Pacific Graphics 2026
Y. He, N. Thürey, and L. Liu
(Guest Editors)

COMPUTER GRAPHICS *forum*
Volume 45 (2026), Number 7

Supplementary Material: Projective Affine Body Dynamics for Multibody Systems

Zimeng Ye[1,2], Xiaowei He[1], Yuzhong Guo[1], Yin Yang[3], Chenfanfu Jiang[4] and Hongan Wang[1,2]

[1]Institute of Software, Chinese Academy of Sciences, China
[2]University of Chinese Academy of Sciences, China
[3]The University of Utah, USA
[4] University of California, Los Angeles, USA

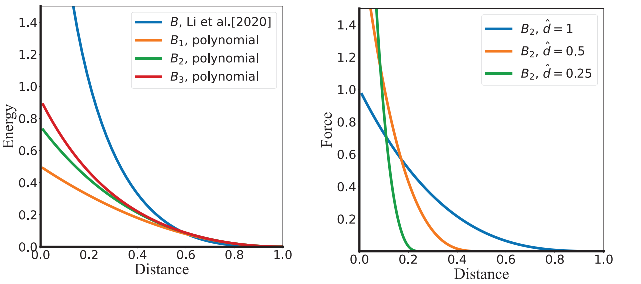

Figure 1: *Illustration of the polynomial barrier functions (left) as well as its first order derivative with varying $\hat{d}$ (right).*

1. Constraints

In this section, we provide a detailed formulation of the potential energy Ψ_ξ associated with each bond ξ. To facilitate a unified framework for modeling joints, contact, and friction, we classify constraints into two categories: soft kinematic constraints and hard kinematic constraints, drawing inspiration from Macklin et al. [MEM*19].

1.1. Soft Kinematic Constraints

To impose soft kinematic constraints, such as those modeled by a spring, a quadratic energy potential is defined based on the constraint function and a stiffness parameter κ

$$\Psi_{soft}(d) = \frac{1}{2}\kappa C^2(d:l). \tag{1}$$

where l represents the rest length, and $C(d:l) = d - l$ denotes the distance constraint [MHHR07]. Consequently, the positive and negative components are given by:

$$\frac{\varphi_{soft}{}^{+}(d)}{d} = \kappa,\quad \varphi^{-}_{soft}(d) = -\kappa l. \tag{2}$$

1.2. Hard Kinematic Constraints

Similarly, to impose hard kinematic constraints—such as gluing two points from different bodies—a straightforward approach is to apply D'Alembert's principle [MEM*19] or perform a change of variables [CLL*22]. To integrate effectively with SISSM, we adopt the barrier method to handle both hard inequality and bilateral constraints. Inspired by Li et al. [LFS*20], the hard kinematic constraint potential can possibly be formulated as

$$\Psi_{hard}(d) = \kappa B(d), \tag{3}$$

where the barrier function $B(\cdot)$ used in Li et al. [LFS*20] is defined as

$$B(d) = \begin{cases} -(d-\hat{d})^2 \log(\frac{d}{\hat{d}}), & 0 < d < \hat{d} \\ 0, & d \geq \hat{d} \end{cases}. \tag{4}$$

Here, $\hat{d}$ denotes a clamping threshold used to strictly prevent interpenetration. In real-time applications, prioritizing stability takes precedence, and minor interpenetration is generally deemed acceptable. The primary concern revolves around devising efficient mechanisms to swiftly rectify the interpenetration state without compromising the simulation's stability. Given that the barrier function employed in IPC becomes ineffective when the distance, denoted as d, is less than or equal to zero ($d \leq 0$), we opt for utilizing an approximate polynomial function. The log barrier function can be represented as an infinite sum of terms, each expressed in relation to the function's derivatives. By neglecting terms with odd exponents, the approximate polynomial function is defined as

$$B_N(d) = \begin{cases} \sum_{n=1}^{N} \frac{1}{n}\left(1-\frac{d}{\hat{d}}\right)^n, & 0 < d < \hat{d} \\ 0, & d \geq \hat{d} \end{cases}, \tag{5}$$

where N is the highest order. The positive and negative components are derived from the binomial theorem:

$$\begin{aligned} \frac{\varphi^{+}_{hard}}{d} &= \frac{1}{d^2}\sum_{n=1}^{\lfloor N/2 \rfloor} \binom{N}{2n} \left(\frac{d}{\hat{d}}\right)^{2n} \\ \varphi^{-}_{hard} &= -\frac{1}{d}\sum_{n=1}^{\lfloor N/2 \rfloor} \binom{N}{2n-1} \left(\frac{d}{\hat{d}}\right)^{2n-1} \end{aligned} \tag{6}$$

where $\lfloor N/2 \rfloor$ denotes the largest integer value not greater than $N/2$, $\begin{pmatrix} N \\ 2n \end{pmatrix}$ and $\begin{pmatrix} N \\ 2n-1 \end{pmatrix}$ represent binomial coefficients. Note that the denominators d^2 and d can be canceled, implying that both functions are smooth across the boundary at $d = 0$.

2. Joint Formulations

A joint is commonly defined as a set of constraints that limits the relative motion between two bodies. To constrain both the relative translation and rotation, a direct approach is to formulate a joint by defining all constraints as a combination of either position or rotation constraints. However, since ABD employs a linear affine transform for rotation, it is more convenient for us to specify position constraints rather than rotation constraints for joints. Consequently, we convert all rotation constraints into equivalent position constraints.

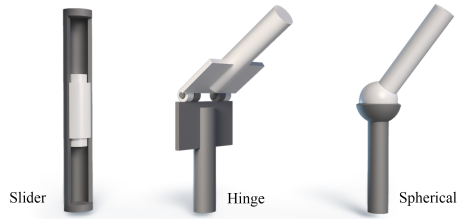

Figure 2: *Illustration of three lower pairs: slider, hinge and spherical joints.*

Given that the majority of joints employed in practical applications can be effectively modeled as so-called lower pairs [Bau11], our focus centers on three types of joints: the spherical, hinge, and slider joints.

2.1. Spherical Joint

To model an unconstrained spherical joint, a straightforward distance constraint is employed to connect two points from different objects

$$C_0(d:0) = \|\mathbf{y}_{i0} - \mathbf{y}_{j0}\| = 0, \tag{7}$$

where the constraint length l is set to zero. If we would like to impose additional rotational constraints, we should first glue $\mathbf{y}_{i0}$ and $\mathbf{y}_{j0}$ together, and then introduce more distance constraints to restrict the rotation. For simplicity, let us consider a two-dimensional case for example, and the purpose is to restrict the rotation angle between body i and j to be θ. We first calculate a new point $\mathbf{y}_{i1}$ on body i,

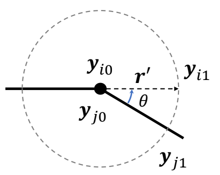

Figure 3: *Spherical.*

$$\mathbf{y}_{i1} = \mathbf{y}_{i0} + \mathrm{R}_i \mathbf{r}_1, \tag{8}$$

and then calculate the other new point $\mathbf{y}_{j1}$ from body j as follows

$$\mathbf{y}_{j1} = \mathbf{y}_{j0} + \mathrm{R}(\theta)\mathrm{R}_j \mathbf{r}_1, \tag{9}$$

where R is the rotation matrix. Thus, a spherical joint characterized by a joint angle of θ can be realized by employing the constraint presented in Eq. 7 in conjunction with the subsequent constraint

$$C_1(d:0) = \|\mathbf{y}_{i1} - \mathbf{y}_{j1}\| = 0. \tag{10}$$

In a three-dimensional space, to constrain all rotational DOFs, typically three distance constraints are sufficient—one for each axis. However, we observed that when a distance constraint is applied only on one side of $\mathbf{y}_{i0}$(or $\mathbf{y}_{j0}$), it could possibly lead to an asymmetric motion. Our solution is to employ two distance constraints for each rotational DOF, with one on each side but equidistant from $\mathbf{y}_i(\mathbf{y}_j)$. The introduction of redundant constraints not only aids in stabilizing spherical joints but also enhances the stability of hinge and slider joints.

2.2. Hinge Joint

An unconstrained hinge joint is modeled with two distance constraints along the rotation axis.

$$C_1(d:0) = \left\|\mathbf{y}_{i1} - \mathbf{y}_{j1}\right\| = 0, \;\; C_2(d:0) = \left\|\mathbf{y}_{i2} - \mathbf{y}_{j2}\right\| = 0 \tag{11}$$

To constrain the rotation angle, we select a new point $\mathbf{y}'_i$ from body i as

$$\mathbf{y}_{i0} = \frac{\mathbf{y}_{i1} + \mathbf{y}_{i2}}{2} + \mathrm{R}_i \mathbf{r}_0, \tag{12}$$

and calculate the other point from body j as

$$\mathbf{y}_{j0} = \frac{\mathbf{y}_{j1} + \mathbf{y}_{j2}}{2} + \mathrm{R}(\theta|\mathbf{a})\mathrm{R}_j \mathbf{r}_0, \tag{13}$$

where $\mathbf{a}$ is the rotation axis, θ is the rotation angle.

2.3. Slider Joint

A slider joint only allows relative translation between two bodies along a slider axis. We establish a slider joint utilizing two pairs of points, namely, $(\mathbf{y}_{i1}, \mathbf{y}_{j1})$ and $(\mathbf{y}_{i2}, \mathbf{y}_{j2})$, akin to the hinge joint. In the course of sliding, it is imperative to dynamically generate two distance constraints to confine the relative translation along the slider axis. The dynamically created points on body i is simply defined as

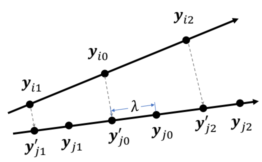

Figure 4: *Slider.*

$$\mathbf{y}'_{i1} = \mathbf{y}_{i1}, \;\; \mathbf{y}'_{i2} = \mathbf{y}_{i2}, \tag{14}$$

while the points on body j is calculated as the following projection

$$\begin{aligned} \mathbf{y}'_{j1} &= \mathbf{y}_{j1} + \lambda \mathbf{n}_j \\ \mathbf{y}'_{j2} &= \mathbf{y}_{j2} + \lambda \mathbf{n}_j \end{aligned}, \tag{15}$$

where $\mathbf{y}_{i0}$ is the middle point $\overline{\mathbf{y}_{i1}\mathbf{y}_{i2}}$ and $\mathbf{y}'_{j0}$ presents its projection on line $\overline{\mathbf{y}_{j1}\mathbf{y}_{j2}}$, $\mathbf{n}_j$ is the direction of line $\overline{\mathbf{y}_{j1}\mathbf{y}_{j2}}$, λ represents the offset distance of $\mathbf{y}'_{j0}$ from $\mathbf{y}_{j0}$ whose value can also be manually controlled.

A slider joint that allows both relative translation and twisting between two bodies is simply implemented by imposing the following two constraints

$$C_1(d:0) = \left\|\mathbf{y}'_{i1} - \mathbf{y}'_{j1}\right\| = 0, \;\; C_2(d:0) = \left\|\mathbf{y}'_{i2} - \mathbf{y}'_{j2}\right\| = 0. \tag{16}$$

To further limit the twisting angle, supplementary constraints can be imposed, akin to those employed in Eq. 12 and 13.

3. Contact and Friction

Similar to the original IPC formulation, each affine body is modeled with a thickness of $\hat{d}$, as illustrated in Figure 5. At the beginning of each simulation step, a discrete collision detection algorithm identifies contact pairs, with each contact point located on the affine body surface. For each contact pair, we model the contact potential using the hard $B_N(d)$, where d is defined to be the signed distance along the contact normal of affine body i

$$d = \left(\mathbf{y}_j - \mathbf{y}_i\right) \cdot \mathbf{n}_i. \tag{17}$$

Note that $\mathbf{y}_j - \mathbf{y}_i$ is initially aligned with the contact normal $\mathbf{n}_i$. However, as the state of the affine body is updated during each iteration, the direction of $\mathbf{y}_j - \mathbf{y}_i$ may deviate from $\mathbf{n}_i$. To avoid the computational cost of performing collision detection at every iteration, we continue to use the contact pairs identified at the beginning of each simulation step. The projection of $\mathbf{y}_j$ onto the normal vector is then used as the constraint point for affine body j, denoted as $\mathbf{y}_j^{\perp}$ in Figure 5.

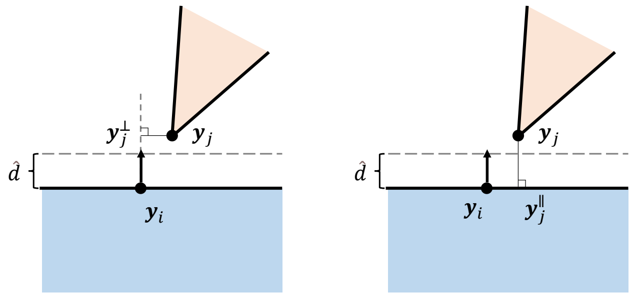

Figure 5: *Illustrate of a contact pair. Left: $\mathbf{y}_j^{\perp}$ represents the projection of $\mathbf{y}_i$ on the normal direction; Right: $\mathbf{y}_j^{\parallel}$ represents the projection of $\mathbf{y}_i$ on the tangential direction.*

We adopt the method proposed by Müller et al. [MMC*20] to model static friction. Specifically, we compute the relative motion between the contact pair along both the normal and tangential directions, given by $\|\mathbf{y}_j^{\perp} - \mathbf{y}_i\|$ and $\|\mathbf{y}_j^{\parallel} - \mathbf{y}_i\|$, respectively. If the tangential motion satisfies the condition $\|\mathbf{y}_j^{\parallel} - \mathbf{y}_i\| < \mu_s \|\mathbf{y}_j^{\perp} - \mathbf{y}_i\|$, where μ_s is the static friction coefficient, we enforce a distance constraint to set $\|\mathbf{y}_j^{\parallel} - \mathbf{y}_i\| = 0$.

4. Convergence

To evaluate whether the local projection perturbs the contact or joint configurations in the previous global step, we set up two boxes in Figure 6 and select a contact pair with an initial interpentraiton depth of 0.1. Both penetration-distance curves, before and after projecting the affine matrices onto SO(3), are plotted in Figure 7. The purpose of projecting each affine body back onto the SO(3) manifold is to obtain the converged translation and orientation of each affine body that satisfy all constraints, rather than to force the affine matrix to converge to a rotation matrix. Therefore, as the number of iterations increases, the local projection no longer perturbs the contact configuration once both the translation and orientation of the affine body have converged, even if the affine matrix itself has not converged to a rotation matrix.

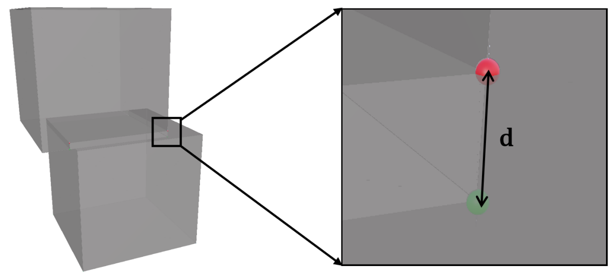

Figure 6: *Illustration of the two boxes and a contact pair that is selected to evaluate the influence of the local solve to the global solve.*

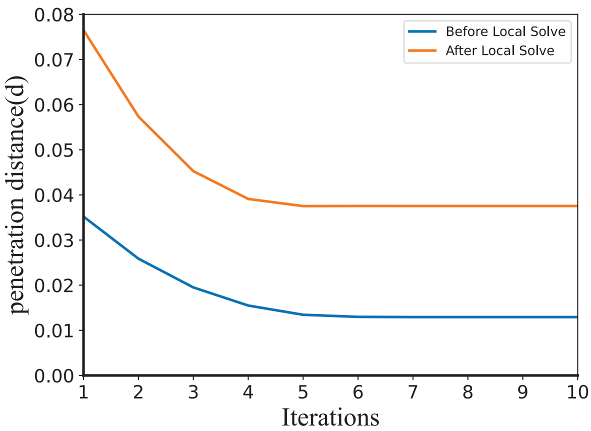

Figure 7: *Illustration of the penetration-distance curves before and after projecting the affine matrices onto SO(3).*

References

[Bau11] BAUCHAU O. A.: *Flexible multibody dynamics*, vol. 176. Springer, 2011. 2

[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. 1

[LFS*20] LI M., FERGUSON Z., SCHNEIDER T., LANGLOIS T., ZORIN D., PANOZZO D., JIANG C., KAUFMAN D. M.: Incremental potential contact: Intersection-and inversion-free, large-deformation dynamics. *ACM Trans. Graph. 39*, 4 (aug 2020). URL: https://doi.org/10.1145/3386569.3392425, doi:10.1145/3386569.3392425. 1

[MEM*19] MACKLIN M., ERLEBEN K., MÜLLER M., CHENTANEZ N., JESCHKE S., MAKOVIYCHUK 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

[MHHR07] MÜLLER M., HEIDELBERGER B., HENNIX M., RATCLIFF J.: Position based dynamics. *Journal of Visual Communication and Image Representation 18*, 2 (2007), 109–118. 1

[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. 3