh.

In constraint-based simulation, character deformable dynamics occurs because of the motion of the constraints ("inertial dynamics") and because of contact; with inertial dynamics typically being the most visible part as it often occurs everywhere on the character

[†] joint first author; bohanwan@usc.edu
[‡] joint first author; mianlunz@usc.edu
[§] jnb@usc.edu

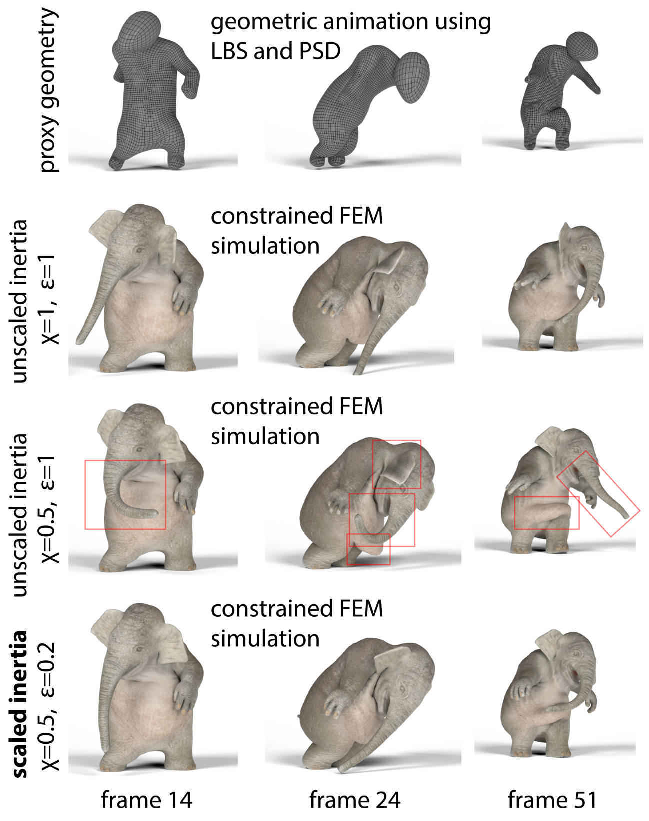
proxy geometry
geometric animation using LBS and PSD
unscaled inertia χ=1, ε=1
constrained FEM simulation
unscaled inertia χ=0.5, ε=1
constrained FEM simulation
scaled inertia χ=0.5, ε=0.2
constrained FEM simulation
frame 14
frame 24
frame 51

Figure 1: ***Independent control of frequency and inertial amplitude.*** *Top row: mesh animation of the geometric proxy, created by a professional animator using linear blend skinning (LBS) and pose-space deformation (PSD). Second row: baseline simulation. Third row: material elastic properties have been adjusted to linearly scale the deformation frequency spectrum by 0.5x. Inertial amplitude automatically grows and cannot be adjusted independently of the frequency; if one changes material properties to make the object stiffer so as to lessen the amplitude, then the frequency changes also. Last row: our adjusted physics makes it possible to scale down inertial deformable dynamics to a level similar in the baseline simulation, while* not *changing the frequency spectrum. This is achieved by adding inertial force scaling terms to the equations of motion. As such, the system can be timestepped in the usual way, and we can apply standard contact resolution methods to robustly resolve self-contact. Parameters χ and ε denote the relative change of frequency and scaling of inertial forces (unmodified motion: $\chi = \varepsilon = 1$), respectively.*

and across the entire motion timeline. In this paper, we demonstrate that the process of tweaking material properties suffers from an important previously unidentified practical limitation: by following the laws of physics, it is impossible to simultaneously adjust the inertial deformation amplitude and frequency merely by tweaking the physical model. We give a method that avoids this limitation, by observing that any dynamical system can be expressed in an arbitrary reference coordinate frame, including frames that accelerate ("non-inertial frames"), as long as proper resulting inertial forces are added to the physical system (also called fictitious forces, or d'Alembert forces). Our key idea is to transform the model into a properly selected non-inertial coordinate frame, compute the correct inertial forces, *but then scale them*; and finally transform back to the world-coordinate inertial frame. Such a transformation gives direct control over the inertial deformation amplitude of the character's soft tissue dynamics (Figures 1, 2). The resulting deformable dynamics is non-physical, but contains arbitrary and adjustable (deflated, exaggerated, etc.) soft-tissue deformation amplitudes as often needed and used in character animation. We then derive equations for how to adjust not just the deformation amplitude of the soft tissue of the character, but also its frequency, gravity sag and damping decay rate, by adjusting both the material properties and the inertial force scale. In summary, our work contributes the ability to adjust the most salient visual characteristics of the character's deformable dynamics without "guessing" the model physical properties, and with the flexibility to independently adjust frequency and amplitude. Our method modifies the standard constraint-based dynamics equations of motion in a minimal way, merely by adding additional terms to the right-hand side. Our method can therefore be used with any existing numerical integrator, contact resolution scheme and easily supports both homogeneous and inhomogeneous material properties without any special treatment: our examples have artist-adjusted spatially varying material properties in their ears, trunks, leaves and other similar parts.

2. Related work

Bringing physically realistic deformations to geometric animation methods is a widely studied topic in computer animation. There are many methods to add physics to skeleton rigs. In [CBC*05], characters were animated using force-driven rigs, with the forces described using the rig building blocks. One can also construct a deformation basis from a skeleton embedded into a tetrahedral mesh, and combine it with domain-decomposition to simulate articulated deformable characters [KJ12]. Piovarči et al. created a physically-inspired stretching model to evaluate the stretching caused by the gravitational force or muscle contractions [PMĎ15]. Liu et al. coupled skeleton and soft-tissue dynamics and employed a novel pose-based plasticity model for human-like skeleton-driven soft body characters [LYWG13]. Galoppo et al. handled large-area contacts on a skeleton-driven surface mesh, producing rich soft-tissue deformations [GOT*07]. Shi et al. learned secondary deformations from a few example animation sequences of the surface mesh, and applied it to skeleton-driven animations [SZT*08]. Gilles et al. combined frame-based skinning models with physically based deformable object simulation [GBFP11], and Martin et al. biased physically based simulation to example shapes [MTGG11]. The fast physically-based simulation system in [KP11] simulates two-way interaction between the skeleton, the deformable body, and the environment. McAdams et al. presented an algorithm for soft-tissue FEM deformation for skeleton-driven characters, based on a novel discretization of co-rotational elasticity over a hexahedral lattice [MZS*11]. Malgat et al. gave a layering approach to add simulation detail where needed by artists [MGL*15]. In [HMT*12, HTC*13], Hahn et al. formulated the soft-body FEM equations of motions in the rig-space. The method in [XB16] enriched the rigged character animations with secondary model-reduced soft-tissue Finite Element Method (FEM) dynamics, and optionally incorporated pose-space deformation (PSD) [LCF00]. Li et al. presented a system to combine arbitrary triangle mesh animations with physically based Finite Element Method (FEM) simulation, enabling control both in space and time [LXB17]. Researchers also added physics to facial animation, by forming facial blendshapes as a basis of forces [BSC16], using the original animation as per-frame rest poses [KBB*17], or augmenting real-time performance capture with physics [BS19]. Researchers at Pixar, focusing on providing state-of-the-art nonlinear elasticity models (we use their neo-Hookean material [SGK18] in our work), also demonstrated how to effectively use constraints to add secondary soft-tissue dynamics [SGK18, SGK19, KDGI19].

Previous work has therefore successfully investigated how to enrich geometric/kinematic animation with more compelling physical effects. However, physics is generally difficult to tune for non-experts, as the typical physical parameters such as, for example, Young's modulus or acceleration of gravity, do not necessarily easily translate into good intuition for visual output. In the view of an animation artist, it is very important to not just add physics to geometric/kinematic animation, but also directly control its key visual characteristics. We provide such a method, focusing on key visual characteristics such as inertial amplitude, frequency, gravity sag and damping. Previous work did not address this problem directly; and for a good reason, as it is (under correct physics) impossible to independently adjust the two key visual characteristics, frequency and inertial amplitude. Actually, to the best of our knowledge, this hindering coupling of frequency and amplitude has not been previously identified. The work of [WZB17] investigated a related problem for model-reduced systems combined with domain decomposition for botanical simulation, but did not address virtual characters, unreduced physical systems or systems that are *not* decomposed into domains. Xu et al. [XSZB15] controlled deformation amplitudes via nonlinear stress-strain material curves. However, doing so changes the actual nonlinear material, and as such modifies the static shapes also. Our method makes it possible to keep the static shapes as is, and only adjust the secondary dynamics. This facilitates animation design, as animations are often created in layers, and dynamics can be considered as an additional "layer" on top of static shapes. Furthermore, unlike our work, Xu et al. could not simultaneously control sag, and remained in the realm of what is possible under the laws of physics. In [LXB17], the authors combined physics with geometric animation using geometric rotation-aware interpolation. Their method cannot simulate correct physically based self-contact and character's interaction with other objects; we give a comparison in Figure 7.

Our work adjusts frequency using vibration theory [Sha90, DBC*15]. Previous work controlled natural frequencies of vibration using mesh coarsening [CLMK17, CLK*19]. This was done by adjusting frequencies of a coarse mesh to match those of a fine mesh; they did not discuss how to adjust inertial amplitude, sag or damping. A technique to adjust damping has been provided in [XB17]; however, the work did not discuss how to simultaneously adjust frequency, inertial amplitude and sag.

3. Coupling physics to geometric animation

We assume a standard dynamics-enabled computer animation pipeline whereby the animation system consists of two components: geometric animation, and physically based simulation. In geometric animation, one animates a point cloud using geometric techniques, such as using an embedded skeleton and linear blend skinning, using "sliders" and blendshapes, using pose-space deformation, or other nonlinear deformers, or any technique from geometric shape modeling (ARAP [SA07], BBW [JBPS11], variational methods [BK04], etc.). The point cloud may have connectivity (a triangle mesh), although this is not necessary in our method. Physically based simulation consists of a simulation mesh with material properties such as mass density, Young's modulus, Poisson's ratio, volume preservation; the specific list of parameters depends on the elastic material model. Most commonly, the simulation mesh is a tetrahedral mesh, although it could also be a triangle mesh for thin-shell simulation (cloth). Finally, the character's output triangle mesh is embedded into the simulation mesh, and driven using any suitable method such as barycentric weights, mean-value coordinates or other higher-order shape interpolation method.

The purpose of geometric animation is to provide the artist with easy and intuitive animation control. The animator can animate the character by specifying intuitive controls to geometric animation such as skeleton joint-angles, blendshape sliders, or 3d handle positions. The joint angles do not even need to be manually keyframe-animated, but can come from another process, such as motion capture or motion control. The purpose of physically based simulation is to provide soft-tissue dynamics and character collision response, as well as correct any "non-physical" behavior of geometric techniques, such as volume preservation violations, unwanted collisions or overly "robotic" motion. Physically based animation is coupled to geometric animation using constraints. Most practical systems employ one-way coupling, whereby physically based simulation is coupled to the output of the geometric shape, but the geometric stage is not affected by the physics; we adopt this approach in our work as well. We note that two-way coupling would entail the soft-tissue dynamics "pulling" on and modifying the geometric motion, e.g., modifying the character's angles as a result of elastic inertia or collisions. Although this is in principle better than one-way coupling, most systems in practice do not adopt this approach because it is (1) slower, and (2) makes it harder for the artists to control the animation.

In our method, the point cloud of the geometric stage may, or may not be, a subset of the output character's triangle mesh. The point cloud may be as simple as a few points undergoing translations, or general rigid body motion; or as complex as a fully rigged triangle mesh undergoing highly nonlinear shape deformation. In all cases, the point cloud serves to define the constraints for physically based simulation (Figure 3). In the sunflower example, for example, we want to animate the sunflower dynamic deformations resulting from holding it by the root and "yanking" it to the right. In the jellyfish example, we apply rigid body motion to a rigid jellyfish "core" point cloud (the "geometric proxy"), and then observe the highly dynamic detail of the soft head and tentacles.

We employ the Finite Element Method (FEM) for the character's

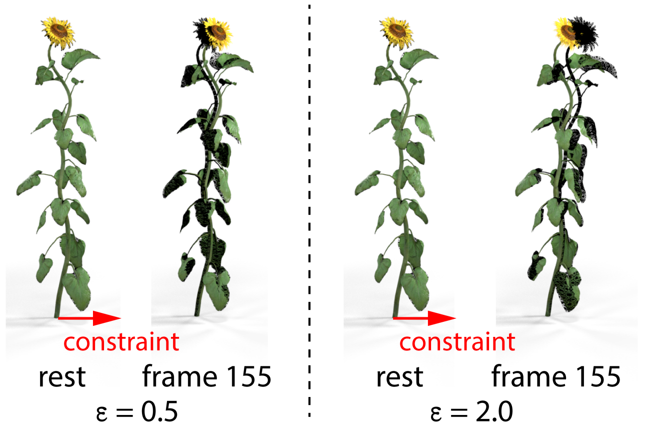

Figure 2: ***Adjusting inertial amplitude:*** *The geometric proxy here consists of four vertices at the very bottom of the sunflower; it is geometrically animated to abruptly start moving right with constant velocity, then abruptly stop at frame 150. The abrupt start and stop cause inertial dynamics (seen at frame 155) whereby the sunflower is "yanked" to the left and right, respectively, relative to its rigid configuration. Solid mesh shows the baseline motion ($\varepsilon = 1$) and wireframe shows motion with adjusted inertial amplitude (left: $\varepsilon = 0.5$, right: $\varepsilon = 2$). Observe that $\varepsilon = 0.5$ and $\varepsilon = 2.0$ generate less and more inertial dynamics compared to $\varepsilon = 1$, as expected. Linear sunflower. We show a nonlinear sunflower in the video.*

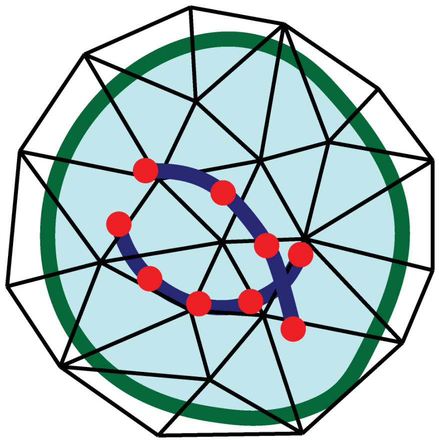

Figure 3: ***Constrained simulation overview.*** *The blue lines denote the geometrically animated proxy. While this is often a mesh (potentially with non-manifold topology), only its vertices (the "point cloud") are needed in our work; if more precise constraints are needed, the point cloud can be refined. The output character triangle mesh is denoted in green (also potentially with non-manifold topology); it is embedded into the tetrahedral mesh. The tetrahedral mesh can be constrained, for example, to the current positions of the red vertices, forming the constraint matrix C and right-hand-side vector $x(t)$. For example, one may constrain material tet mesh positions to the point cloud, using barycentric constraints.*

physically based soft-tissue dynamics. The equations of motion are

$$\mathbf{M}\ddot{\mathbf{u}} + \mathbf{D}\dot{\mathbf{u}} + \mathbf{f}_{int}(\mathbf{u}) = \mathbf{M}\mathbf{g}, \tag{1}$$

$$\text{subject to } \mathbf{C}\mathbf{u} = \mathbf{S}\mathbf{x}(t), \tag{2}$$

where $\mathbf{M} \in \mathbb{R}^{3n\times 3n}$ is the tet mesh mass matrix, n to be the number of tet mesh vertices, $\mathbf{f}_{int}(\mathbf{u})$ are the internal elastic forces, $\mathbf{D} = d_m\mathbf{M} + d_s\mathbf{K}$ is the Rayleigh's damping matrix, $\mathbf{g} \in \mathbb{R}^{3n}$ is the gravity acceleration and $\mathbf{u} \in \mathbb{R}^{3n}$, $\dot{\mathbf{u}} \in \mathbb{R}^{3n}$ and $\ddot{\mathbf{u}} \in \mathbb{R}^{3n}$ represent the displacements (away from the tet mesh rest shape), velocity and acceleration of the tet mesh vertices, respectively. The time-varying positions $\mathbf{x}(t)$ define the motion of the geometrically animated point cloud, $\mathbf{S}$ is a selection matrix that selects or combines the relevant degrees of freedom, and $\mathbf{C} \in \mathbb{R}^{m\times 3n}$ is the constraint matrix that constraints the tet mesh degrees of freedom to the point cloud. The internal elastic forces depend on the chosen elastic material properties, which may be selected in any suitable way, i.e., isotropic materials, anisotropic materials, homogeneous, non-homogeneous, linear, nonlinear, co-rotational, etc. Although we use the Finite Element Method and volumetric solid simulation, we note that other elastic models could be substituted instead without changing the structure of Equations 1 and 2, e.g., mass-spring systems, cloth simulation, elastic rod simulation, etc.

Obviously, different material properties produce different animation outputs. For example, larger Young's modulus stiffens the deformation. Therefore, it is necessary to provide the artist with a flexible, accurate and easy method to adjust the animations. A straightforward requirement is that this should be doable, as much as possible, without changing the geometric motion. Otherwise, the method loses flexibility and becomes much more burdensome and conceptually harder to understand, as even minor changes require re-executing even the geometric animation pipeline. In the next sections, we identify important desired visual properties for the animators to control, and then give a method to meet them in practice.

4. Animation control desiderata

We now identify four important visual properties of animations. In later sections, we will describe how to precisely set these properties, providing animators with useful high-level control over soft-body simulations.

Vibration frequency: Obviously, this is a primary visual characteristic of deformable dynamics and it is very useful for the animators to adjust it. Three-dimensional elastic objects are complex, and there is no such thing as a single vibration frequency. However, a reasonable representative choice is the lowest natural frequency of vibration of the object, under the boundary conditions imposed by the attachments to the geometric proxy point cloud. It can be computed by solving the generalized constrained eigenvalue problem [LSY97, XB16],

$$\mathbf{K}\mathbf{u} = \lambda\mathbf{M}\mathbf{u}, \tag{3}$$

$$\text{subject to } \mathbf{C}\mathbf{u} = \mathbf{S}\mathbf{x}. \tag{4}$$

where $\mathbf{K}$ is the stiffness matrix of the tetrahedral FEM mesh (in the rest shape), and the eigenvectors are $\mathbf{u}$ and the corresponding eigenvalues are λ. We solve the eigenproblem once, before start-

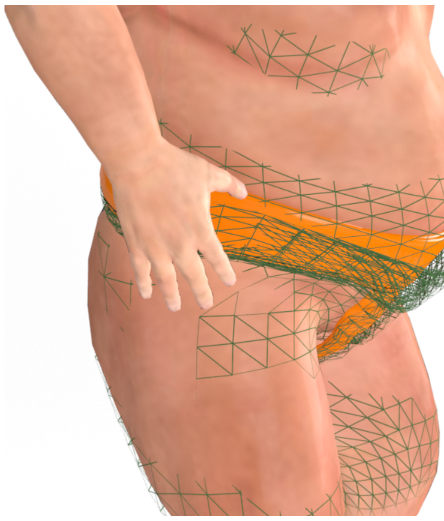

Figure 4: ***Sag control:*** *We used our technique (Section 5) to increase sag 3x (shown in wireframe) relative to original simulation (solid), while keeping frequency and deformation amplitude (as measured from sagged configuration) unmodified.*

ing the physically based modeling session, based on the character's rest shape and constraints to the point cloud positions $\mathbf{x}$ in the rest shape; and keep the smallest eigenvalue λ. The relationship between eigenvalue and frequency is $\lambda = \omega^2 = (2\pi)^2\nu^2$. Therefore, the character's natural frequency is $\nu = \sqrt{\lambda}/(2\pi)$, and the natural vibration period is $T = 1/\nu$. For simplicity and typically without much loss in practice, the above analysis ignores the changes in the lowest eigenvalue due to the changed boundary conditions during the animation. It also ignores higher vibration frequencies. However, we prefer to describe the vibration frequency with just a single number as this is easy to understand and adjust. When we adjust the vibration frequency ν in Section 5, we will do so in a way whereby *the entire frequency* spectrum linearly re-scales, i.e., the ratios between the frequencies remain constant. Such one-dimensional frequency control is easy to understand as it corresponds to making the dynamics "slower" or "faster".

Gravity sag: Sag represents how much the character's soft tissue "sinks downward" under its own weight. It is an important animation characteristic commonly discussed and adjusted in computer animation literature [TKA11]. The sag ζ under gravity can be computed by solving the static equilibrium equation $\mathbf{f}_{int}(\mathbf{u}) = \mathbf{Mg}$, and defining $\zeta = ||\mathbf{u}||$.

Inertial amplitude: Deformable dynamics originates from the motion of the geometric proxy point cloud. Recall the sunflower example in Figure 2 whereby the movement of the constrained vertices causes the sunflower to bend. Similarly, the geometric motion of the character proxy geometry causes bellies, thighs and cheeks to vibrate. If the proxy geometry moves with constant *velocity* (no rotations or accelerations), the soft tissue experiences no inertial forces and the amplitude is zero. In the presence of accelerations, however, such as during translational acceleration, rotations or non-rigid motion of the point cloud, there are inertial forces acting on the soft tissue, causing deformable dynamics. The goal of our paper is to provide the animator with intuitive control over the resulting deformation amplitude. As explained previously, with correct physics, the amplitude is coupled with frequency; in Section 5, we will give our approach to decouple them.

So, how can one define such a concept of amplitude? Our idea is to define amplitude as deviation away from the equilibrium configuration of the object, given the current constraints as imposed by the geometric point cloud proxy. Let p_i be the equilibrium displacement of tet mesh vertex i under the geometric point cloud constraints at the current timestep, i.e., the displacement of vertex i in a simulation that is completely devoid of any dynamics, such as those obtained using a static solve. Further, let R_i be the rotation matrix giving the local orientation of the material around vertex i relative to the rest shape orientation. We compute R_i as the volume-weighted average of the deformation gradients of tets around vertex i, followed by polar decomposition [MG04]. We can then express the total displacement from the elastic equilibrium as $u_i = p_i + R_i q_i$, where q_i is the "dynamic displacement" of the vertex i, expressed in its current local coordinate system. We define amplitude as the root-mean-square of dynamic displacements,

$$A = \sqrt{\frac{1}{m}\int_0^{t_{\max}} < \mathbf{Mq}, \mathbf{q} > dt}, \tag{5}$$

where we have assembled all vertex dynamic displacements into a vector $\mathbf{q}$, and where the inner-product is weighted with the mass matrix $\mathbf{M}$ to account for any non-uniform tet meshing, and the scalar m is the total object mass. In practice, we discretize the integral via the simulation timesteps.

Damping: We use the familiar Rayleigh damping in the form of $\mathbf{D} = d_m\mathbf{M} + d_k\mathbf{K}$, where d_m and d_k are the mass and stiffness damping coefficients. The modal damping factor [JP02] of the lowest natural frequency of vibration is

$$\xi = \frac{1}{2}\left(\frac{d_m}{\omega} + d_k\omega\right), \tag{6}$$

where $\omega = 2\pi\nu$ is the undamped lowest natural frequency of vibration. Much like with frequency, we are interested here in the lowest natural frequency, as opposed to the entire spectrum. We made this choice because this is often the dominant visual effect in animations, and the resulting one-dimensional analysis makes it tractable to tune the damping in a straightforward manner. The deformation amplitude decays with the envelope $\exp(-\xi\omega t)$ [JP02]. The salient visual characteristic of this decay is how long it takes for the amplitude to decay to a certain percentage of the original value, such as 50%:

$$e^{-\xi\omega t_D} = \frac{1}{2} \quad \Rightarrow \quad t_D = \frac{\log 2}{\xi\omega} = \frac{2\log 2}{d_m + 4\pi^2 d_k \nu^2}. \tag{7}$$

This relationship will permit us to tune d_m and d_k so that a given artist-chosen t_D is met (Section 5).

5. Meeting the desiderata

The first and "obvious" approach to meet the desiderata of Section 4 is to scale the various parameters of the physical model. One can scale mass, stiffness (Young's modulus) and gravity acceleration using scalars δ, γ, and μ,

$$\mathbf{M} \leftarrow \delta\mathbf{M}, \quad \mathbf{K} \leftarrow \gamma\mathbf{K}, \quad \mathbf{g} \leftarrow \mu\mathbf{g}. \tag{8}$$

In order to tune the damping factor, we scale the two damping coefficients with scalars c_m and c_k,

$$d_m \leftarrow c_m d_m, \quad d_k \leftarrow c_k d_k. \tag{9}$$

These scalings produce equations of motion

$$\delta\mathbf{M}\ddot{\mathbf{u}} + (c_m d_m \delta\mathbf{M} + c_k d_k \gamma\mathbf{K})\dot{\mathbf{u}} + \gamma\mathbf{f}_{int}(\mathbf{u}) = \mu\delta\mathbf{M}\mathbf{g}, \tag{10}$$

$$\text{subject to} \quad \mathbf{C}\mathbf{u} = \mathbf{S}\mathbf{x}(t). \tag{11}$$

We note that if one scales the time using a substitution $t = \kappa\tau$, for some constant $\kappa > 0$, one obtains a system of equations that have the same form as Equations 10 and 11, i.e, one does not introduce a fundamentally new equation with new capabilities. Although it at first seems that the scalings of Equations 10 and 11 should be sufficient to meet the desiderata, we discovered that the frequency and inertial amplitude are inherently coupled. For simplicity, we will now demonstrate this fact on a one-dimensional example. Suppose a 1D mass particle with mass m and position $x = x(t) \in \mathbb{R}$ is attached via a spring of stiffness k to a geometrically animated target $x_0 = x_0(t)$, i.e., the equations of motion are $m\ddot{x} + c(\dot{x} - \dot{x}_0) + k(x - x_0) = 0$. We are interested in the current displacement $z(t) = x(t) - x_0(t)$ from the static position $x_0(t)$. Suppose we start from rest $x(0) = \dot{x}(0) = x_0(0) = \dot{x}_0(0) = 0$, and suppose the animator animates x_0 so that it accelerates with a short acceleration pulse at $t = 0$ to velocity v_0. It can then be shown [JP02] that we have, for all $t \geq 0$,

$$z(t) = \frac{v_0}{\omega_d} exp\left(-\frac{c}{2m}t\right) sin(\omega_d t), \tag{12}$$

where $\omega_d = \sqrt{4mk - c^2}/2m$, and therefore, the amplitude at time t is $\frac{v_0}{\omega_d} exp\left(-\frac{c}{2m}t\right)$. This equation shows that as one tunes m, k, c to increase or decrease the oscillation frequency ω_d, the inertial amplitude decreases or increases, respectively, due to the presence of the $1/\omega_d$ term. One cannot simultaneously control frequency and amplitude merely by tweaking m, c and k. Due to linear superposition, a similar effect occurs with continuous non-instantaneous time-varying acceleration $\ddot{x}_0$, and also with general three-dimensional constrained dynamics governed by Equations 1 and 2. We now present our approach that enables the decoupling of frequency and inertial amplitude.

5.1. Scaling the inertial amplitude

In constraint-based simulation, soft-tissue dynamics originates from the constraints imposed by the motion of the geometric proxy; specifically, from the accelerations of its vertices. Intuitively, the greater the acceleration of the constraints, the greater the dynamic deformation amplitude. A first idea is to simply geometrically scale the geometric proxy and its motion, so as to lessen or boost the accelerations. In addition to complicating the formulation by maintaining artificially scaled objects, this also causes difficulties with interactions with the external world and other characters, as one cannot easily define and resolve collisions between two characters or the external world scaled by different factors.

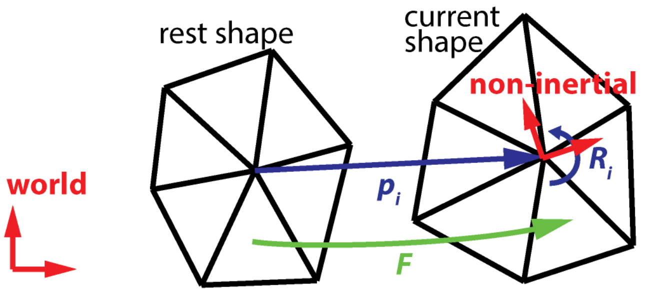

Figure 5: ***The non-inertial coordinate frame.*** *Rotation R_i is obtained by applying polar decomposition to the weighted deformation gradients (denoted by F) in the vertex 1-ring neighborhood.*

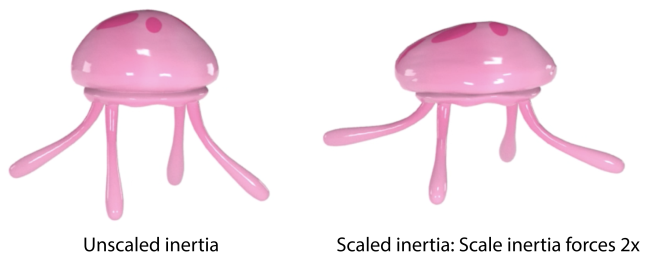

Figure 6: ***Scaling inertial forces:*** *The geometric proxy in this case is a point cloud in the "core" of this jellyfish. It was animated to spin with constant rotational velocity along the vertical axis. Left: correct physics. Right: we scale inertial forces 2x ("adjusted physics"); this causes the tentacles and base to deform more. The inertial forces in this example consist primarily of the centrifugal and Coriolis forces.*

Instead, our idea is to convert each vertex of the tet mesh into its own non-inertial reference frame (Figure 5). This causes an inertial force to each vertex, and our idea is to scale this force, either uniformly or by different amounts for different vertices for spatially varying amplitude control (as done in some of our jellyphant, sunflower and jellyfish examples). Finally, we then convert the equations of motion back to the inertial world-coordinate reference frame. As already previously explained in Section 4, we can express $\mathbf{u} = \mathbf{p} + \mathbf{R}\mathbf{q}$, where $\mathbf{p} = \mathbf{p}(t) \in \mathbb{R}^{3n}$ contains the static deformations p_i of all vertices at the current timestep, and $\mathbf{R} = \mathbf{R}(t) \in \mathbb{R}^{3n \times 3n}$ contains the 3x3 rotation matrices R_i down its diagonal. Equations 1 and 2 then become

$$\mathbf{M}\mathbf{R}\ddot{\mathbf{q}} + \mathbf{D}\mathbf{R}\dot{\mathbf{q}} + \mathbf{f}_{int}(\mathbf{p} + \mathbf{R}\mathbf{q}) =$$

$$\mathbf{M}\mathbf{g} - \mathbf{M}(\ddot{\mathbf{p}} + \ddot{\mathbf{R}}\mathbf{q} + 2\dot{\mathbf{R}}\dot{\mathbf{q}}) - \mathbf{D}(\dot{\mathbf{p}} + \dot{\mathbf{R}}\mathbf{q}), \tag{13}$$

$$\text{subject to } \mathbf{C}\mathbf{R}\mathbf{q} = \mathbf{S}\mathbf{x}(t) - \mathbf{C}\mathbf{p}. \tag{14}$$

Let $\varepsilon \geq 0$ be a scaling factor for the inertial forces $\mathbf{M}(\ddot{\mathbf{p}} + \ddot{\mathbf{R}}\mathbf{q} + 2\dot{\mathbf{R}}\dot{\mathbf{q}})$ in Equation 13, yielding

$$\mathbf{M}\mathbf{R}\ddot{\mathbf{q}} + \mathbf{D}\mathbf{R}\dot{\mathbf{q}} + \mathbf{f}_{int}(\mathbf{p} + \mathbf{R}\mathbf{q}) =$$

$$\mathbf{M}\mathbf{g} - \varepsilon\mathbf{M}(\ddot{\mathbf{p}} + \ddot{\mathbf{R}}\mathbf{q} + 2\dot{\mathbf{R}}\dot{\mathbf{q}}) - \mathbf{D}(\dot{\mathbf{p}} + \dot{\mathbf{R}}\mathbf{q}), \tag{15}$$

$$\text{subject to } \mathbf{C}\mathbf{R}\mathbf{q} = \mathbf{S}\mathbf{x}(t) - \mathbf{C}\mathbf{p}. \tag{16}$$

Finally, we substitute $\mathbf{q} = \mathbf{R}^T(\mathbf{u} - \mathbf{p})$, and obtain our "adjusted" physics equations, expressed in the inertial world-coordinate frame,

$$\mathbf{M}\ddot{\mathbf{u}} + \mathbf{D}\dot{\mathbf{u}} + \mathbf{f}_{int}(\mathbf{u}) = \mathbf{M}\mathbf{g} +$$
$$(1-\varepsilon)\mathbf{M}\Big(\ddot{\mathbf{p}} + (\ddot{\mathbf{R}}\mathbf{R}^T + 2\dot{\mathbf{R}}\dot{\mathbf{R}}^T)(\mathbf{u}-\mathbf{p}) + 2\dot{\mathbf{R}}\mathbf{R}^T(\dot{\mathbf{u}}-\dot{\mathbf{p}})\Big), \quad (17)$$
$$\text{subject to } \mathbf{C}\mathbf{u} = \mathbf{S}\mathbf{x}(t). \quad (18)$$

Observe that the terms $\mathbf{M}\ddot{\mathbf{p}}, \mathbf{M}\ddot{\mathbf{R}}\mathbf{R}^T(\mathbf{u}-\mathbf{p}), 2\mathbf{M}\dot{\mathbf{R}}\dot{\mathbf{R}}^T(\mathbf{u}-\mathbf{p}), 2\mathbf{M}\dot{\mathbf{R}}\mathbf{R}^T(\dot{\mathbf{u}}-\dot{\mathbf{p}})$ are the linear acceleration inertial force, angular acceleration inertial force, centrifugal force and Coriolis force, respectively. When $\varepsilon = 0$, the inertial motion caused by the constraints is canceled out, producing no deformable dynamics. When $\varepsilon = 1$, motion is unmodified, i.e., the inertial forces are not scaled and act as usual. We get stronger and weaker inertial forces (and therefore larger and smaller inertial amplitudes) by setting $\varepsilon > 1$ (Figure 6) and $\varepsilon < 1$, respectively. We compute the static displacements $\mathbf{P}$ using a static solver, at each simulation timestep, by solving

$$\mathbf{f}_{int}(\mathbf{p}) = \mathbf{M}\mathbf{g}, \quad (19)$$
$$\text{subject to } \mathbf{C}\mathbf{p} = \mathbf{S}\mathbf{x}(t). \quad (20)$$

In order to compute $\dot{\mathbf{p}}$ and $\ddot{\mathbf{p}}$, we use 5-point finite differences,

$$\ddot{\mathbf{p}}_i = \frac{-\mathbf{p}_{i+2} + 16\mathbf{p}_{i+1} - 30\mathbf{p}_i + 16\mathbf{p}_{i-1} - \mathbf{p}_{i-2}}{12h^2}. \quad (21)$$

In order to compute $\mathbf{R}$, $\dot{\mathbf{R}}$ and $\ddot{\mathbf{R}}$, we first compute deformation gradients of all tets. We then compute the deformation gradient at each vertex as a volume-weighted average of adjacent tets. The first and second time derivatives of these vertex deformation gradients can be easily computed analytically because the deformation gradient is linear in vertex displacements. We then compute the polar decomposition (yielding $\mathbf{R}$), as well as the first and second time derivatives of polar decomposition [BZ11], yielding $\dot{\mathbf{R}}$ and $\ddot{\mathbf{R}}$. We can now apply both the scalings of Equations 8 and 9 and the inertial force scalings, obtaining the final modified equations of motion,

$$\delta\mathbf{M}\ddot{\mathbf{u}} + (c_m d_m \delta\mathbf{M} + c_k d_k \gamma\mathbf{K})\dot{\mathbf{u}} + \gamma\mathbf{f}_{int}(\mathbf{u}) = \mu\delta\mathbf{M}\mathbf{g} +$$
$$(1-\varepsilon)\delta\mathbf{M}\Big(\ddot{\mathbf{p}} + (\ddot{\mathbf{R}}\mathbf{R}^T + 2\dot{\mathbf{R}}\dot{\mathbf{R}}^T)(\mathbf{u}-\mathbf{p}) + 2\dot{\mathbf{R}}\mathbf{R}^T(\dot{\mathbf{u}}-\dot{\mathbf{p}})\Big), \quad (22)$$
$$\text{subject to } \mathbf{C}\mathbf{u} = \mathbf{S}\mathbf{x}(t). \quad (23)$$

Observe that these equations are minimally perturbed from standard equations of motion 1 and 2; the only change is the presence of the ε term that scales inertial forces. As such, Equations 22 and 23 can be timestepped using the usual numerical integrators. We note that the inclusion of rotations $\mathbf{R}$ in the calculation of inertial forces is important. The rotations provide centrifugal, Coriolis and angular acceleration forces. In our jellyphant, sumo and jellyfish examples, these forces were often of similar magnitude as the translational acceleration inertia forces, and cannot be neglected without visual artifacts.

5.2. Meeting the animation control desiderata

We now describe how we meet the desiderata of Section 4. Before the adjustment, the system has a frequency ν, sag ζ, inertial amplitude A, and damping half-time t_d. Denote the desired after-scaling version of quantity x by $\bar{x}$. Absolute values of these quantities are typically not intuitive to the animators as they involve too much technical jargon, i.e., "make lowest frequency 3Hz". Instead, it is more intuitive for animators to provide input in relative terms, i.e., "make object vibrate 2x faster". Therefore, the animator prescribes ratios $\bar{\nu}/\nu$, $\bar{\zeta}/\zeta$, $\bar{A}/A$, $\bar{t_D}/t_D$. In order to derive the modified properties that meet the desiderata exactly, we use linear physics approximations. Although they are not exact under large deformations, they still provide meaningful guidance that can be used in an iterative design process.

Vibration frequency: The after-scaling eigenvalue problem is

$$\overline{\mathbf{K}}\bar{\mathbf{u}} = \bar{\lambda}\overline{\mathbf{M}}\bar{\mathbf{u}}, \quad (24)$$
$$\text{subject to } \mathbf{C}\bar{\mathbf{u}} = \mathbf{S}\mathbf{x}. \quad (25)$$

where $\overline{\mathbf{M}} = \delta\mathbf{M}$ and $\overline{\mathbf{K}} = \gamma\mathbf{K}$. The new lowest eigenvalue is $\bar{\lambda} = \frac{\gamma}{\delta}\lambda$. Consequently, the new lowest frequency is

$$\bar{\nu} = \frac{\sqrt{\bar{\lambda}}}{2\pi} = \sqrt{\frac{\gamma}{\delta}}\frac{\sqrt{\lambda}}{2\pi} = \sqrt{\frac{\gamma}{\delta}}\nu = \chi\nu, \quad \text{where } \chi = \sqrt{\frac{\gamma}{\delta}}. \quad (26)$$

It follows that we need to set $\chi = \bar{\nu}/\nu$. Note that once χ has been set, we can set γ and δ in any way to satisfy $\chi^2 = \gamma/\delta$. Factors γ and δ are never exposed individually in our work; and actually, this redundancy is a consequence of the fact that one can multiply both sides of the equations of motion with an arbitrary constant.

Sag amplitude after scaling can be computed as follows, which gives us a formula to set μ, given the value of χ determined above.

$$\bar{\zeta} = ||\overline{\mathbf{K}}^{-1}\overline{\mathbf{M}}\bar{\mathbf{g}}|| = \frac{\mu\delta}{\gamma}\zeta = \frac{\mu}{\chi^2}\zeta \quad \Rightarrow \quad \mu = \chi^2\frac{\bar{\zeta}}{\zeta}. \quad (27)$$

Damping: The after-scaling Rayleigh damping is $\bar{D} = c_m d_m \delta\mathbf{M} + c_k d_k \gamma\mathbf{K}$. In our system, the user controls the damping by prescribing a single parameter, namely the half-decay time t_D; hence, there is only one equation for two unknowns c_m and c_k. Stiffness damping is generally superior to mass damping and often preferred in simulation: it does not damp rigid body motion, and removes spurious high spatial frequencies. For this reason, we use only stiffness damping (i.e., we set $c_m = d_m = 0$). We then obtain

$$\bar{t}_D = \frac{\log 2}{2\pi^2 c_k d_k \bar{\nu}^2} = \frac{1}{c_k\chi^2}t_D \quad \Rightarrow \quad c_k = \frac{1}{\chi^2}\frac{t_D}{\bar{t_D}}. \quad (28)$$

Inertial amplitude is non-trivial to adjust because it depends not just on χ and ε, but also on the specific time profile of the constraints. To illustrate this, consider two extremes: (1) the geometric proxy undergoes constant and permanent acceleration, and (2) constraints just deliver a short acceleration impulse. In scenario (1), inertial amplitude can be solved using a static equilibrium, similarly to sag amplitude, $A_{constant} = ||\mathbf{K}^{-1}\mathbf{M}\ddot{\mathbf{p}}||$; this leads to the formula $\varepsilon = \chi^2\bar{A}/A$. In scenario (2), the short acceleration impulse produces an initial velocity $\mathbf{v_0} = h\ddot{\mathbf{p}}$, and hence we have $\overline{\mathbf{v_0}} = \varepsilon\mathbf{v_0}$. The kinetic energy will transfer to the potential energy,

$$\frac{1}{2}\mathbf{v_0}^T\mathbf{M}\mathbf{v_0} = \frac{1}{2}\mathbf{u_{max}}^T\mathbf{K}\mathbf{u_{max}}. \quad (29)$$

The inertial amplitude is $A_{impulse} = ||\mathbf{u_{max}}||$. We have

$$\frac{1}{2}\overline{\mathbf{u_{max}}}^T\overline{\mathbf{K}}\overline{\mathbf{u_{max}}} = \frac{1}{2}\frac{\varepsilon^2}{\chi^2}\mathbf{u_{max}}^T\overline{\mathbf{K}}\mathbf{u_{max}}, \quad (30)$$

and therefore $\varepsilon = \chi\bar{A}/A$. Under general constraints, the system will deviate from the idealized conditions (1), and (2). We address this using a two-stage process. In the first stage, we set ε based on whichever case (1) and (2) is most similar to the specific constraint motion in the animation; then re-run the simulation under the $\chi, \mu, c_k, \varepsilon$ as determined in this section. This will result in frequency, sag, and damping that (under linear physics) match those requested by the animator, but the amplitude will only be matched approximately. We can then compute the actual amplitude A_{actual} using Equation 5. Finally, we observe that if one *only* changes ε to ε' (as opposed to also χ, μ, c_k), the output amplitude scales approximately *linearly* in ε'/ε. We therefore set $\varepsilon' = \varepsilon\bar{A}/A_{\text{actual}}$ and re-run the simulation. Due to the presence of nonlinearities in f_{int} and due to contact, the output amplitude will still not match $\bar{A}$. However, the output amplitude is generally (modulo contact) a monotonic function of ε, and we can continue tweaking ε, for example, using bisection. Although this requires re-simulation, we emphasize that in the absence of our "adjusted physics" technique, no amplitude control is possible at all, due to coupling between amplitude and frequency.

6. Results

We demonstrate our method on four examples (Table 1). The constraints on the sunflower and jellyfish undergo rigid body motion; this geometric proxy motion was scripted by us. Our method can be used for cartoon animation effects [CPIS02, GDO07]; namely, controlling squash and stretch of soft deformable objects, as seen in our sumo and "jellyphant" examples. These two examples are driven by non-rigid constraints; the non-rigid motions of the proxy were animated by a professional animator using linear blend skinning and pose-space deformation [LCF00]. We timestepped our deformable models with implicit backward Euler [BW98] and implicit Bathe's trapezoid-BDF2 method [XB17]. We use implicit Bathe's method in our sunflower, jellyfish and sumo examples, and implicit backward Euler in the jellyphant. In Figure 7, we compare our method to a recent method that permits the animator to combine geometric motion with physics [LXB17]. It can be seen that our method produces superior contact handling.

Performance of our modified equations is analyzed in Table 1. The jellyphant example needs a large number of substeps because of very challenging self-contact; this has nothing to do with our method; it occurs with and without our approach. When the point cloud undergoes rigid body motion and there is no gravity (jellyfish example), static solving is not needed because one can just translate and rotate the neutral tet mesh. In the sumo and jellyphant examples, the static solving time can be seen to be approximately 1/4 of the dynamic simulation time, due to good temporal coherence among the substeps. Therefore, when a standard method spends a unit of computation time, our simulation with adjusted physics spends 1.25 units because it needs to do the static solves, i.e., 25% overhead. This overhead is relatively small, but certainly not negligible. That said, our method offers the ability to adjust the dynamics amplitude that cannot be easily achieved with other methods. The calculation of rotations, their time derivatives and inertial forces is very fast; it takes 1000x less time than the total timestep computation in the sumo and jellyphant examples (Table 1, column

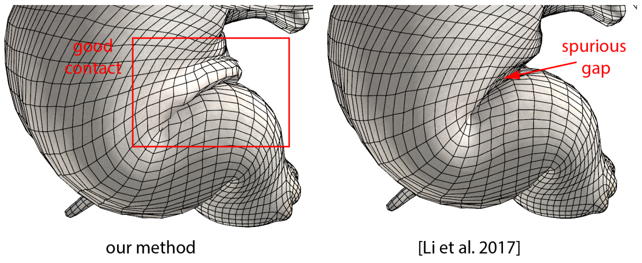

Figure 7: ***Comparison to [LXB17]:*** *Left: Our method only modifies the inertial terms, but otherwise follows standard equations of motion. As such, one can add character self-contact handling in any usual way (we use implicit penalty-based contact). This results in correct contact handling, and good fold formation at the contact site. Right: The geometric rotation-aware blending between the static shape and dynamic output produces collision artifacts. In this case, a wide spurious gap appears at the place where there should be tight contact. This gap cannot be removed even with the "blending-aware" contact handling technique presented in [LXB17] because that technique only works when geometric interpolation causes a collision; it cannot resolve the opposite case, namely spurious gap formation.*

"adjusted"). Evaluating the adjusted physics terms has a negligible memory overhead as the static solves can be performed on the fly together with the dynamic solves, and can re-use the same internal force and tangent stiffness matrix calculation datastructures.

In Figure 8, we demonstrate that, under linear physics, we can keep two of the frequency, damping and inertial amplitude fixed while modifying the third one. Figure 10 demonstrates that we can adjust the sag while keeping frequency unchanged. In Figure 11, we demonstrate that we can adjust inertial amplitude independently of frequency, using our scaling of inertial forces. Figure 9 gives an analysis on a nonlinear example and demonstrates a typical animator "user story" of simultaneously adjusting both the frequency and inertial amplitude.

Acknowledgements

This research was sponsored in part by NSF (IIS-1911224), USC Annenberg Fellowship to Bohan Wang and Mianlun Zheng, Bosch Research and Adobe Research.

Conflict of interest statement

The jellyphant mesh and its geometric proxy motion was obtained at Ziva Dynamics. Jernej Barbič is a shareholder, CTO and board member of Ziva Dynamics. Ziva Dynamics was not involved in this research. Nothing in this paper is to be understood as endorsement of Ziva Dynamics or its products.

7. Conclusion

We contributed the observation that the frequency and amplitude of soft tissue dynamics in a constrained system are inherently coupled and cannot be adjusted independently using correct physics.

Model	Core type	Gravity	Material model	frames	substeps	static [min]	dynamic [min]	adjusted [min]
Sunflower	Rigid	✓	Linear FEM	300	1	0.17	1.3	0.0053
Sunflower	Rigid	✓	StVK	300	1	0.2	4.0	0.0052
Jellyfish	Rigid	X	StVK	313	5	0	6.5	0.011
Sumo	Deformable	✓	StVK	400	4	6.3	25.3	0.026
Jellyphant	Deformable	✓	stable neo-Hookean	120	50	107	401	0.36

Table 1: ***Performance.*** *Columns "frames" and "substeps" give the total number of graphics frames (at 24 FPS) and the number of simulation steps per graphical frame, respectively. Columns "static", "dynamic" and "adjusted" give the total simulation time (for all frames) to perform the static solves, dynamic solves (including our approach), and the time to evaluate the adjusted physics terms in the equations of motion, respectively. We add a volume preservation term to StVK which prevents large compressions and successfully guards against material collapse. Stable neo-Hookean is from [SGK18].*

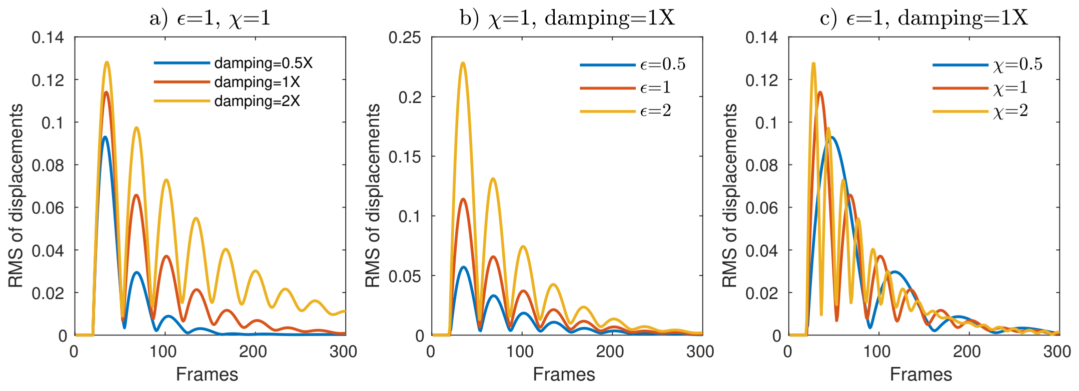

Figure 8: ***Control of frequency, damping and inertial amplitude while keeping the other two quantities fixed.*** *Linear sunflower model.*

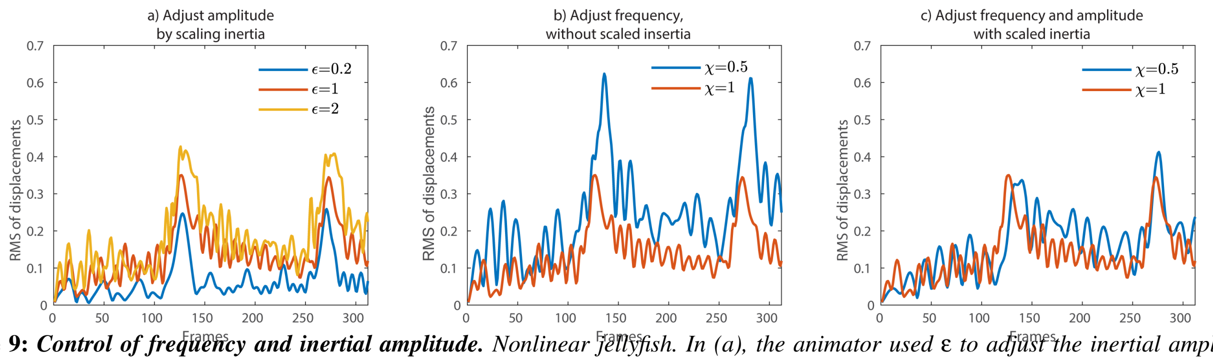

Figure 9: ***Control of frequency and inertial amplitude.*** *Nonlinear jellyfish. In (a), the animator used ε to adjust the inertial amplitude; she did not change the frequency. In (b), she did not scale the inertia forces, but just adjusted the frequency. This automatically caused the amplitudes to grow. In (c), she corrected this, using scaled inertia forces; and now, the amplitudes are back to what they were originally, but the frequency is now half of the original one, as desired.*

We gave a method to decouple them, by transforming the equations of motion to non-inertial coordinate frames, and then scaling the inertial forces. Furthermore, for linear physics, we derived precise formulas for how to adjust the physical parameters such as mass, stiffness (Young's modulus), gravity acceleration, Rayleigh stiffness damping and inertial force scaling factor, to meet prescribed output visual characteristic such as natural vibration frequency, inertial amplitude, gravity sag and damping. In the absence of our method, animators need to tweak the physical parameters directly, and their relationship to the output visual characteristic is complex and un-intuitive. Furthermore, independently tweaking frequency and amplitude is not possible. Our method enables the animator to directly and independently control the salient visual characteristics of constrained dynamics, which shortens the trial and error process during animation design. We demonstrated our method by animating the geometric proxy point cloud using linear blend skinning and pose-space deformation and simulating the soft-tissue dynamics using the Finite Element Method; but the idea is general and applicable to other geometric animation and soft-body simulation methods. In computer animation, we are primarily interested in dynamic vibrations (often colloquially referred to as "jiggling" by animators), and hence we only considered under-damped systems. Damping also changes the natural frequency of vibration; incorporating this effect would be relatively straightforward and we

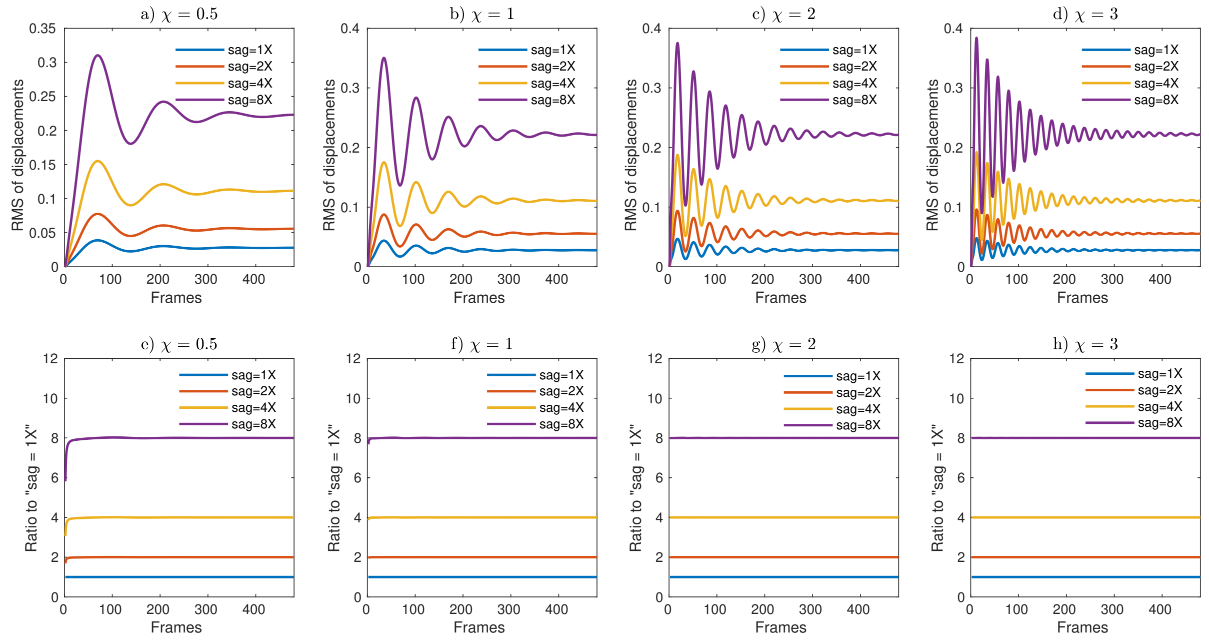

Figure 10: ***Interplay of frequency and sag.*** *Linear sunflower model. Bottom row shows that sag is as prescribed, regardless of frequency.*

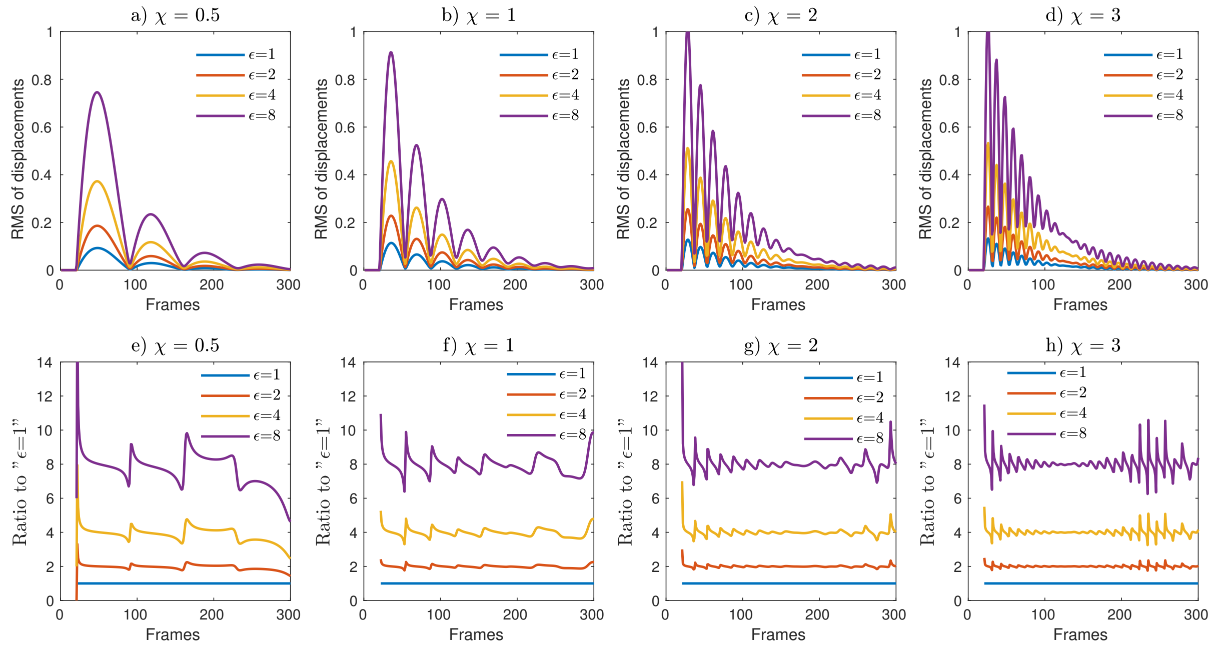

Figure 11: ***Interplay of frequency and inertial amplitude scaling***. *Linear sunflower model. Bottom row shows that the amplitude approximately matches the prescribe one, regardless of frequency. While linearity makes it easier to directly match the desired amplitudes, the curves in the bottom row are not flat due to the presence of damping which causes oscillations to go out of phase.*

leave it for future work. As they are often secondary in nature to inertial dynamics, we did not attempt to modify the magnitude of dynamic deformations occurring due to contact; doing so would be interesting future work. Our physical parameter adjustment formulas assume linear physics. Although such formulas are not exact for nonlinear large deformation motion, they still provide guidance to the animator. Creating more precise guidance for nonlinear dynamics is important future work.

References

[BK04] BOTSCH M., KOBBELT L.: An intuitive framework for real-time freeform modeling. vol. 23, pp. 630–634. 3

[BS19] BARRIELLE V., STOIBER N.: Realtime performance-driven physical simulation for facial animation. In *Computer Graphics Forum* (2019), vol. 38, pp. 151–166. 3

[BSC16] BARRIELLE V., STOIBER N., CAGNIART C.: Blendforces: A dynamic framework for facial animation. In *Computer Graphics Forum* (2016), vol. 35, pp. 341–352. 3

[BW98] BARAFF D., WITKIN A. P.: Large Steps in Cloth Simulation. In *Proc. of ACM SIGGRAPH 98* (July 1998), pp. 43–54. 8

[BZ11] BARBIČ J., ZHAO Y.: Real-time large-deformation substructuring. *ACM Trans. on Graphics (SIGGRAPH 2011) 30*, 4 (2011), 91:1–91:7. 7

[CBC*05] CAPELL S., BURKHART M., CURLESS B., DUCHAMP T., POPOVIĆ Z.: Physically based rigging for deformable characters. In *Proc. of Symposium on Computer Animation* (2005), pp. 301–310. 1, 2

[CLK*19] CHEN Y. J., LEVIN D. I., KAUFMANN D., ASCHER U., PAI D. K.: Eigenfit for consistent elastodynamic simulation across mesh resolution. In *Symp. on Computer Animation (SCA)* (2019), pp. 1–13. 3

[CLMK17] CHEN D., LEVIN D. I. W., MATUSIK W., KAUFMAN D. M.: Dynamics-aware numerical coarsening for fabrication design. *ACM Trans. Graph. 36*, 4 (2017). 3

[CPIS02] CHENNEY S., PINGEL M., IVERSON R., SZYMANSKI M.: Simulating cartoon style animation. In *Proc. of the 2nd Int. Symp. on Non-Photorealistic Animation and Rendering* (2002), pp. 133–138. 8

[DBC*15] DAVIS A., BOUMAN K. L., CHEN J. G., RUBINSTEIN M., DURAND F., FREEMAN W. T.: Visual vibrometry: Estimating material properties from small motions in video. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)* (2015), pp. 5335–5343. 3

[GBFP11] GILLES B., BOUSQUET G., FAURE F., PAI D. K.: Frame-based elastic models. *ACM Trans. on Graphics (TOG) 30*, 2 (2011). 2

[GDO07] GARCIA M., DINGLIANA J., O'SULLIVAN C.: A physically based deformation model for interactive cartoon animation. In *VRIPHYS Workshop* (2007). 8

[GOT*07] GALOPPO N., OTADUY M. A., TEKIN S., GROSS M., LIN M. C.: Soft articulated characters with fast contact handling. In *Computer Graphics Forum* (2007), vol. 26, pp. 243–253. 2

[HMT*12] HAHN F., MARTIN S., THOMASZEWSKI B., SUMNER R., COROS S., GROSS M.: Rig-space physics. *ACM Trans. on Graphics (TOG) 31*, 4 (2012), 1–8. 1, 2

[HTC*13] HAHN F., THOMASZEWSKI B., COROS S., SUMNER R. W., GROSS M.: Efficient simulation of secondary motion in rig-space. In *Symp. on Computer Animation (SCA)* (2013), pp. 165–171. 1, 2

[JBPS11] JACOBSON A., BARAN I., POPOVIĆ J., SORKINE O.: Bounded biharmonic weights for real-time deformation. *ACM Trans. on Graphics (TOG) 30*, 4 (2011), 78. 3

[JP02] JAMES D. L., PAI D. K.: DyRT: Dynamic Response Textures for Real Time Deformation Simulation With Graphics Hardware. *ACM Trans. on Graphics 21*, 3 (2002), 582–585. 5, 6

[KBB*17] KOZLOV Y., BRADLEY D., BÄCHER M., THOMASZEWSKI B., BEELER T., GROSS M.: Enriching facial blendshape rigs with physical simulation. In *Computer Graphics Forum* (2017), vol. 36, pp. 75–84. 3

[KDGI19] KIM T., DE GOES F., IBEN H.: Anisotropic elasticity for inversion-safety and element rehabilitation. *ACM Trans. on Graphics (SIGGRAPH 2019) 38*, 4 (2019). 1, 3

[KJ12] KIM T., JAMES D. L.: Physics-based character skinning using multidomain subspace deformations. *IEEE Trans. on Visualization and Computer Graphics 18*, 8 (2012), 1228–1240. 2

[KP11] KIM J., POLLARD N. S.: Fast simulation of skeleton-driven deformable body characters. *ACM Transactions on Graphics (TOG) 30*, 5 (2011), 1–19. 2

[LCF00] LEWIS J. P., CORDNER M., FONG N.: Pose Space Deformations: A Unified Approach to Shape Interpolation and Skeleton-Driven Deformation. In *Proc. of ACM SIGGRAPH* (2000), pp. 165–172. 3, 8

[LSY97] LEHOUCQ R., SORENSEN D., YANG C.: *ARPACK Users' Guide: Solution of large scale eigenvalue problems with implicitly restarted Arnoldi methods*. Tech. rep., Comp. and Applied Mathematics, Rice Univ., 1997. 4

[LXB17] LI Y., XU H., BARBIČ J.: Enriching triangle mesh animations with physically based simulation. *IEEE Transactions on Visualization and Computer Graphics 23*, 10 (2017), 2301–2313. 1, 3, 8

[LYWG13] LIU L., YIN K., WANG B., GUO B.: Simulation and control of skeleton-driven soft body characters. *ACM Transactions on Graphics (TOG) 32*, 6 (2013), 1–8. 2

[MG04] MÜLLER M., GROSS M.: Interactive Virtual Materials. In *Proc. of Graphics Interface 2004* (2004), pp. 239–246. 5

[MGL*15] MALGAT R., GILLES B., LEVIN D. I., NESME M., FAURE F.: Multifarious hierarchies of mechanical models for artist assigned levels-of-detail. In *Symp. on Computer Animation (SCA)* (2015). 2

[MTGG11] MARTIN S., THOMASZEWSKI B., GRINSPUN E., GROSS M.: Example-based elastic materials. *ACM Trans. on Graphics (SIGGRAPH 2011) 30*, 4 (2011), 72. 2

[MZS*11] MCADAMS A., ZHU Y., SELLE A., EMPEY M., TAMSTORF R., TERAN J., SIFAKIS E.: Efficient elasticity for character skinning with contact and collisions. *ACM Trans. on Graphics (SIGGRAPH 2011) 30*, 4 (2011). 2

[PMĎ15] PIOVARČI M., MADARAS M., ĎURIKOVIČ R.: Physically inspired stretching for skinning animation of non-rigid bodies. In *Proc. of Spring Conference on Computer Graphics* (2015), pp. 47–53. 2

[SA07] SORKINE O., ALEXA M.: As-rigid-as-possible surface modeling. In *Symp. on Geometry Processing* (2007), vol. 4, pp. 109–116. 3

[SGK18] SMITH B., GOES F. D., KIM T.: Stable neo-hookean flesh simulation. *ACM Trans. Graph. 37*, 2 (2018), 12:1–12:15. 3, 9

[SGK19] SMITH B., GOES F. D., KIM T.: Analytic eigensystems for isotropic distortion energies. *ACM Trans. Graphics (TOG) 38*, 1 (2019). 3

[Sha90] SHABANA A. A.: *Theory of Vibration, Volume II: Discrete and Continuous Systems*. Springer–Verlag, 1990. 3

[SZT*08] SHI X., ZHOU K., TONG Y., DESBRUN M., BAO H., GUO B.: Example-based dynamic skinning in real time. *ACM Transactions on Graphics (TOG) 27*, 3 (2008), 1–8. 2

[Tis13] TISSUE: Weta Digital: Tissue Muscle and Fat Simulation System, 2013. 1

[TKA11] TWIGG C. D., KAČIĆ-ALESIĆ Z.: Optimization for sag-free simulations. In *Symp. on Computer Animation (SCA)* (2011), pp. 225–236. 5

[WZB17] WANG B., ZHAO Y., BARBIČ J.: Botanical materials based on biomechanics. *ACM Trans. on Graphics (TOG) 36*, 4 (2017), 1–13. 3

[XB16] XU H., BARBIČ J.: Pose-space subspace dynamics. *ACM Trans. on Graphics (SIGGRAPH 2016) 35*, 4 (2016). 1, 2, 4

[XB17] XU H., BARBIČ J.: Example-based damping design. *ACM Trans. on Graphics (SIGGRAPH 2017) 36*, 4 (2017). 3, 8

[XSZB15] XU H., SIN F., ZHU Y., BARBIČ J.: Nonlinear material design using principal stretches. *ACM Trans. on Graphics (SIGGRAPH 2015) 34*, 4 (2015), 75:1–75:11. 3