Global Momentum Preservation for Position-based Dynamics

Alex Dahl
University of Maryland, Baltimore County
adahl1@umbc.edu

Adam W. Bargteil
University of Maryland, Baltimore County
adamb@umbc.edu

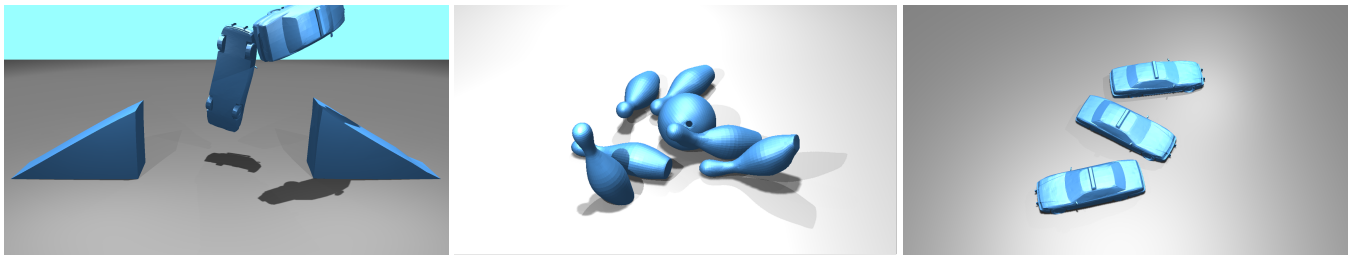

Figure 1: Images from several example scenes using our technique.

ABSTRACT

Position-based dynamics has emerged as an exceedingly popular approach for animating soft body dynamics. Unfortunately, the basic approach suffers from artificial loss of angular momentum. We propose a simple approach to preserve global linear and angular momenta of bodies by directly tracking these quantities and adjusting velocities to ensure they are preserved. This approach entails negligible computational cost, requires less than 25 lines of code, and exactly preserves global linear and angular momenta.

CCS CONCEPTS

• **Computing methodologies** → **Procedural animation**; **Physical simulation**.

KEYWORDS

position-based dynamics, linear and angular momentum, computer animation, physics simulation

ACM Reference Format:
Alex Dahl and Adam W. Bargteil. 2019. Global Momentum Preservation for Position-based Dynamics. In *Motion, Interaction and Games (MIG '19), October 28–30, 2019, Newcastle upon Tyne, United Kingdom.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3359566.3360078

1 INTRODUCTION

Position-based dynamics has emerged as a popular tool for computer animation of soft body dynamics. However, the approach suffers from significant loss of angular momentum, especially for stiff materials. For softer materials the loss of angular momentum is less objectionable, however, if the common mass-proportional damping model is employed, as in *Bullet*, it will damp rigid body motion.[1]

Inspired by the pioneering work of Terzopoulos and Witkin [1988], we propose a straightforward fix: decouple rigid body motion and deformation. We explicitly track the global linear and angular momenta of the body through time and use these quantities to correct nodal velocities. Specifically, we advance the position-based dynamics as usual keeping track of any collision impulses (or other external forces) applied to the object. At the end of the timestep, we update the global linear and angular momenta of the body based on the impulses applied during the timestep. We then correct the velocities of the soft body mesh to preserve the global momentum.

This approach is simple to implement, adds negligible cost, and exactly preserves global linear and angular momenta. We have implemented our approach for soft bodies in the popular *Bullet* physics engine [Coumans 2014]. Frames from several example scenes can be seen in Figure 1.

Our contributions include both demonstrating analytically and experimentally that position-based dynamics does not preserve energy or angular momentum and presenting an approach to restore this lost momentum.

2 RELATED WORK

For a detailed review of position-based dynamics, we refer the reader to the survey by Bender and colleagues [2014b] or the more recent course notes by Bender and colleagues [2017].

The idea of treating elastic deformation with constraints dates at least to the work of Provot [1995] who enforced a maximum extension to springs in a spring-mass system, an idea commonly refered to as *strain limiting*. Perhaps the first to entirely forgo force calculations and treat elasticity exclusively through constraints was Jakobsen [2001] in his *Fysix* system. Müller and colleagues [2007] generalized this approach to general constraints and coined the term *position-based dynamics*. Stam [2009] implemented a variation of position-based dynamics for the *Nucleus* engine in *Maya*. In

[1] We note that Müller and colleagues [2007] did introduce a damping model that avoids damping rigid modes and is very similar to the approach we propose. However, most position-based dynamics codes do not seem to employ this model.

MIG '19, October 28–30, 2019, Newcastle upon Tyne, United Kingdom

ACM ISBN 978-1-4503-6994-7/19/10.
https://doi.org/10.1145/3359566.3360078

all these approaches, compliance of elastic bodies is achieved by explicitly choosing not to enforce constraints exactly. Typically one iterates over the constraints approximately enforcing them one-by-one, with more iterations resulting in a stiffer material.

Due to the method's unconditional stability it has seen widespread adoption, especially in interactive environments and has remained a popular research topic. Kelager and colleagues [2010] introduced a triangle bending model. Macklin and Müller [2013] extended the method to fluids. Macklin and colleagues [2014] introduced a unified system able to handle soft bodies, rigid bodies, and fluids. Bender and colleagues [2014a] extended PBD to continuum models by applying constraints to a computed stress. In concurrent work, Mueller and colleagues [2014] added constraints on the components of Green's strain, allowing for anisotropy and separate handling of stretch and shear deformations. Macklin and colleagues [2016] introduced XPBD, which addressed the dependence of apparent stiffness on iteration count and time step and results in a method that more closely resembles traditional implicit integration, though they retain the general approach of solving directly for positions rather than velocities.

Our approach is inspired by the work of Terzopoulos and Witkin [1988], which factors motion into rigid body motion and deformation. We similarly track a body's global linear and angular momenta and enforce these as a constraint on the particles' velocities. Our approach is very similar to the damping model originally proposed by Müller and colleagues [2007] in that both involve projecting out rigid modes. While their approach sought to avoid damping rigid modes, our approach seeks to preserve rigid momentum.

Finally, we note that Bender and colleagues [2017] discuss reducing damping by adopting a higher order velocity update based on BDF2 [English and Bridson 2008]. This approach is quite effective at reducing damping, but can result in instability and does not exactly preserve momentum.

3 BACKGROUND

For simplicity of exposition, we assume that our soft body is represented by a mesh of *nodes* and that edges represent constraints between those nodes, similar to a traditional Spring-Mass network. Generalizations beyond this case, for example to strain-based constraints, should be straightforward.

Algorithm 1 Position-based Dynamics

```
1:  for Node n : nodes do
2:      n.oldpos = n.pos
3:      n.vel += dt · frc / n.mass
4:      n.pos += dt · n.vel
5:  end for
6:  for i<niterations do
7:      projectConstraints(constraints, nodes)
8:  end for
9:  for Node n : nodes do
10:     n.vel = (n.pos - n.oldpos) / dt
11: end for
```

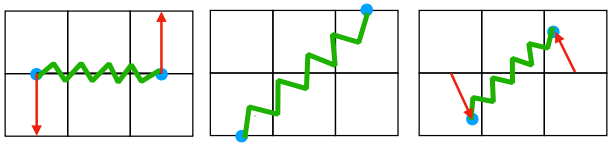

Figure 2: A simple example demonstrating the loss of angular momentum present in PBD.

3.1 Position-based Dynamics

Algorithm 1 summarizes the basic position-based dynamics algorithm. We first loop over all the nodes integrating velocities (based on external forces such as gravity) and positions forward in time. Then there are some number of iterations where constraints are enforced by directly updating the nodes' positions. Importantly, constraints are not typically solved exactly, allowing for some deformation. Finally, we loop over the nodes again setting the velocity based on the change in position.

Line 10 in Algorithm 1 is critical to understanding the loss of angular momentum when using PBD. Without line 10, PBD would resemble more traditional constraint-based approaches in simulation. Examples of such approaches include using Lagrange multipliers (i.e. pressure values) to enforce the divergence free constraint in fluid simulation and strain limiting in cloth simulation. In these approaches, kinetic energy is explicitly removed from the system instead of being stored as internal energy. Line 10, however, casts PBD as an optimization method that solves for positions directly and updates velocities instead of the more traditional approach that solves for velocities and then updates positions [Baraff and Witkin 1998]. Thus the artificial loss of angular momentum in PBD stems not from explicit removal of energy, but from the linearization in the velocity update in line 10.

3.2 A Simple Example

We begin with a simple example that demonstrates why PBD leads to loss of angular momentum (see Figure 2). Imagine two particles with mass 1 at positions $(1,0)^T$ and $(-1,0)^T$ connected by a constraint. Let the first particle have velocity $(0,1)^T$ and the other $(0,-1)^T$. If we take a timestep of size 1 second, the particles positions will be $(1,1)^T$ and $(-1,-1)^T$, respectively. Now if we enforce the constraint the particles will move to $(1/\sqrt{2}, 1/\sqrt{2})^T$ and $(-1/\sqrt{2}, -1/\sqrt{2})^T$, respectively. When we update the velocities based on the change in position, we get $(1/\sqrt{2}-1, 1/\sqrt{2})^T$) and $(-1/\sqrt{2}+1, -1/\sqrt{2})^T$). Clearly the (kinetic) energy of the system, $\frac{1}{2}\mathbf{v}^T\mathbf{M}\mathbf{v}$, before the timestep was 1 joules and after the timestep it has been reduced to $2-\sqrt{2} \approx 0.59$ joules. More than 40% of the energy has been lost. Similarly, the angular momentum decreases in magnitude from 2 kg·m/s to $\sqrt{2} \approx 1.41$ kg·m/s, dumping almost 30% of the angular momentum.

We stress that this example is simple and clearly demonstrates that basic PBD necessarily lose energy, but that the example is not typical of simulations that use position-based dynamics. In practice the loss of angular momentum is less pronounced, primarily because constraints are not fully satisfied. However, the loss of angular momentum is usually significant enough to be noticed. Figure 3 shows the falloff of kinetic energy and angular momentum

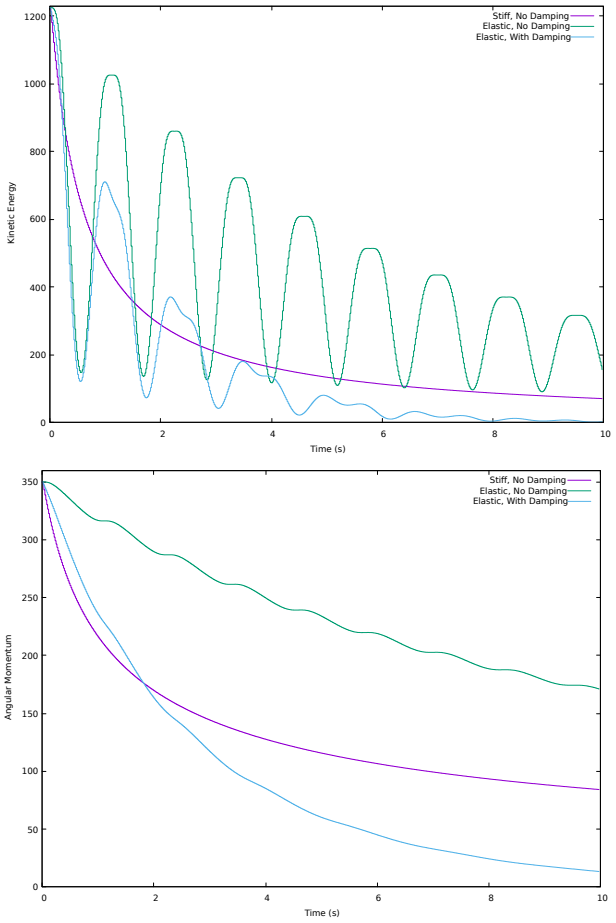

Figure 3: Falloff in kinetic energy (top) and angular momentum (bottom) for a simple example of two mass points connected by a single spring. The purple curves show the falloff when the constraint is perfectly enforced, the green curves show the falloff when the constraint is softened, and the blue curves show the falloff when the constraint is soft, but mass-proportional damping is added.

of several three dimensional versions of this simple example of two mass point connected by a single constraint. Video results are included in the supplemental material. The purple curves show the falloff when the constraint is perfectly enforced, the green curves show the falloff when the constraint is softened, and the blue curves show the falloff when the constraint is soft, but mass-proportional damping is added.

4 METHOD

In this section we describe our approach for momentum preservation. Suppose that after time integration and collision processing the velocity of the body is described by a vector $\hat{\mathbf{v}}$ of nodal velocities with length $3n$, where the soft body mesh has n nodes. Also suppose that the rigid linear and angular momenta, $\mathbf{P}_r$ and $\mathbf{L}_r$ respectively, are 3-vectors that are updated through time to account for collision

Algorithm 2 Momentum Preservation

```
1:  m = 0.0;
2:  com = 0.0;
3:  I = 0.0;
4:  P_pbd = 0.0;
5:  L_pbd = 0.0;
6:  for Node n : nodes do
7:      m += n.mass
8:      com += n.mass · n.pos
9:  end for
10: com /= mass
11: for Node n : nodes do
12:     r = n.pos - com
13:     I += n.mass · star(r) · star(r).transpose()
14:     P_pbd += n.mass · n.vel
15:     L_pbd += n.mass · r.cross(n.vel)
16: end for
17: v_cor = (P_r - P_pbd) / m
18: omega_cor = I.inverse() · (L_r - L_pbd)
19: for Node n : nodes do
20:     n.vel += v_cor + omega_cor.cross(n.pos - com)
21: end for
```

impulses and external forces. For example, if an impulse, $\mathbf{i}$, is applied to node n_i with position $\mathbf{x}_i$, then the rigid linear and angular momenta are updated as

$$\mathbf{P}_r \mathrel{+}= m_i \mathbf{i} \tag{1}$$

$$\mathbf{L}_r \mathrel{+}= m_i \left(\mathbf{x}_i - \mathbf{com}\right) \mathbf{i}, \tag{2}$$

where **com** is the body's center of mass.

Finally, let $\mathbf{P}_{pbd}$, $\mathbf{L}_{pbd}$, m, and $\mathbf{I}$ be the linear momentum, angular momentum, total mass, and inertia tensor, respectively, implied by the mass, position, and velocity of the PBD particles. These quantities are easily computed by looping twice over the nodes (see Algorithm 2). We can then compute corrections for the linear and angular velocities,

$$\mathbf{v}_{cor} = \left(\mathbf{P}_r - \mathbf{P}_{pbd}\right)/m \tag{3}$$

$$\omega_{cor} = \mathbf{I}^{-1}\left(\mathbf{L}_r - \mathbf{L}_{pbd}\right), \tag{4}$$

and update the nodal velocities. Pseudocode is given in Algorithm 2. Here the star(·) operator applied to vector $\mathbf{r}$ yeilds the cross-product matrix,

$$\mathbf{r}^* = \begin{pmatrix} 0 & -r_z & r_y \\ r_z & 0 & -r_x \\ -r_y & r_x & 0 \end{pmatrix} \tag{5}$$

After initializing accumulators (lines 1-5), the first loop computes the center of mass (lines 6-10). The second computes the inertia tensor and the soft body simulation's estimate of the linear and angular momenta (lines 11-16). Then a velocity correction is computed (lines 17-18) and a third loop applies this correction to the nodes (lines 19-21).

A more complete derivation is given in Appendix A.

Integration into Position-based Dynamics. We used *Bullet* for our Position-based Dynamics engine. Integrating our approach into *Bullet* was straightforward. Our algorithm updates the soft

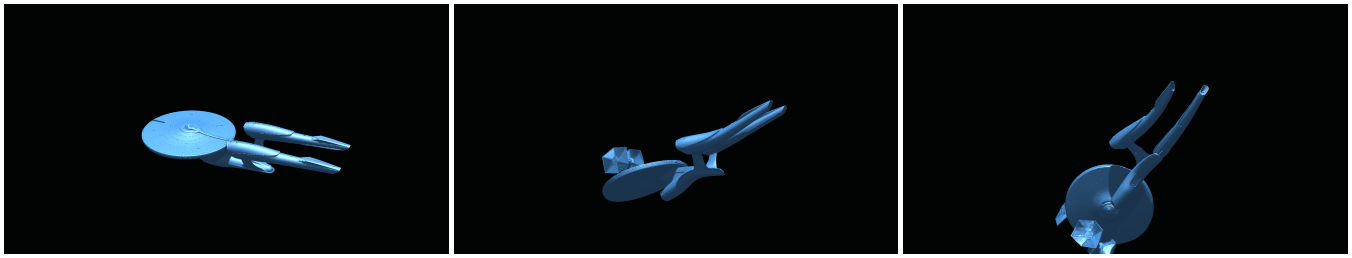

Figure 4: Frames from another scene created using our approach for momentum preservation.

bodies' velocities directly at the end of the timestep, i.e. after line 11 in Algorithm 1. Additionally, as *Bullet* processes collisions, applying impulses, we update the global linear and angular momenta of our soft bodies. An alternative approach would track not only the linear and angular momenta through time, but also integrate these to track position and orientation as in a typical rigid body simulation. Then we could treat the constraint that the soft bodies position and orientation match the tracked values just like any other position-based dynamics constraint and include them in the constraint projection in line 7 of Algorithm 1.

5 RESULTS AND DISCUSSION

Because momentum is difficult to gauge from still frames, please see the accompanying video for several demonstrations of our technique and comparisons to the behavior of position-based dynamics without our momentum preservation approach. Table 1 shows the time taken to perform the correction per frame compared to the total time of the simulation per frame. The video begins with our simple example with two mass points connected by a constraint. We then show another didactic, but more complex, example of a rotating cube. A few more practical examples follow and frames from these scenes are in Figure 1. Several frames from our final example are shown in Figure 4.

We note that friction was disabled in all examples to avoid muddling the effects of friction and damping. Additionally, in all comparisons, our momentum preserving example used the same damping coefficient as the damped default version.

Limitations and Future Work. The biggest limitation of our approach is that in a large subset of practical applications it is solving a non-problem—the loss of angular momentum present in many position-based dynamics simulations is not noticeable unless compared against a momentum preserving simulation. However, we believe that if our approach was added as a feature in position-based dynamics packages, it would probably be used by default because it dramatically improves results in extreme cases, mildly improves results in most cases, and in no case degrades results.

Our general approach of separating rigid motion from deformation should be applicable to almost any soft body simulation and would be useful in other simulation environments prone to loss of angular momentum due to numerical issues or approximation errors.

In summary, our technique for momentum preservation for position-based dynamics is simple and entails negligible computational cost. We encourage anyone using position-based dynamics to incorporate our approach.

REFERENCES

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

[Bender et al. 2014a] Bender, Koschier, Charrier, and Weber. 2014a. Position-Based Simulation of Continuous Materials. *Computers & Graphics* 44, 0 (2014), 1 – 10. https://doi.org/10.1016/j.cag.2014.07.004

[Bender et al. 2017] Bender, Müller, and Macklin. 2017. Position-Based Simulation Methods in Computer Graphics. In *EUROGRAPHICS 2017 Tutorials*. Eurographics Association. https://doi.org/10.2312/egt.20171034

[Bender et al. 2014b] Bender, Müller, Otaduy, Teschner, and Macklin. 2014b. A Survey on Position-Based Simulation Methods in Computer Graphics. *Comput. Graph. Forum* 33, 6 (Sept. 2014), 228–251. https://doi.org/10.1111/cgf.12346

[Coumans 2014] Coumans. 2014. Bullet Physics Library. http://bulletphysics.org/.

[English and Bridson 2008] English and Bridson. 2008. Animating Developable Surfaces Using Nonconforming Elements. *ACM Trans. Graph.* 27, 3, Article 66 (Aug. 2008), 5 pages. https://doi.org/10.1145/1360612.1360665

[Jakobsen 2001] Jakobsen. 2001. Advanced Character Physics. In *Game Developers Conference 2001*.

[Kelager et al. 2010] Kelager, Niebe, and Erleben. 2010. A Triangle Bending Constraint Model for Position-Based Dynamics. In *Workshop in Virtual Reality Interactions and Physical Simulation "VRIPHYS" (2010)*, Kenny Erleben, Jan Bender, and Matthias Teschner (Eds.). The Eurographics Association. https://doi.org/10.2312/PE/vriphys/vriphys10/031-037

[Macklin and Müller 2013] Macklin and Müller. 2013. Position Based Fluids. *ACM Trans. Graph.* 32, 4, Article 104 (July 2013), 12 pages. https://doi.org/10.1145/2461912.2461984

[Macklin et al. 2016] Macklin, Müller, and Chentanez. 2016. XPBD: Position-based Simulation of Compliant Constrained Dynamics. In *Proceedings of the 9th International Conference on Motion in Games (MIG '16)*. ACM, New York, NY, USA, 49–54. https://doi.

Table 1: Table of simulation timing with the total number of vertices in the scene, average simulation time per frame in milliseconds, average time correcting the velocities per frame in milliseconds, and the percentage of the average simulation time per frame spent performing the correction. Timing does not include rendering.

Examples	# Vertices	Average Time Per Frame (ms)	Average Correction Per Frame (ms)	Percentage in Correction
4 Cubes	400	3.117	0.014	0.45%
Bowling	475	5.603	0.019	0.33%
Car Ramp	3272	42.997	0.194	0.45%
Car Spin	4734	61.043	0.229	0.38%
Enterprise	4334	45.569	0.264	0.58%

org/10.1145/2994258.2994272

[Macklin et al. 2014] Macklin, Müller, Chentanez, and Kim. 2014. Unified Particle Physics for Real-time Applications. *ACM Trans. Graph.* 33, 4, Article 153 (July 2014), 12 pages. https://doi.org/10.1145/2601097.2601152

[Müller et al. 2014] Müller, Chentanez, Kim, and Macklin. 2014. Strain Based Dynamics. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA '14)*. Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 149–157. http://dl.acm.org/citation.cfm?id=2849517.2849542

[Müller et al. 2007] Müller, Heidelberger, Hennix, and Ratcliff. 2007. Position Based Dynamics. *J. Vis. Comun. Image Represent.* 18, 2 (April 2007), 109–118. https://doi.org/10.1016/j.jvcir.2007.01.005

[Provot 1995] Provot. 1995. Deformation constraints in a mass-spring model to describe rigid cloth behavior. In *IN GRAPHICS INTERFACE*. 147–154.

[Stam 2009] Stam. 2009. Nucleus: Towards a unified dynamics solver for computer graphics. In *2009 11th IEEE International Conference on Computer-Aided Design and Computer Graphics*. 1–11. https://doi.org/10.1109/CADCG.2009.5246818

[Terzopoulos and Witkin 1988] Terzopoulos and Witkin. 1988. Physically Based Models with Rigid and Deformable Components. *IEEE Comput. Graph. Appl.* 8, 6 (Nov. 1988), 41–51. https://doi.org/10.1109/38.20317

A DERIVATION

Letting $\boldsymbol{\Psi} = (\mathbf{P}_r\ \mathbf{L}_r)^T$, we can setup the following system

$$\begin{pmatrix} \mathbf{M} & \mathbf{J}^T \\ \mathbf{J} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{v} \\ \lambda \end{pmatrix} = \begin{pmatrix} \mathbf{M}\hat{\mathbf{v}} \\ \boldsymbol{\Psi} \end{pmatrix}. \tag{6}$$

Here, $\mathbf{M}$ is the $3n \times 3n$ nodal mass matrix for the soft body mesh, $\mathbf{v}$ contains the $3n$ nodal velocities that preserve the global momentum, λ contains 6 Lagrange multipliers, and $\mathbf{J}$ is $6 \times 3n$ matrix where each 6×3 block is given by

$$\mathbf{J}_i = \begin{pmatrix} m_i\mathbf{I}_3 \\ m_i\mathbf{r}_i^* \end{pmatrix} \tag{7}$$

where $\mathbf{I}_3$ is the 3×3 identity matrix, $\mathbf{r}_i$ is the vector from the body's center of mass to node i, and $\mathbf{r}^*$ is the cross product matrix.

Taking the Schurr Compliment to remove the Lagrange multipliers we have,

$$\mathbf{v} = \hat{\mathbf{v}} + \mathbf{M}^{-1}\mathbf{J}^T \left(\mathbf{J}\mathbf{M}^{-1}\mathbf{J}^T\right)^{-1} (\boldsymbol{\Psi} - \mathbf{J}\hat{\mathbf{v}}). \tag{8}$$

Unfortunately, this matrix equation obscures what is intuitively very straightforward. $\mathbf{J}\hat{\mathbf{v}}$ gives the length 6 vector of linear and angular momenta of the soft body mesh. Subtracting this from the rigid momenta ψ gives the required correction. The 6×6 matrix $\left(\mathbf{J}\mathbf{M}^{-1}\mathbf{J}^T\right)^{-1}$ simplifies to

$$\left(\mathbf{J}\mathbf{M}^{-1}\mathbf{J}^T\right)^{-1} = \begin{pmatrix} \frac{1}{m}\mathbf{I}_3 & \mathbf{0} \\ \mathbf{0} & \mathbf{I}^{-1} \end{pmatrix}, \tag{9}$$

where m and $\mathbf{I}$ are the mass and inertia tensor for the body. Thus, this matrix simply converts the momentum correction to a velocity correction. $\mathbf{M}^{-1}\mathbf{J}^T$ is a $3n \times 6$ matrix of where each 3×6 block is given by

$$\begin{pmatrix} \mathbf{I}_3 \\ \mathbf{r}_i^{*T} \end{pmatrix}, \tag{10}$$

which maps the velocity correction to the soft body nodes.

We note that while Algorithm 2 seems entirely reasonable, it was not immediately obvious to us. For example, we did not anticipate that momentum would be converted to velocity in the 6-dimensional rigid body space. We also note that another view of our approach is that we are projecting out the rigid degrees of freedom from the soft body velocity and replacing them with our alternatively tracked values.