

Projective Affine Body Dynamics for Multibody Systems

Zimeng Ye^{1,2}, Xiaowei He^{†1}, Yuzhong Guo¹, Yin Yang³, Chenfanfu Jiang⁴ and Hongan Wang^{1,2}

¹Institute of Software, Chinese Academy of Sciences, China

²University of Chinese Academy of Sciences, China

³The University of Utah, USA

⁴University of California, Los Angeles, USA



Figure 1: Robotic dog. Our method offers a real-time simulation solution for multibody systems with intricate joints and contact. The simulation of a quadrupedal robotic dog walking on sandy terrain achieves a frame rate of 56 FPS.

Abstract

Multibody systems have widespread applications in diverse fields such as robotics, entertainment, and animation. Their capability to model and simulate intricate interactions among interconnected bodies makes them invaluable in various domains. However, complexity arises with factors like non-smooth contact handling, nonlinearity in flexible joints, and parallelization challenges. We introduce a stable and highly parallel GPU algorithm within affine body dynamics for solving constrained multibody dynamics with nonlinear constraints. Our innovation involves reformulating constrained multibody dynamics into a variational form, treating the system as a set of affine bodies connected with peridynamic bonds. This formulation provides a unified model for affine body dynamics, constraints, and contact within the peridynamics framework. It also facilitates the integration of the semi-implicit successive substitution method to solve nonlinear optimization in a global-local iterative manner. The proposed method obviates the necessity of assembling a global Hessian, rendering it highly suitable for efficient implementation on GPUs. This allows real-time simulation of intricate interactions involving both rigid and flexible bodies, encompassing contact, joint constraints, and friction.

CCS Concepts

• **Computing methodologies** → **Physical simulation**;

1. Introduction

Multibody systems manifest two distinctive features: system components undergoing finite motions and constraints imposing re-

strictions on the relative motions. Modern tools for multibody dynamics analysis have been developed to address both linear and nonlinear constraints with arbitrary topologies [Bau11, DZX*20, PT23]. Nevertheless, the situation can become significantly complex when considering additional factors, including non-smooth contact handling [MEM*19a, AEF22], nonlinearity in flexible bod-

[†] Corresponding: xiaowei@iscas.ac.cn

ies [DZX*20], and the parallelization of algorithms to leverage modern computing architectures [NSMH14, TNA10].

The prevailing method for solving constrained multibody dynamics typically begins with formulating the equations of motion with degrees of freedoms (DOFs) for rigid bodies. Subsequently, an additional step involves deriving constraint forces, consolidating the problem into a Linear Complementarity Problem (LCP) [ATK17]. Although formulating multibody dynamics as an LCP problem reduces the numerical implementation complexity, it does not adequately address our requirements for modeling complex real-world scenarios. For instance, both the contact and joint constraints may demonstrate non-smoothness and nonlinearity. To address non-smoothness in contact handling, Chen et al. [CLL*22] introduced Incremental Potential Contact (IPC) into multibody dynamics to convert the contact problem into a nonlinear optimization problem; therefore, a unified Newton barrier method can be used to solve complex multibody systems involving contact, friction and articulation constraints. However, Newton's method encounters performance bottlenecks when solving problems involving nonlinear constraints. Note that as the contacts are actively changing during the simulation, the time-consuming construction of the global Hessian needs to be updated frequently. Moreover, the storage pattern of the Hessian matrix becomes unpredictable due to the influence of active contacts on the values of non-zero off-diagonal elements. This complexity poses challenges for achieving an efficient GPU implementation.

This work aims to introduce a GPU-friendly, highly parallel method for addressing the nonlinear optimization problem in constrained multibody dynamics within affine body dynamics. We conceptualize the multibody system as a set of affine bodies subject to rigidity constraints, which allows the use of a projective local/global alternating solver. This procedure exhibits similarities to the solver proposed by Li et al. [LLK19]. Notably, a significant distinction lies in our formulation of the global energy function, expressed in terms of affine coordinates rather than the reduced degrees of freedom (DOFs) of rigid bodies. This distinctive approach enables the modeling of constraints as peridynamic bonds, facilitating the integration of the semi-implicit successive substitution method (SISSM) [LHG*23] to address nonlinear terms in the global step. The major contributions of this work include:

- A projective method for solving nonlinear optimization problems that is Hessian-free and relies solely on the first-order derivatives of the objective function;
- A novel method to calculate the inverse of the 12×12 block diagonal matrix, facilitating the development of a highly GPU-compatible algorithm for constrained multibody dynamics;
- A method to recover angular velocities of rigid bodies from affine body dynamics by motivating its relationship to both rigid body dynamics and deformable body dynamics.

2. Related Work

2.1. Multibody Dynamics

Multibody dynamics is an old subject that focuses on the analysis and simulation of the dynamic behavior of interconnected rigid or flexible bodies. While its study can be traced back to the end

of the 19th century, it was not modernized until the 1960s when computers were developed [Bre08]. Addressing the intricacies of highly complex mechanical systems, which encompass numerous interconnected rigid or deformable bodies with joints, contacts, and friction, poses several challenges. These challenges encompass aspects such as system representation [Bar96, FM17, GPB*19], computational costs [MSF20, WWB*19], stability during large time-steps [MMC*20], and more [DJB12].

In pure rigid body dynamics, early works simply use impulses to prevent interpenetration, yet stability is only guaranteed with some ad hoc treatments, such as velocity thresholding [MC95]. Guendelman et al. [GBF03] introduced an innovative time integration scheme, eliminating the requirement for ad hoc threshold velocities. Additionally, they proposed a novel shock propagation method that enhances the efficiency and visual accuracy in the stacking of objects. Kaufman et al. [KEP05] proposed to employ a novel contact model that uses mass, location, and velocity information at the moment of maximum compression to reduce the sensitivity of rigid body behaviors to variations in contact configuration. Catto [Cat05] presented a method for caching contact forces that allows contact points to move from step to step. This temporal coherence helps amortize the cost of computing accurate contact forces over several frames. Erleben [Erl07] formulated multibody dynamics as a velocity-based complementarity problem and used an iterative projected Gauss-Seidel (PGS) solver to achieve improvements in both performance and quality. With the advancement of modern GPUs, Tonge et al. [TBV12] expedited the method further by parallelizing each iteration and addressing contacts in blocks. This approach yields wall clock performance comparable to PGS while mitigating visible artifacts. Macklin et al. [MEM*19b] expanded the solution approach to address nonlinear complementarity problems using a non-smooth Newton iteration. This extension enables the support of nonlinear dynamic models, encompassing hyperelastic deformable bodies and articulated rigid mechanisms.

To guarantee intersection-free configurations at every time step, Ferguson et al. [FLS*21] borrowed ideas from continuum mechanics and proposed to introduce the incremental potential contact framework to reduced coordinates and rigid body dynamics for time integration and contact response. Lan et al. [LKL*22] introduced an affine body dynamics (ABD) framework that treats rigid bodies as affine bodies, thus enabling the simulation of both stiff and soft materials within a unified framework. With the assistance of ABD, Chen et al. [CLL*22] introduced a simulation framework for multibody dynamics employing universal variational integration. This framework facilitates the coupling of rigid and soft bodies interconnected by both linear and nonlinear constraints. Recently, Giles et al. [GDY25] use the augmented Lagrangian formulation to enhance vertex block descent, therefore achieving superior performance compared to the state-of-the-art alternatives.

2.2. Projective Dynamics

The inception of projective dynamics can be traced to position-based dynamics [MHHR07, BMO*14], where force-based actions are replaced with position-based constraints. While the general position-based approach proves robust in modeling diverse physical phenomena, including elasticity [MC11], contact, friction [MMCK14], and fluid incompressibility [MM13], its reliability

ity is hindered by sensitivity to numerical parameters such as the simulation time step and iteration count [MMC16].

Liu et al. [LBOK13] led the effort to reframe the implicit Euler integration method as an energy minimization problem, solving the linear system globally and addressing all nonlinear terms locally. Bouaziz et al. [BML*14] formalized this method by introducing a distance measure to bridge the global and local solvers. Consequently, the global system matrix remains constant and can thus be prefactored during initialization, enabling highly efficient global solves throughout the runtime. Overby et al. [OBLN17] further interpreted projective dynamics from the perspective of the alternating direction method of multipliers (ADMM), extending the framework to more general hyperelastic models and dynamic constraints.

While projective dynamics inherently exhibits nonlinear characteristics, its convergence behavior closely resembles that of an iterative method employed to solve linear systems. Wang [Wan15] therefore proposed to use the Chebyshev approach to accelerate the convergence rate. While earlier methodologies, such as position-based dynamics or projective dynamics, cater to a restricted range of materials, Liu et al. [LBK17] redefined projective dynamics as a quasi-Newton method. They suggested enhancing the convergence speed by integrating the quasi-Newton method with L-BFGS. He et al. [HWW18] introduced projective dynamics into peridynamics to model versatile elastoplastic materials with different dimensions. To improve the robustness of projective optimization under large rotational motions, Brown and Narain [BN21] proposed a rotation-aware ADMM formulation with adaptive weighting strategies. Recent work has also explored improved treatments of rotational degrees of freedom. Romanyà-Serrasolsas et al. [RSCO25] introduced a differentiable rotation dynamics formulation based on Lie algebra representations, enabling efficient computation of rotational derivatives.

Regrettably, the inherent projective nature of the proposed simulator necessitates the definition of the elastic constitutive model in a specific form, and the methodology for modeling general hyperelastic materials remains elusive. In the context of projective dynamics, effectively modeling general hyperelastic materials involves the incorporation of hyperelastic energy functions into the overarching solution. Lu et al. [LHG*23] have consequently presented the semi-implicit successive substitution method (SISSM) to address hyperelastic energy functions in the global solve. The adaptability afforded by employing SISSM to incorporate nonlinear terms into the global step not only facilitates a GPU-friendly implementation, but also expands the range of possibilities for projective dynamics in addressing nonlinear optimization problems.

3. Foundations

Following the standard ABD by Lan et al. [LKL*22], the kinematics of an affine body is described with a time-varying linear transform $\mathbf{A}(t) \in \mathbb{R}^{3 \times 3}$ and a translation $\mathbf{c}(t) \in \mathbb{R}^3$. The world coordinates of a material point ξ in body i are given as an affine map

$$\mathbf{y}_{i\xi} = \mathbf{A}_i \mathbf{x}_{i\xi} + \mathbf{c}_i = \mathbf{J}_{i\xi} \mathbf{q}_i, \quad (1)$$

where $\mathbf{x}_{i\xi}$ are the material coordinates defined in body i 's frame, $\mathbf{q} = \left(\mathbf{c}^T, \mathbf{a}_1^T, \mathbf{a}_2^T, \mathbf{a}_3^T \right)^T \in \mathbb{R}^{12}$ with $\mathbf{A} = [\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3]^T$ represent the

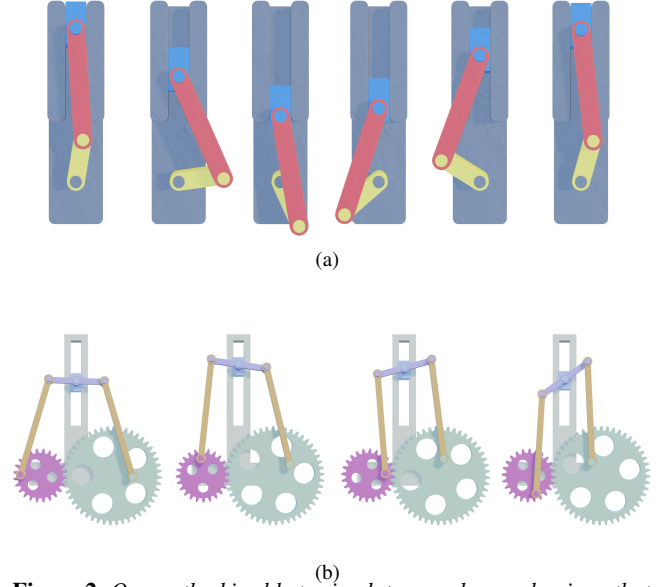


Figure 2: Our method is able to simulate complex mechanisms that involve different kinds of joints, contact and friction.

generalized coordinates of body i , and $\mathbf{J}_{i\xi} = [\mathbf{I}_3, \mathbf{I}_3 \otimes \mathbf{x}_{i\xi}]$ denotes the Jacobian with $\mathbf{I}_3$ representing the 3×3 identity matrix. To rigidify each affine body, a stiff orthogonality potential

$$V_{\perp}(\mathbf{q}) = \kappa \left\| \mathbf{A} \mathbf{A}^T - \mathbf{I} \right\|_F^2 \quad (2)$$

is further defined to restore rigid body's rotational DOFs with energy minimization, where κ is a constant used to control rigidity.

When each affine body is simulated independently, i.e., no contacts or constraints are considered, the global Hessian for the whole system is simply a 12×12 block diagonal matrix, therefore its implementation can be fully parallelized with GPU acceleration. However, when contacts or constraints are included, off-diagonal terms will be introduced into the global Hessian, and the unpredictable pattern of the global Hessian can greatly complicate the parallelization as contacts or constraints are actively updated during the simulation.

To alleviate the computational cost, Lan and colleagues [LKL*22] proposed to utilize the i-AABBs to speed up the global Hessian assembly. However, it is still not possible to estimate and pre-allocate adequate space to store the corresponding 12×12 off-diagonal blocks due to the sparsity of collision detection. For a GPU friendly implementation, it is preferred that the optimal memory cost is a linear function of the number of DOFs and whose upper limit can be estimated at the beginning of simulation. Therefore, the objective of this work is to develop a matrix free solver that can model contact, friction, and joints simultaneously and run parallelly on GPU.

4. Projective Affine Body Dynamics

To battle above mentioned challenges, let us first revisit the optimization problem for multibody dynamics, then give our solution.

Table 1 summarizes all notations to facilitate the following discussion.

Table 1: Notations.

Notation	Description
h	Time step.
$\mathbf{q}$	Generalized coordinates of affine bodies.
$\mathbf{c}$	Translational coordinates of affine bodies.
$\mathbf{A}$	Affine matrix of affine bodies
$\mathbf{J}$	Jacobian of affine bodies.
$\mathbf{M}$	Generalized mass matrix of affine bodies.
$\mathbf{H}$	12×12 coefficient matrix for each affine body.
Π	Affine moment of inertia.
$\mathbf{v}$	Translational velocity of rigid bodies.
ω	Angular velocity of rigid bodies.
$[\omega]_{\times}$	Skew-symmetric matrix of ω .
$\mathbf{R}$	Rotation matrix of rigid bodies.
Ψ	Potential energy of a bond.
$\dot{\Psi}$	Time derivative of Ψ with respect to d .
$\mathbf{f}$	Generalized external force.
$\mathbf{y}$	World coordinates of contact (constraint) points.
$\mathbf{x}$	Material coordinates of contact (constraint) points.
$\mathbf{n}$	Normal vector along a bond.
$\mathbf{Q}$	Quaternion of rigid bodies.

4.1. Motivation

Assume a multibody system is modeled as a set of N rigid bodies connected by joints to restrict their relative motion. The dynamics of a rigid body is uniquely determined by its generalized position $\mathbf{q}$, velocity $\dot{\mathbf{q}}$ and acceleration $\ddot{\mathbf{q}}$. Each joint is then defined as functions of the generalized coordinates that can impose from zero to six constraints on the relative motion of a pair of rigid bodies. Note if the constraint is a linear function of the generalized coordinates, its time derivative can be uniformly expressed as a product of the Jacobian matrix and the generalized velocity [Cha13].

Modeling multibody dynamics becomes challenging when introducing nonlinear constraints, such as incorporating contact potential. Our pivotal approach involves conceptualizing a multibody system as a set of affine bodies connected by peridynamic bonds. These bonds subsequently impose additional forces on each affine body, and the equation of motion for affine bodies could possibly be written as [UK92, LKL*22]

$$\mathbf{M}\ddot{\mathbf{q}} = -\sum_{\xi} \nabla \Psi_{\xi}(\mathbf{q}) + \mathbf{f}, \quad (3)$$

where $\mathbf{q} \in \mathbb{R}^{12N}$ represents generalized coordinates of affine bodies, $\mathbf{f} \in \mathbb{R}^{12N}$ represents the generalized external force, and Ψ_{ξ} represents the bond energy connected to the material point ξ . Following the update rule of an implicit time integration [BML*14]

$$\begin{aligned} \mathbf{q}^{n+1} &= \mathbf{q}^n + h\dot{\mathbf{q}}^{n+1} \\ \dot{\mathbf{q}}^{n+1} &= \dot{\mathbf{q}}^n + h\mathbf{M}^{-1} \left(-\sum_{\xi} \nabla \Psi_{\xi}(\mathbf{q}^{n+1}) + \mathbf{f} \right), \end{aligned} \quad (4)$$

solving the above equation of motion for affine bodies is equivalent

to solving the following optimization problem

$$\arg \min_{\mathbf{q}^{n+1}} \frac{1}{2h^2} \left\| \mathbf{M}^{\frac{1}{2}} (\mathbf{q}^{n+1} - \mathbf{s}^n) \right\|_F^2 + \sum_{\xi} \Psi_{\xi}(\mathbf{q}^{n+1}), \quad (5)$$

where $\mathbf{s}^n = \mathbf{q}^n + h\dot{\mathbf{q}} + h^2\mathbf{M}^{-1}\mathbf{f}$. Before detailing the solution to the above optimization problem, we first derive several relationships that are fundamental to our method.

4.1.1. The relationship to rigid body dynamics

Theorem 1 If $\mathbf{f}_{i\xi}$ is the concentrated force imposed on affine body i , then Newton's second law in terms of affine coordinates is formulated as

$$\mathbf{M}_i \ddot{\mathbf{q}}_i = \sum_{\xi} \mathbf{J}_{i\xi}^T \mathbf{f}_{i\xi}, \quad (6)$$

where given the mass density distribution function ρ , the generalized mass matrix $\mathbf{M}$ is 12×12 block diagonal

$$\mathbf{M}_i = \begin{bmatrix} m_i \mathbf{I} & 0 & 0 & 0 \\ 0 & \Pi_i & 0 & 0 \\ 0 & 0 & \Pi_i & 0 \\ 0 & 0 & 0 & \Pi_i \end{bmatrix}, \quad \Pi_i = \int_{\Omega} \rho \mathbf{x}_i \mathbf{x}_i^T dV. \quad (7)$$

In addition, we define Π_i as the *affine moment of inertia*.

Proof Given concentrated forces $\mathbf{f}_{i\xi}$, we have the force density distribution function $\mathbf{f}(\mathbf{x}') = \sum_{\xi} \mathbf{f}_{i\xi}(\mathbf{x}') = \sum_{\xi} \mathbf{f}_{i\xi} \delta(\mathbf{x} - \mathbf{x}')$, where $\delta(\mathbf{x} - \mathbf{x}')$ is the Dirac delta function. By applying Newton's second law, we have

$$\begin{aligned} \rho(\mathbf{x}') \ddot{\mathbf{y}}(\mathbf{x}') &= \mathbf{f}(\mathbf{x}') \\ \int_{\Omega} \rho(\mathbf{x}') \ddot{\mathbf{y}}(\mathbf{x}') d\Omega' &= \int_{\Omega} \mathbf{J}^T(\mathbf{x}') \mathbf{f}(\mathbf{x}') d\Omega' \\ \int_{\Omega} \rho(\mathbf{x}') \mathbf{J}^T(\mathbf{x}') \mathbf{J}(\mathbf{x}') \ddot{\mathbf{q}}_i d\Omega' &= \sum_{\xi} \int_{\Omega} \mathbf{J}^T(\mathbf{x}') \mathbf{f}_{i\xi}(\mathbf{x}') d\Omega' \\ \mathbf{M}_i \ddot{\mathbf{q}}_i &= \sum_{\xi} \mathbf{J}_{i\xi}^T \mathbf{f}_{i\xi} \end{aligned} \quad (8)$$

Thus, the formulation for $\mathbf{M}_i$ and Π_i can be derived from the above derivation. $\square$

A deeper understanding of affine body dynamics can be achieved by further examining the properties of the rotation group $\text{SO}(3)$. For any $\mathbf{R}(t) \in \text{SO}(3)$, it is well known that its time derivative is given by

$$\dot{\mathbf{R}}(t) = [\omega]_{\times} \mathbf{R}(t), \quad [\omega]_{\times} = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix} \quad (9)$$

where ω is the angular velocity expressed in the world frame, and $[\omega]_{\times}$ denotes its associated skew-symmetric matrix. Note that the time derivative of the rotation matrix depends not only on the angular velocity but also on the current orientation of the rotation matrix itself. The following lemma presents a similar relationship between the affine matrix $\mathbf{A}$ and the angular velocity ω .

Lemma 1 If an affine body is under a rigid transformation, we have $\dot{\mathbf{A}} = [\omega]_{\times} \mathbf{A}$.

Proof According to polar decomposition, the affine matrix has a factorization of the form $\mathbf{A} = \mathbf{R}\mathbf{S}$, where $\mathbf{R}$ is the rotation matrix and $\mathbf{S}$ is a positive semi-definite Hermitian matrix. When an affine body undergoes a rotation transformation, the matrix $\mathbf{S}$ remains unchanged, we thus have

$$\dot{\mathbf{A}} = \dot{\mathbf{R}}\mathbf{S} = [\omega] \times \mathbf{R}\mathbf{S} = [\omega] \times \mathbf{A}$$

□

4.1.2. The relationship to deformable body dynamics

Theorem 2 Let $\phi(\mathbf{x})$ be the energy density function defined over the material space of affine body i , then Newton's second law in terms of affine coordinates is formulated as

$$\mathbf{M}_i \ddot{\mathbf{q}}_i = - \int_{\Omega} \nabla_{\mathbf{q}_i} \phi(\mathbf{x}) d\Omega. \quad (10)$$

Proof According to the constitutive relation in continuum mechanics, the elastic force is formulated as $\mathbf{f}(\mathbf{x}) = - \left[\frac{\partial \phi(\mathbf{x})}{\partial \mathbf{y}} \right]^T$. By taking the following chain rule

$$-\nabla_{\mathbf{q}_i} \phi = \left[-\frac{\partial \phi}{\partial \mathbf{y}} \cdot \frac{\partial \mathbf{y}}{\partial \mathbf{q}_i} \right]^T = \mathbf{J}^T(\mathbf{x}) \mathbf{f}(\mathbf{x}),$$

and replace the right hand side of Equation 8, we can verify theorem 4.3. □

Lemma 2 If $\Psi_{\xi}(\mathbf{q})$ is the potential energy of a bond connected to affine bodies, we have

$$\mathbf{M}_i \ddot{\mathbf{q}}_i = \sum_{\xi} -\nabla_{\mathbf{q}_i} \Psi_{\xi}(\mathbf{q}) \quad (11)$$

Proof The above lemma can be verified by defining the energy density function as $\phi(\mathbf{x}') = \sum_{\xi} \Psi_{\xi} \delta(\mathbf{x} - \mathbf{x}')$, and substitute $\phi(\mathbf{x}')$ into Equation 10. □

Lemma 3 (transformation of affine moment of inertia) After affine transformation, the current affine moment of inertia is updated to $\Pi_i = \mathbf{A}_i \Pi_0 \mathbf{A}_i^T$, where Π_0 represents the initial value calculated in the material space.

Proof According to the definition of Π_i , it follows

$$\begin{aligned} \Pi_i &= \int_{\Omega} \rho \mathbf{x}_i \mathbf{x}_i^T d\Omega \\ &= \int_{\Omega} \rho (\mathbf{A}_i \mathbf{x}_0) (\mathbf{A}_i \mathbf{x}_0)^T d\Omega \\ &= \mathbf{A}_i \left(\int_{\Omega} \rho \mathbf{x}_0 \mathbf{x}_0^T d\Omega \right) \mathbf{A}_i^T \\ &= \mathbf{A}_i \Pi_0 \mathbf{A}_i^T \end{aligned}$$

□

4.2. Projective Semi-implicit Solver

For simplicity, we uniformly refer to the bond associated with the material point ξ as bond ξ for following discussions. Consider an arbitrary bond ξ that connects affine bodies i and j . The two contact points in world-space coordinates are given as follows:

$$\begin{aligned} \mathbf{y}_{i\xi} &= \mathbf{A}_i \mathbf{x}_{i\xi} + \mathbf{c}_i = \mathbf{J}_{i\xi} \mathbf{q}_i \\ \mathbf{y}_{j\xi} &= \mathbf{A}_j \mathbf{x}_{j\xi} + \mathbf{c}_j = \mathbf{J}_{j\xi} \mathbf{q}_j \end{aligned} \quad (12)$$

Assuming that the energy of bond ξ is a function of $d = |\mathbf{y}_{i\xi} - \mathbf{y}_{j\xi}|$, the dynamics of the affine multibody system can be formulated as the following optimization problem:

$$\arg \min_{\mathbf{q}} \frac{1}{2h^2} \left\| \mathbf{M}^{\frac{1}{2}} (\mathbf{q} - \mathbf{s}^n) \right\|_F^2 + \sum_{\xi} \Psi_{\xi}(d), \quad s.t. \quad V_{\perp}(\mathbf{q}) = 0, \quad (13)$$

where the function $V_{\perp}(\mathbf{q}) = 0$ ensures the rigidity of each affine body. Motivated by projected Jacobi or Gauss-Seidel that are commonly used in rigid body dynamics [TBV12], our approach to solving the constrained optimization problem is divided into two steps. In the global step, we temporarily ignore the constraints of $V_{\perp}(\mathbf{q}) = 0$ and apply the semi-implicit successive substitution method [HLG*25, LHG*23] to solve the following nonlinear optimization problem:

$$\mathbf{q}^{k+\frac{1}{2}} = \arg \min_{\mathbf{q}} \frac{1}{2h^2} \left\| \mathbf{M}^{\frac{1}{2}} (\mathbf{q} - \mathbf{s}^n) \right\|_F^2 + \sum_{\xi} \Psi_{\xi}(d). \quad (14)$$

Then, in the local step, we project $\mathbf{q}^{k+\frac{1}{2}}$ to the nearest feasible point to independently solve the following optimization problem for each affine body

$$\mathbf{q}^{k+1} = \arg \min_{\mathbf{p}} \frac{1}{2} \left\| \mathbf{q}^{k+\frac{1}{2}} - \mathbf{p} \right\|_F^2, \quad s.t. \quad V_{\perp}(\mathbf{p}) = 0. \quad (15)$$

The two steps above are alternately iterated until a user-defined threshold is met or the maximum number of iterations is reached.

4.2.1. Global Solve

Minimizing Eq. 14 is equivalent to finding a solution that satisfies the following equations

$$\mathbf{M}_i (\mathbf{q}_i - \mathbf{s}_i^n) = h^2 \sum_{\xi} \underbrace{\frac{\varphi_{\xi}(d)}{d} \mathbf{J}_{i\xi}^T (\mathbf{J}_{j\xi} \mathbf{q}_j - \mathbf{J}_{i\xi} \mathbf{q}_i)}_{-\frac{\partial \Psi_{\xi}}{\partial \mathbf{q}_i}}, \quad (16)$$

where φ_{ξ} represents the derivative of Ψ_{ξ} with respect to d , $\mathbf{q}_i$ and $\mathbf{q}_j$ represents the generalized coordinates of affine body i and j , respectively. Note that $\varphi_{\xi}(d)$ may include nonlinear terms, which typically necessitate time-consuming Newton-type algorithms with line-search filtering [FLS*21] to ensure robust convergence. However, constructing the global Hessian at each Newton iteration is computationally expensive on both multi-core CPUs and GPGPUs, particularly when the number of bonds significantly exceeds the number of affine bodies [LKL*22]. To address this challenge, we develop a Hessian-free substitution-type algorithm, following the approach proposed by He et al. [HLG*25]. The core idea is to separate the potential force for each bond $-\partial \Psi_{\xi} / \partial \mathbf{q}_i$ into the summation of a positive and a negative part

$$-\frac{\partial \Psi_{\xi}}{\partial \mathbf{q}_i} = \frac{\varphi_{\xi}^{+}(d)}{d} \mathbf{J}_{i\xi}^T (\mathbf{J}_{j\xi} \mathbf{q}_j - \mathbf{J}_{i\xi} \mathbf{q}_i) + \frac{\varphi_{\xi}^{-}(d)}{d} \mathbf{J}_{i\xi}^T (\mathbf{J}_{j\xi} \mathbf{q}_j - \mathbf{J}_{i\xi} \mathbf{q}_i), \quad (17)$$

where φ_{ξ}^{+} indicates $\varphi_{\xi}^{+} \geq 0$ for any d while $\varphi_{\xi}^{-} \leq 0$. At k -th substitution, we treat the positive part implicitly while the negative part

explicitly, therefore the potential force can be linearized as

$$-\left(\frac{\partial \Psi_{\xi}}{\partial \mathbf{q}_i}\right)^{k+\frac{1}{2}} = \frac{\varphi_{\xi}^{+}(d^k)}{d^k} \mathbf{J}_{i\xi}^T \left(\mathbf{J}_{j\xi} \mathbf{q}_j^{k+\frac{1}{2}} - \mathbf{J}_{i\xi} \mathbf{q}_i^{k+\frac{1}{2}} \right) + \frac{\varphi_{\xi}^{-}(d^k)}{d^k} \mathbf{J}_{i\xi}^T \left(\mathbf{J}_{j\xi} \mathbf{q}_j^k - \mathbf{J}_{i\xi} \mathbf{q}_i^k \right). \quad (18)$$

Inserting Eq. (18) into the secant equation of the global energy function, we are able to reformulate Eq. 16 as

$$\mathbf{M}_i \left(\mathbf{q}_i^{k+\frac{1}{2}} - \mathbf{s}_i^n \right) = h^2 \sum_{\xi} \frac{\varphi_{\xi}^{+}(d^k)}{d^k} \mathbf{J}_{i\xi}^T \left(\mathbf{J}_{j\xi} \mathbf{q}_j^{k+\frac{1}{2}} - \mathbf{J}_{i\xi} \mathbf{q}_i^{k+\frac{1}{2}} \right) + h^2 \sum_{\xi} \frac{\varphi_{\xi}^{-}(d^k)}{d^k} \mathbf{J}_{i\xi}^T \left(\mathbf{J}_{j\xi} \mathbf{q}_j^k - \mathbf{J}_{i\xi} \mathbf{q}_i^k \right), \quad (19)$$

which yields a square system of $12N$ linear equations. Analogous to the Jacobi iteration, by decomposing the coefficient matrix into its diagonal and off-diagonal components, the body-wise update for each affine body i is given by:

$$\mathbf{q}_i^{k+\frac{1}{2}} = \left(\mathbf{H}_i^k \right)^{-1} \left(\mathbf{M}_i \mathbf{s}_i^n + h^2 \sum_{\xi} \varphi_{\xi}^{-}(d^k) \mathbf{J}_{i\xi}^T \mathbf{n}_{\xi}^k + h^2 \sum_{\xi} \frac{\varphi_{\xi}^{+}(d^k)}{d^k} \mathbf{J}_{i\xi}^T \mathbf{y}_{j\xi}^k \right), \quad (20)$$

where $\mathbf{H}_i^k$ denotes a symmetric, positive-definite 12×12 matrix as

$$\mathbf{H}_i = \mathbf{M}_i + h^2 \sum_{\xi} \frac{\varphi_{\xi}^{+}(d)}{d} \mathbf{J}_{i\xi}^T \mathbf{J}_{i\xi}, \quad (21)$$

$\mathbf{n}_{\xi}$ denotes the normal vector pointing from $\mathbf{y}_{i\xi}$ to $\mathbf{y}_{j\xi}$

$$\mathbf{n}_{\xi} = \frac{(\mathbf{J}_{j\xi} \mathbf{q}_j - \mathbf{J}_{i\xi} \mathbf{q}_i)}{d}. \quad (22)$$

It is important to note that Eq. 20 now conforms to a fixed-point iteration framework, where each update involves only a separable 12×12 matrix for each affine body i . This property enables our algorithm to be fully compatible with modern GPU architectures, as each affine body can be updated independently.

4.2.2. Local Solve

In the local step, the purpose is to rigidify each affine body by solving the following constrained optimization problem

$$\mathbf{q}^{k+1} = \arg \min_{\mathbf{p}} \frac{1}{2} \left\| \mathbf{q}_i^{k+\frac{1}{2}} - \mathbf{p} \right\|_F^2, \quad s.t. \quad V_{\perp}(\mathbf{p}) = 0. \quad (23)$$

Since the translational component of $\mathbf{q}_i^{k+\frac{1}{2}}$ is independent of the linear transformation and is unconstrained, the translation can be directly set as $\mathbf{c}_i^{k+1} = \mathbf{c}_i^{k+\frac{1}{2}}$. To solve the rotational component, the constrained optimization problem can possibly be reformulated as an unconstrained problem given by:

$$Q(\mathbf{A}_i : \kappa) = \frac{1}{2} \left\| \mathbf{A}_i - \mathbf{A}_i^{k+\frac{1}{2}} \right\|_F^2 + \kappa \left\| \mathbf{A}_i \mathbf{A}_i^T - \mathbf{I} \right\|_F^2, \quad (24)$$

where κ can also be regarded as the Lagrange multiplier. As the value of κ increases, the affine body is expected to become progressively stiffer. However, it is important to note that the optimization problem remains ill-conditioned when κ approaches extremely large values to simulate perfectly rigid bodies. Consider, for instance, the special case where $\kappa = \infty$. In this scenario, minimizing $Q(\mathbf{A}_i : \kappa)$ is equivalent to minimizing $\left\| \mathbf{A}_i \mathbf{A}_i^T - \mathbf{I} \right\|_F^2$. However,

since $\left\| \mathbf{A}_i \mathbf{A}_i^T - \mathbf{I} \right\|_F^2$ admits an infinite number of local minimizers, identifying a unique minimizer for $Q(\mathbf{A}_i : \kappa)$ becomes infeasible, which may lead to instability in the simulation. To address this issue, we apply a polar decomposition to $\mathbf{A}_i^{k+\frac{1}{2}} = \mathbf{R}_i \mathbf{S}_i$, which is a rotation $\mathbf{R}_i$ followed by positive semi-definite symmetric matrix $\mathbf{S}_i$, and directly set $\mathbf{A}_i^{k+1} = \mathbf{R}_i$. The resulting solution is subsequently substituted into Eq. 20 for the next global computation.

4.3. A sparse representation for $\mathbf{H}_i$

One issue that remains unsolved is how to efficiently calculate the inverse of the 12×12 matrix $\mathbf{H}_i$ on GPU for each global step. According to GPU specifics, the total number of registers per Streaming Multiprocessor (SM) is usually limited and many modern GPUs have a hard limit of 255 registers per thread [NVI26]. Therefore, a dense representation for $\mathbf{H}_i$ can not only quickly consume up the limited number of registers for each GPU thread, but also incur a lot of inefficiency in calculating its inverse. For example, directly calculating the inverse of a 12×12 matrix using LU decomposition may require at least 144 registers.

To address this problem, let us revisit the structure of $\mathbf{J}$ in its matrix form

$$\mathbf{J} = \begin{bmatrix} 1 & 0 & 0 & x & y & z & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & x & y & z & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & x & y & z \end{bmatrix}. \quad (25)$$

By exploiting its sparse nature, the 12×12 matrix $\mathbf{H}_i$ can be simplified to

$$\mathbf{H}_i = \begin{bmatrix} a_i \mathbf{I} & \mathbf{e}_x \mathbf{b}_i^T & \mathbf{e}_y \mathbf{b}_i^T & \mathbf{e}_z \mathbf{b}_i^T \\ \mathbf{b}_i \mathbf{e}_x^T & \mathbf{D}_i & 0 & 0 \\ \mathbf{b}_i \mathbf{e}_y^T & 0 & \mathbf{D}_i & 0 \\ \mathbf{b}_i \mathbf{e}_z^T & 0 & 0 & \mathbf{D}_i \end{bmatrix}, \quad (26)$$

where $\mathbf{e}_x$, $\mathbf{e}_y$ and $\mathbf{e}_z$ are the unit vectors along the x-, y- and z-axis, a_i , $\mathbf{b}_i$ and $\mathbf{D}_i$ represent a 1×1 scalar, a 3×1 vector and a 3×3 matrix, whose formula are written as follows

$$a_i = m_i + h^2 \sum_{\xi} \frac{\varphi_{i\xi}^{+}(d)}{d}, \quad \mathbf{b}_i = h^2 \sum_{\xi} \frac{\varphi_{i\xi}^{+}(d)}{d} \mathbf{x}_{i\xi}, \\ \mathbf{D}_i = \Pi_i + h^2 \sum_{\xi} \frac{\varphi_{i\xi}^{+}(d)}{d} \mathbf{x}_{i\xi} \mathbf{x}_{i\xi}^T.$$

Thus, it can be noted that we only need to store a_i , $\mathbf{b}_i$ and $\mathbf{D}_i$ rather than the whole matrix, which helps reduce the required register number from 144 to 13. More importantly, the above formulation also greatly simplifies the procedure in calculating the inverse of

$\mathbf{H}_i$. By invoking the following relationship [HJ12]

$$\begin{bmatrix} \mathbf{I} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix}^{-1} = \begin{bmatrix} \mathbf{I} + \mathbf{B}(\mathbf{D} - \mathbf{CB})^{-1}\mathbf{C} & -\mathbf{B}(\mathbf{D} - \mathbf{CB})^{-1} \\ -(\mathbf{D} - \mathbf{CB})^{-1}\mathbf{C} & (\mathbf{D} - \mathbf{CB})^{-1} \end{bmatrix}$$

where $\mathbf{I}$ can be any identity matrix while $\mathbf{B} = \mathbf{C}^T$, and $\mathbf{D}$ represent another square matrix. After some simple algebraic operations, the inverse of $\mathbf{H}_i$ has the same structure as $\mathbf{H}_i$, which is written as

$$\mathbf{H}_i^{-1} = \begin{bmatrix} (a_i^{-1} + \tilde{a}_i)\mathbf{I} & -\mathbf{e}_x \tilde{\mathbf{b}}_i^T \tilde{\mathbf{D}}_i^{-1} & -\mathbf{e}_y \tilde{\mathbf{b}}_i^T \tilde{\mathbf{D}}_i^{-1} & -\mathbf{e}_z \tilde{\mathbf{b}}_i^T \tilde{\mathbf{D}}_i^{-1} \\ -\tilde{\mathbf{D}}_i^{-1} \tilde{\mathbf{b}}_i \mathbf{e}_x^T & \tilde{\mathbf{D}}_i^{-1} & 0 & 0 \\ -\tilde{\mathbf{D}}_i^{-1} \tilde{\mathbf{b}}_i \mathbf{e}_y^T & 0 & \tilde{\mathbf{D}}_i^{-1} & 0 \\ -\tilde{\mathbf{D}}_i^{-1} \tilde{\mathbf{b}}_i \mathbf{e}_z^T & 0 & 0 & \tilde{\mathbf{D}}_i^{-1} \end{bmatrix}, \quad (27)$$

where $\tilde{a}_i$, $\tilde{\mathbf{b}}_i$ and $\tilde{\mathbf{D}}_i$ are calculated as follows

$$\tilde{a}_i = \tilde{\mathbf{b}}_i^T \tilde{\mathbf{D}}_i^{-1} \tilde{\mathbf{b}}_i, \quad \tilde{\mathbf{b}}_i = \frac{\mathbf{b}_i}{a_i}, \quad \tilde{\mathbf{D}}_i = \mathbf{D}_i - \frac{\mathbf{b}_i \mathbf{b}_i^T}{a_i}.$$

Notice the above formulation of $\mathbf{H}_i$ greatly simplifies the inverse calculation. Furthermore, it is GPU-friendly due to a reduction of the required number of registers.

4.4. Update translational and angular velocities of rigid bodies

Given the solution to the optimization problem, updating the translational velocities of rigid bodies is straightforward, as was typically done in position-based dynamics [MHHR07]. To compute the angular velocities of rigid bodies, Lemma 1 can be applied. Reformulating Lemma 1 to $[\omega]_{\times} = \dot{\mathbf{A}}\mathbf{A}^{-1}$ allows the angular velocity components ω_x , ω_y and ω_z to be directly derived from the off-diagonal entries of $[\omega]_{\times}$. To further reduce floating point errors, the final value of ω is obtained from the average of $[\omega]_{\times}$ and $-[\omega]_{\times}^T$, i.e., $([\omega]_{\times} - [\omega]_{\times}^T)/2$. Algorithm 1 summarizes the complete procedure of our method. The rotational component of $\mathbf{q}$ is obtained from the quaternion using the standard quaternion-to-matrix conversion [Sho85].

5. Results and Discussion

All our algorithms involved in this section are implemented using C++ and CUDA, and the execution of all examples is carried out on an NVIDIA RTX 3080 GPU. Collision detection is accelerated with linear BVH [Kar12]. Coulomb friction is implemented based on [AEF22]. Table 2 shows statistics for major examples.

5.1. Unit Tests

To support multibody dynamics with contact, we follow Macklin et al. [MEM*19a] to implement both the soft and hard constraint functions, and then use a different combination of constraints to model different kinds of joints. The details on the mathematical description on both the constraint functions and joints are given in the supplementary material for completeness. Please also refer to the video to view the demonstration of three basic joints modeled with our method, include the spherical joint, hinge joint and slider joint.

Algorithm 1 Projective Affine Body Dynamics

```

1: while simulating do
2:   Do collision detection;
3:   for affine body  $i$  do
4:      $\mathbf{v}_i^* \leftarrow \mathbf{v}_i^n + h\mathbf{f}_{ext}/m_i$ ;
5:      $\mathbf{c}_i^* \leftarrow \mathbf{c}_i^n + h\mathbf{v}_i^*$ ;
6:      $\mathbf{Q}_i^* \leftarrow \mathbf{Q}_i^n + \frac{h}{2}[\omega_x, \omega_y, \omega_z, 0]\mathbf{Q}_i^n$ ;
7:      $\mathbf{q}_i^* \leftarrow (\mathbf{c}_i^*, \mathbf{Q}_i^*)$ ;
8:      $\mathbf{q}_{i=0}^* \leftarrow \mathbf{q}_i^*$ 
9:   end for
10:  for  $k < n_{max}$  do
11:    for affine body  $i$  do
12:       $\mathbf{q}_i^{k+\frac{1}{2}} \leftarrow$  solve Equation 20; //Global solve
13:       $\mathbf{R}_i \leftarrow \mathbf{A}_i^{k+\frac{1}{2}}$ ; //Local solve
14:       $\mathbf{q}_i^{k+1} \leftarrow (\mathbf{c}_i^{k+\frac{1}{2}}, \mathbf{R}_i)$ ;
15:    end for
16:  end for
17:  for affine body  $i$  do
18:     $\mathbf{q}_i^{n+1} \leftarrow \mathbf{q}_i^{n_{max}}$ ;
19:     $\mathbf{v}_i^{n+1} = (\mathbf{c}_i^{n+1} - \mathbf{c}_i^n)/h$ ;
20:     $\omega_i^{n+1} \leftarrow (\mathbf{A}_i^{n+1} - \mathbf{A}_i^n)(\mathbf{A}_i^n)^{-1}/h$ ;
21:  end for
22: end while

```

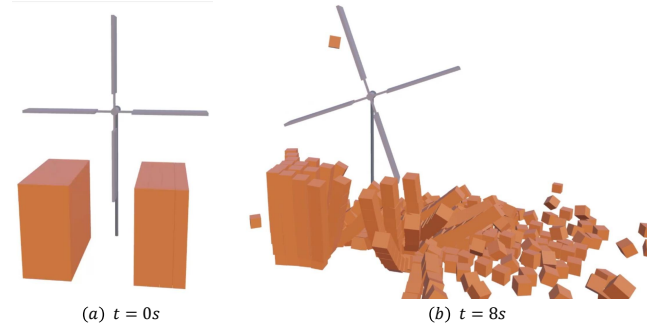


Figure 3: Simulation of a windmill colliding two piles of boxes.

5.1.0.1. Joints Figure 2 illustrates the method's capability to simulate complex mechanisms involving various types of joints, contact, and friction. Figure 2(top) illustrates a mechanism composed of four rigid bodies that are connected together by three hinge joints and a slider joint. Figure 2(bottom) illustrates a mechanism composed of a driven gear (yellow) and a free gear (grey) that are connected together by a sliding linkage. The collision between gears is modeled using a compound shape, with a capsule positioned at the front of each gear.

5.1.0.2. Contact and Friction Figure 3 shows the ability of our method in simulating a windmill colliding two piles of boxes. The windmill is composed of a fixed stand, four wheel blades connected to the fixed stand with driven hinge joint. Notice our method can stably solve contact and friction for large number of rigid bodies.

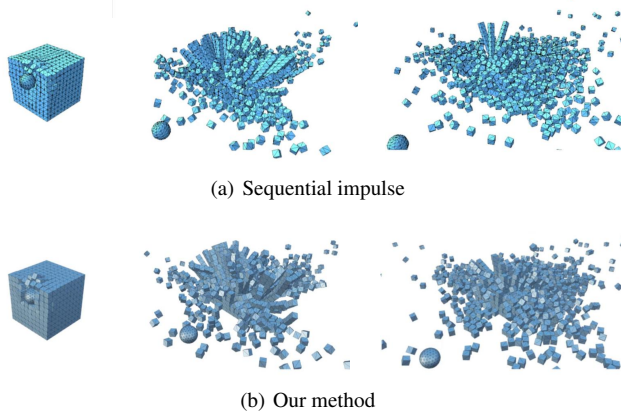


Figure 4: A ball hitting a stack of boxes. Our method is able to generate consistent results compared to the sequence impulse method.

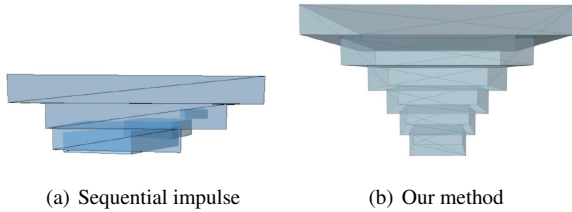


Figure 5: Simulation of a stack of boxes where each box has ten times the mass of the one below it. PhysX results in an unstable simulation while our method handles this scenario stably.

Compared to the Newton barrier method [CLL*22], our method achieves a speedup of more than one order.

5.2. Comparison to other methods

5.2.1. Comparison to the sequential impulse method

Figure 4 presents a comparison between our method and the sequential impulse method [Cat05] implemented in PhysX under identical simulation conditions. Both methods yield comparable results and accurately capture contact and frictional interactions. However, because our method has not yet been fully optimized, it exhibits slightly lower computational performance: it runs at approximately 40 fps, whereas the sequential impulse method achieves approximately 70 fps. Despite this, the sequential impulse method encounters difficulties when dealing with extreme mass ratios. As shown in Figure 5, which depicts a stack of six boxes where each box has ten times the mass of the one below it, the sequential impulse method results in an unstable simulation. In contrast, our method handles this scenario stably.

5.2.2. Comparison to the AVBD method

Figure 6 first demonstrates a simple example using a spring connecting 2 boxes. To be fair, both methods apply the following quadratic energy potential

$$\Psi_j = \frac{1}{2} \kappa_j (C_j(x))^2, \quad (28)$$

where κ_j represents the stiffness, $C_j = |d - l|$ describes the elongation of the constraint which has a rest length of l . Its semi-implicit split of the first order derivative is given in Eq.15 by He

et al. [HLG*25]. When the spring is initially stretched to two times of the rest length, the convergence rate for the first time step is plotted in Figure 6(a). Note that AVBD can obtain the exact solution in a single iteration due to the quadratic convergence of Newton's method. In contrast, the semi-implicit successive substitution method exhibits only linear convergence; consequently, our method requires multiple iterations to achieve a sufficiently accurate solution. Figure 6(b) uses the scenario in Fig. 7, which is composed of an 11×11 grid of boxes connected by 220 springs, where each box is connected to its four nearest neighbors, and all springs are initially stretched toward the center-bottom of the scene. Since the variable stiffness introduced by the augmented Lagrangian strategy in AVBD would increase the total potential energy, we keep all spring stiffnesses in both our method and AVBD identical. As a result, AVBD does not exploit any adaptivity and effectively reduces to standard VBD in this experiment. As can be seen from the figure, our method exhibits slightly faster convergence than VBD during the first few iterations, while both methods eventually converge to similar results.

Figure 8 further presents a comparison between Augmented Projective Affine Body Dynamics (Augmented-PABD) and Augmented Vertex Block Descent (AVBD) in enforcing hard constraints using the following augmented Lagrangian formulation

$$\Psi_j = \frac{1}{2} \kappa_j (C_j(x))^2 + \lambda_j C_j(x), \quad (29)$$

where λ is the Lagrange multiplier. At the beginning of the simulation, we initialize the variables for both methods using $\kappa_j^0 = \kappa_{start}$ and $\lambda_j = 0$, where $\kappa_{start} = 10000$. In subsequent iterations, the primal and dual variables are updated following the augmented Lagrangian method in AVBD [GDY25]

$$\begin{aligned} \kappa_j^{k+1} &= \kappa_j^k + \beta \left| C_j^k(x) \right|, \\ \lambda_j^{k+1} &= \kappa_j^k C_j^k(x) + \lambda_j^k, \end{aligned} \quad (30)$$

where a scaling parameter β is set to a constant of 10000. Meanwhile, we apply the warm starting strategy for both the stiffness and dual variables at every simulation step

$$\kappa_j^0 = \max(\gamma \kappa_j^n, \kappa_{start}) \quad \text{and} \quad \lambda_j^0 = \alpha \gamma \lambda_j^t, \quad (31)$$

where $\alpha = 0.99$ and $\gamma = 0.999$. According to the comparison, it can be noticed our method is slightly better in enforcing hard constraints.

Finally, Figure 9(a) simulates a chain of boxes with a large mass ratio using our method, both with and without the augmented Lagrangian formulation in AVBD [GDY25]. When the augmented Lagrangian formulation is not applied, the distance constraints are not well preserved due to the large mass of the bottom box. However, our method produces results comparable to those of AVBD when the augmented Lagrangian formulation is applied.

5.3. More examples

5.3.1. Bridge

Figure 10 illustrates another jeep using the same suspension to cross a bridge. The bridge is composed of 50 rigid boards, lined together with point connections.

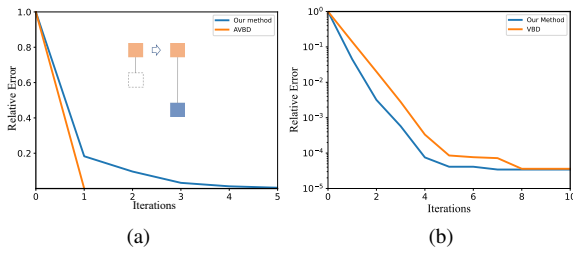


Figure 6: Comparison of convergence between our method and AVBD. (a) When only one rigid box is simulated, the convergence rate of AVBD is faster than our method. (b) In the scenario shown in Fig. 7, our method exhibits slightly faster convergence in the early iterations, but the convergence behavior eventually becomes similar. The relative error is defined in the same way as in [BML*14].

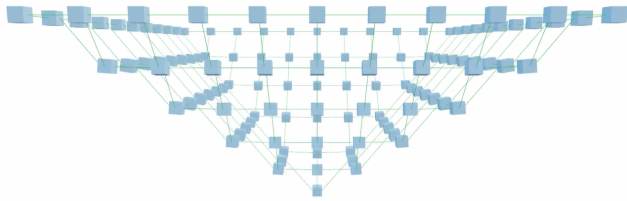


Figure 7: This scenario consists of an 11×11 grid of boxes, where each box is connected to its four neighboring boxes by springs, forming 220 spring constraints in total. Initially, the springs connecting each box to its four neighbors are pre-stretched toward the middle-bottom of the scene.

5.3.2. Coupling with Water

Figure 11 depicts a sailboat navigating the ocean, where the ocean is modeled with Gerstner waves [T*01] and the wake behind the sailboat is simulated with the standard shallow water equation [GNMFP*19]. The interaction between the boat and the ocean is modeled using a weak coupling strategy, in which the boat's buoyancy is computed directly. This example illustrates the robustness of rigid-deformable coupling, demonstrating efficient real-time simulation capabilities.

5.3.3. Coupling with Sand

Figure 1 shows a real-time simulation of a quadrupedal robotic dog walking on sandy terrain. The sandy terrain is simulated using shallow sand equations derived from a height field with a resolution of 256×256 [ZHL*21]. A straightforward periodic motion is applied to the rotation angles of the hinge joints, enabling the dog to move forward under the influence of external forces. The interaction between the dog and sand is similarly modeled using a weak coupling strategy as above. Figure 12 further demonstrates a larger scale of simulation, where a total of 28 jeeps as well as the sand are simulated in real time.

5.3.4. Large-scale stacking and collision

Figure 13 presents balls impacting large-scale $50 \times 50 \times 50$ block piles. Due to the lack of complete performance optimization, the average simulation frame rate is approximately 5 fps.

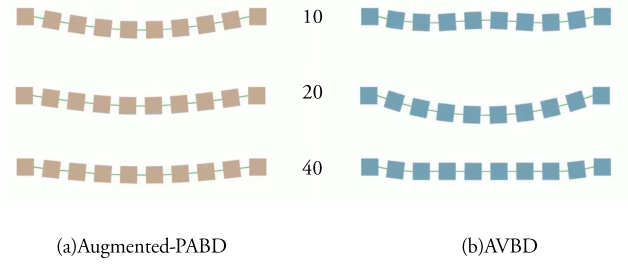


Figure 8: A chain of boxes. In simulating a chain of rigid boxes with hard constraints, Augmented-PABD performs slightly better than AVBD in settling down the simulation.

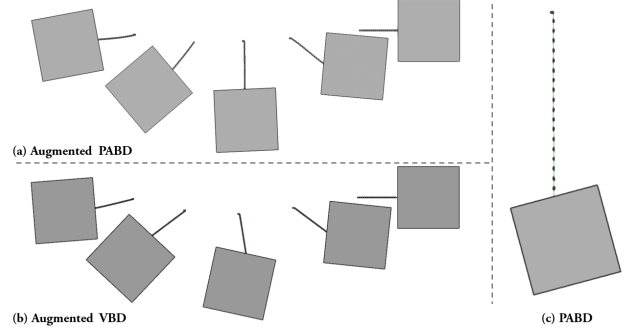


Figure 9: Comparison between our method and AVBD in simulating a chain of boxes with a large mass ratio. Our method produces results (a) comparable to those of AVBD (b). In contrast, when augmented Lagrangian is not applied, the distance constraints are not well enforced (c).

6. Conclusion

We have developed projective affine body dynamics, a stable and highly parallel GPU algorithm within affine body dynamics, for solving constrained multibody dynamics with nonlinear constraints. Our innovation centers on the reformulation of constrained multibody dynamics into a variational framework, wherein the system is regarded as a set of affine bodies connected by peridynamic bonds. This formulation facilitates the integration of the semi-implicit successive substitution method to address nonlinear terms in the global step and enables a Hessian-free real-time solver for multibody dynamics involving nonlinear joint constraints, contact, and friction.

Our method has several limitations. First, it shares the same limitation as the original ABD: simulations can fail under extreme conditions, such as when the rotation between two successive iterations is large. Second, in our approach to handling nonlinear equality constraints with polynomial barrier functions, the decomposition of energy into positive and negative components becomes increasingly complex at higher polynomial orders. Finally, how to extend our method to more complex scenarios involving fluids and deformable objects remains unclear, we aim to address these challenges in future work.

Acknowledgements

We thank the anonymous reviewers for constructive comments and Xukun Luo for demo preparation. This work was supported

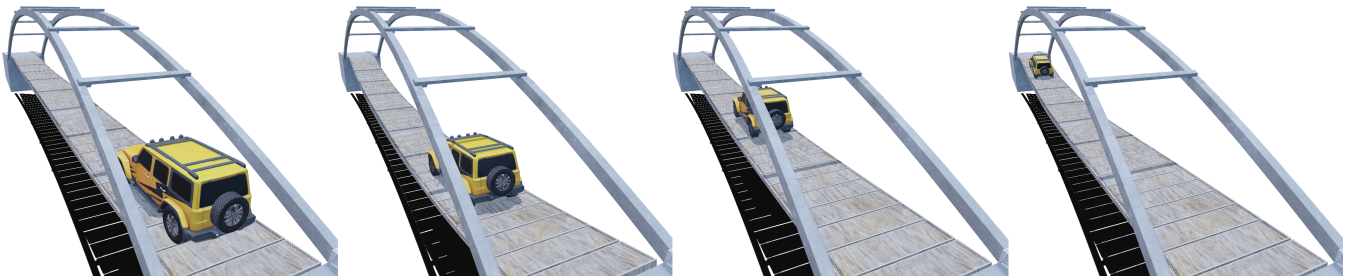


Figure 10: Bridge. A simulation of a jeep crossing a bridge is conducted, where the bridge comprises 50 rigid boards connected by spherical joints. The example runs at an average frame rate of 55 FPS.

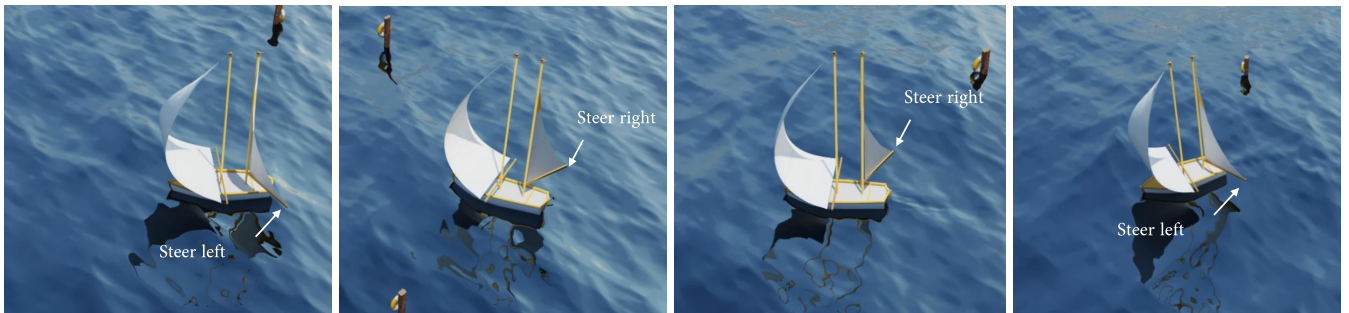


Figure 11: Sailboat. We simulate a sailboat comprised of a rigid body, two deformable sails, and a rigid horizontal bar connected to the sailboat body with a hinge joint to steer the sailboat. The whole simulate achieves an average rate of 42 FPS.

Table 2: Time statistics. N_{body} is the number of affine bodies, N_{joint} is the number of joints, N_{iter} is the iteration number, $t_{col}(ms)$ is computational cost for collision detection, $t_{solve}(ms)$ is the computational cost for our projective solver in solving the multibody system only. Computational costs for other materials as well as the coupling are not considered in this table.

Example	N_{body}	N_{joint}	N_{iter}	$t_{col}(ms)$	$t_{solve}(ms)$
Figure 1	9	8	20	2.22	2.40
Figure 2(a)	4	4	20	1.97	2.37
Figure 2(b)	8	7	20	1.26	2.17
Figure 3	515	5	40	1.87	4.89
Figure 4(b)	1001	0	20	3.15	2.65
Figure 5(b)	6	0	30	1.39	2.25
Figure 9(a)	19	18	10	0	1.28
Figure 9(c)	19	18	10	0	1.25
Figure 10	59	55	20	1.73	2.05
Figure 11	136	10	30	3.23	2.76
Figure 12	140	112	30	0.92	2.54
Figure 13(a)	125001	0	20	72.56	87.55
Figure 13(b)	125001	0	20	74.93	90.24

by the New Generation Artificial Intelligence-National Science and Technology Major Project of China (No. 2025ZD0123902), the National Natural Science Foundation of China (No.92570206, No.62302490), and the Basic Research Project of ISCAS (No. ISCAS-JCMS-202403).

References

- [AEF22] ANDREWS S., ERLEBEN K., FERGUSON Z.: Contact and friction simulation for computer graphics. In *ACM SIGGRAPH 2022 Courses*. 2022, pp. 1–172. [1](#), [7](#)
- [ATK17] ANDREWS S., TEICHMANN M., KRY P. G.: Geometric stiffness for real-time constrained multibody dynamics. In *Computer Graphics Forum* (2017), vol. 36, Wiley Online Library, pp. 235–246. [2](#)
- [Bar96] BARAFF D.: Linear-time dynamics using lagrange multipliers. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1996), SIGGRAPH '96, Association for Computing Machinery, p. 137–146. URL: <https://doi.org/10.1145/237170.237226>, doi:10.1145/237170.237226. [2](#)
- [Bau11] BAUCHAU O. A.: *Flexible multibody dynamics*, vol. 176. Springer, 2011. [1](#)
- [BML*14] BOUAZIZ S., MARTIN S., LIU T., KAVAN L., PAULY M.: Projective dynamics: Fusing constraint projections for fast simulation. *ACM Trans. Graph.* 33, 4 (jul 2014). URL: <https://doi.org/10.1145/2601097.2601116>, doi:10.1145/2601097.2601116. [3](#), [4](#), [9](#)
- [BMO*14] BENDER J., MÜLLER M., OTADUY M. A., TESCHNER M., MACKLIN M.: A survey on position-based simulation methods in computer graphics. In *Computer graphics forum* (2014), vol. 33, Wiley Online Library, pp. 228–251. [2](#)
- [BN21] BROWN G. E., NARAIN R.: Wrapd: Weighted rotation-aware admm for parameterization and deformation. *ACM Transactions on Graphics* 40, 4 (2021), 82:1–82:14. doi:10.1145/3450626.3459942. [3](#)
- [Bre08] BREMER H.: Elastic multibody dynamics. *Springer Science+Business Media, BV* 13 (2008), 85–90. [2](#)
- [Cat05] CATTO E.: Iterative dynamics with temporal coherence. In *Game developer conference* (2005), vol. 2. [2](#), [8](#)

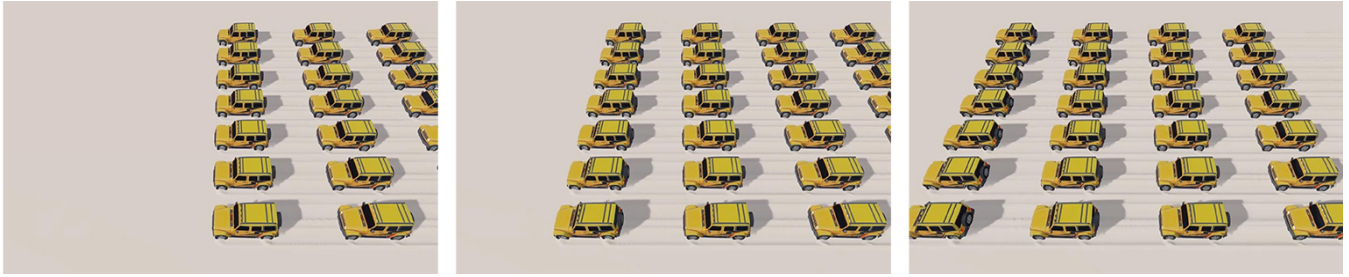


Figure 12: Wading of jeeps on sand. This scenario is screen-recorded, and the frame rate, accounting for both simulation and rendering, is approximately 40 FPS.

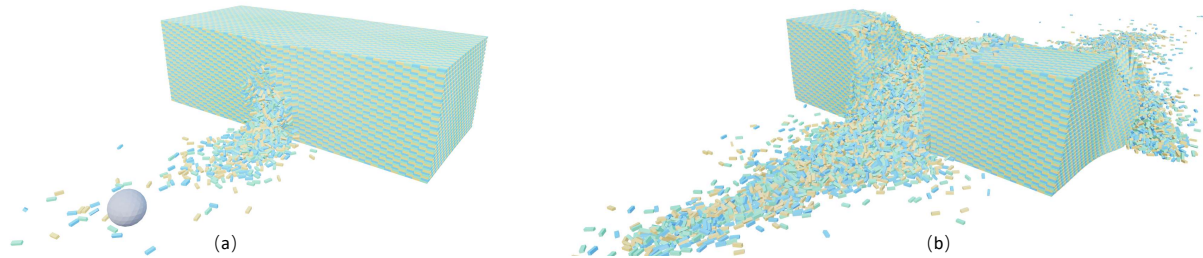


Figure 13: Impact of balls with different masses and velocities on a large-scale $50 \times 50 \times 50$ block pile: (a) smaller mass and lower velocity; (b) larger mass and higher velocity.

- [Cha13] CHAPPUIS D.: Constraints derivation for rigid body simulation in 3d, 2013. 4
- [CLL*22] CHEN Y., LI M., LAN L., SU H., YANG Y., JIANG C.: A unified newton barrier method for multibody dynamics. *ACM Trans. Graph.* 41, 4 (jul 2022). URL: <https://doi.org/10.1145/3528223.3530076>, doi:10.1145/3528223.3530076. 2, 8
- [DJB12] DE JALON J. G., BAYO E.: *Kinematic and dynamic simulation of multibody systems: the real-time challenge*. Springer Science & Business Media, 2012. 2
- [DZX*20] DENG Y., ZHAO Y., XU H., LIN F., WANG Q.: Rigid-flexible coupling modelling and dynamic performance analysis of novel flexible road wheel. *Proceedings of the Institution of Mechanical Engineers, Part K: Journal of Multi-body Dynamics* 234, 1 (2020), 67–81. 1, 2
- [Erl07] ERLEBEN K.: Velocity-based shock propagation for multibody dynamics animation. *ACM Trans. Graph.* 26, 2 (jun 2007), 12–es. URL: <https://doi.org/10.1145/1243980.1243986>, doi:10.1145/1243980.1243986. 2
- [FLS*21] FERGUSON Z., LI M., SCHNEIDER T., GIL-URETA F., LANGLOIS T., JIANG C., ZORIN D., KAUFMAN D. M., PANOZZO D.: Intersection-free rigid body dynamics. *ACM Trans. Graph.* 40, 4 (jul 2021). URL: <https://doi.org/10.1145/3450626.3459802>, doi:10.1145/3450626.3459802. 2, 5
- [FM17] FRÂNCU M., MOLDOVEANU F.: Unified simulation of rigid and flexible bodies using position based dynamics. *VRIPHYS* 49 (2017), 58. 2
- [GBF03] GUENDELMAN E., BRIDSON R., FEDKIW R.: Nonconvex rigid bodies with stacking. *ACM Trans. Graph.* 22, 3 (jul 2003), 871–878. URL: <https://doi.org/10.1145/882262.882358>, doi:10.1145/882262.882358. 2
- [GDY25] GILES C., DIAZ E., YUKSEL C.: Augmented vertex block descent. *ACM Trans. Graph.* 44, 4 (jul 2025). URL: <https://doi.org/10.1145/3731195>, doi:10.1145/3731195. 2, 8
- [GNMFP*19] GARCÍA-NAVARRO P., MURILLO J., FERNÁNDEZ-PATO J., ECHEVERRIBAR I., MORALES-HERNÁNDEZ M.: The shallow water equations and their application to realistic cases. *Environmental Fluid Mechanics* 19 (2019), 1235–1252. 9
- [GPB*19] GISSLER C., PEER A., BAND S., BENDER J., TESCHNER M.: Interlinked sph pressure solvers for strong fluid-rigid coupling. *ACM Trans. Graph.* 38, 1 (jan 2019). URL: <https://doi.org/10.1145/3284980>, doi:10.1145/3284980. 2
- [HJ12] HORN R. A., JOHNSON C. R.: *Matrix Analysis*. Cambridge university press, 2012. 7
- [HLG*25] HE X., LIU S., GUO Y., SHI J., QIAO Y.: A semi-implicit sph method for compressible and incompressible flows with improved convergence. In *Computer Graphics Forum* (2025), vol. 44, Wiley Online Library, p. e70043. 5, 8
- [HWW18] HE X., WANG H., WU E.: Projective peridynamics for modeling versatile elastoplastic materials. *IEEE Transactions on Visualization and Computer Graphics* 24, 9 (2018), 2589–2599. doi:10.1109/TVCG.2017.2755646. 3
- [Karl12] KARRAS T.: Maximizing parallelism in the construction of bvhs, octrees, and k-d trees. In *Proceedings of the Fourth ACM SIGGRAPH/Eurographics conference on High-Performance Graphics* (2012), pp. 33–37. 7
- [KEP05] KAUFMAN D. M., EDMUNDS T., PAI D. K.: Fast frictional dynamics for rigid bodies. *ACM Trans. Graph.* 24, 3 (jul 2005), 946–956. URL: <https://doi.org/10.1145/1073204.1073295>, doi:10.1145/1073204.1073295. 2
- [LBK17] LIU T., BOUAZIZ S., KAVAN L.: Quasi-newton methods for real-time simulation of hyperelastic materials. *ACM Trans. Graph.* 36, 3 (may 2017). URL: <https://doi.org/10.1145/2990496>, doi:10.1145/2990496. 3
- [LBOK13] LIU T., BARGTEIL A. W., O'BRIEN J. F., KAVAN L.: Fast simulation of mass-spring systems. *ACM Trans. Graph.* 32, 6 (nov 2013). URL: <https://doi.org/10.1145/2508363.2508406>, doi:10.1145/2508363.2508406. 3
- [LHG*23] LU Z., HE X., GUO Y., LIU X., WANG H.: Projective peridynamic modeling of hyperelastic membranes with contact. *IEEE Transactions on Visualization and Computer Graphics* (2023), 1–14. doi:10.1109/TVCG.2023.3271511. 2, 3, 5
- [LKL*22] LAN L., KAUFMAN D. M., LI M., JIANG C., YANG Y.: Affine body dynamics: Fast, stable and intersection-free simulation of stiff materials. *ACM Trans. Graph.* 41, 4 (jul 2022).

- URL: <https://doi.org/10.1145/3528223.3530064>, doi: 10.1145/3528223.3530064. 2, 3, 4, 5
- [LLK19] LI J., LIU T., KAVAN L.: Fast simulation of deformable characters with articulated skeletons in projective dynamics. In *Proceedings of the 18th Annual ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2019), pp. 1–10. 2
- [MC95] MIRTICH B., CANNY J.: Impulse-based simulation of rigid bodies. In *Proceedings of the 1995 symposium on Interactive 3D graphics* (1995), pp. 181–ff. 2
- [MC11] MÜLLER M., CHENTANEZ N.: Solid simulation with oriented particles. In *ACM SIGGRAPH 2011 Papers* (New York, NY, USA, 2011), SIGGRAPH '11, Association for Computing Machinery. URL: <https://doi.org/10.1145/1964921.1964987>, doi: 10.1145/1964921.1964987. 2
- [MEM*19a] MACKLIN M., ERLEBEN K., MÜLLER M., CHENTANEZ N., JESCHKE S., MAKOVYCHUK V.: Non-smooth newton methods for deformable multi-body dynamics. *ACM Trans. Graph.* 38, 5 (oct 2019). URL: <https://doi.org/10.1145/3338695>, doi:10.1145/3338695. 1, 7
- [MEM*19b] MACKLIN M., ERLEBEN K., MÜLLER M., CHENTANEZ N., JESCHKE S., MAKOVYCHUK V.: Non-smooth newton methods for deformable multi-body dynamics. *ACM Trans. Graph.* 38, 5 (oct 2019). URL: <https://doi.org/10.1145/3338695>, doi:10.1145/3338695. 2
- [MHHR07] MÜLLER M., HEIDELBERGER B., HENNIX M., RATCLIFF J.: Position based dynamics. *Journal of Visual Communication and Image Representation* 18, 2 (2007), 109–118. 2, 7
- [MM13] MACKLIN M., MÜLLER M.: Position based fluids. *ACM Trans. Graph.* 32, 4 (jul 2013). URL: <https://doi.org/10.1145/2461912.2461984>, doi:10.1145/2461912.2461984. 2
- [MMC16] MACKLIN M., MÜLLER M., CHENTANEZ N.: Xpbd: position-based simulation of compliant constrained dynamics. In *Proceedings of the 9th International Conference on Motion in Games* (2016), pp. 49–54. 3
- [MMC*20] MÜLLER M., MACKLIN M., CHENTANEZ N., JESCHKE S., KIM T.-Y.: Detailed rigid body simulation with extended position based dynamics. In *Computer graphics forum* (2020), vol. 39, Wiley Online Library, pp. 101–112. 2
- [MMCK14] MACKLIN M., MÜLLER M., CHENTANEZ N., KIM T.-Y.: Unified particle physics for real-time applications. *ACM Trans. Graph.* 33, 4 (jul 2014). URL: <https://doi.org/10.1145/2601097.2601152>, doi:10.1145/2601097.2601152. 2
- [MSF20] MUNAWAR A., SRISHANKAR N., FISCHER G. S.: An open-source framework for rapid development of interactive soft-body simulations for real-time training. In *2020 IEEE International Conference on Robotics and Automation (ICRA)* (2020), IEEE, pp. 6544–6550. 2
- [NSMH14] NEGRUT D., SERBAN R., MAZHAR H., HEYN T.: Parallel computing in multibody system dynamics: why, when, and how. *Journal of Computational and Nonlinear Dynamics* 9, 4 (2014), 041007. 2
- [NVI26] NVIDIA: Nvidia ampere gpu architecture tuning guide, 2026. URL: <https://docs.nvidia.com/cuda/ampere-tuning-guide/index.html>. 6
- [OBLN17] OVERBY M., BROWN G. E., LI J., NARAIN R.: Admm $\supseteq$ projective dynamics: Fast simulation of hyperelastic models with dynamic constraints. *IEEE Transactions on Visualization and Computer Graphics* 23, 10 (2017), 2222–2234. doi:10.1109/TVCG.2017.2730875. 3
- [PT23] PROBST T., TESCHNER M.: Monolithic friction and contact handling for rigid bodies and fluids using sph. In *Computer Graphics Forum* (2023), vol. 42, Wiley Online Library, pp. 155–179. 1
- [RSCO25] ROMANYÁ-SERRASOLSAS M., CASA FRANCA J. J., OTADUY M. A.: Painless differentiable rotation dynamics. *ACM Transactions on Graphics* 44, 4 (2025). doi:10.1145/3730944. 3
- [Sho85] SHOEMAKE K.: Animating rotation with quaternion curves. In *Proceedings of the 12th Annual Conference on Computer Graphics and Interactive Techniques* (1985), ACM, pp. 245–254. doi:10.1145/325334.325242. 7
- [T*01] TESSENDORF J., ET AL.: Simulating ocean water. *Simulating nature: realistic and interactive techniques. SIGGRAPH 1*, 2 (2001), 5. 9
- [TBV12] TONGE R., BENEVOLENSKI F., VOROSHILOV A.: Mass splitting for jitter-free parallel rigid body simulation. *ACM Trans. Graph.* 31, 4 (jul 2012). URL: <https://doi.org/10.1145/2185520.2185601>, doi:10.1145/2185520.2185601. 2, 5
- [TNA10] TASORA A., NEGRUT D., ANITESCU M.: Gpu-based parallel computing for the simulation of complex multibody systems with unilateral and bilateral constraints: an overview. *Multibody Dynamics: Computational Methods and Applications* (2010), 283–307. 2
- [UK92] UDWADIA F., KALABA R.: A new perspective on constrained motion. *Proceedings of the Royal Society of London. Series A: Mathematical and Physical Sciences* 439, 1906 (1992), 407–410. 4
- [Wan15] WANG H.: A chebyshev semi-iterative approach for accelerating projective and position-based dynamics. *ACM Trans. Graph.* 34, 6 (nov 2015). URL: <https://doi.org/10.1145/2816795.2818063>, doi:10.1145/2816795.2818063. 3
- [WWB*19] WANG Y., WEIDNER N. J., BAXTER M. A., HWANG Y., KAUFMAN D. M., SUEDA S.: Redmax: efficient & flexible approach for articulated dynamics. *ACM Trans. Graph.* 38, 4 (July 2019). URL: <https://doi.org/10.1145/3306346.3322952>, doi: 10.1145/3306346.3322952. 2
- [ZHL*21] ZHU K., HE X., LI S., WANG H., WANG G.: Shallow sand equations: Real-time height field simulation of dry granular flows. *IEEE Transactions on Visualization and Computer Graphics* 27, 3 (2021), 2073–2084. doi:10.1109/TVCG.2019.2944172. 9