Continuous Penalty Forces

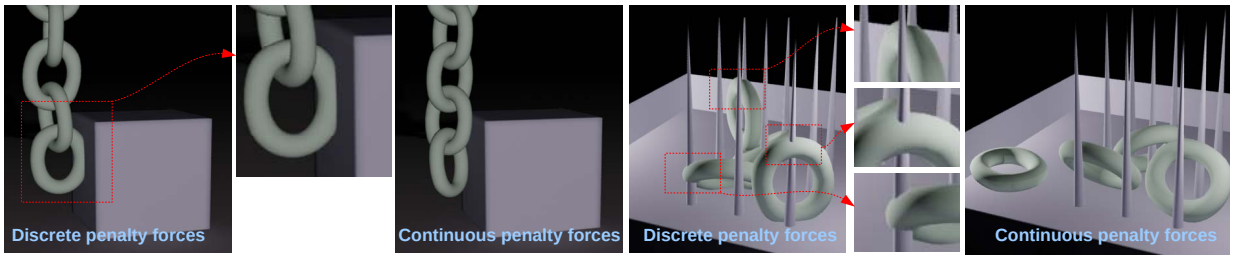
Min Tang*
State Key Lab of CAD&CG,
Zhejiang University

Dinesh Manocha†
University of North Carolina
at Chapel Hill

Miguel A. Otaduy‡
URJC Madrid

Ruofeng Tong§
State Key Lab of CAD&CG,
Zhejiang University

http://gamma.cs.unc.edu/CPF/



(a) Sticking in Chain Benchmark (b) Penetrations in Rings Benchmark

Figure 1: Sticking and penetration problems: *We highlight sticking and penetration problems in the Chain Benchmark (a)*($14.3k$ *triangles,* $60fps$*) and the Rings Benchmark (b)*($7.4k$ *triangles,* $35fps$*) with traditional penalty methods. Our novel continuous penalty force formulation can alleviate these problems based on continuous collision and force computation.*

Abstract

We present a simple algorithm to compute continuous penalty forces to determine collision response between rigid and deformable models bounded by triangle meshes. Our algorithm computes a well-behaved solution in contrast to the traditional stability and robustness problems of penalty methods, induced by force discontinuities. We trace contact features along their deforming trajectories and accumulate penalty forces along the penetration time intervals between the overlapping feature pairs. Moreover, we present a closed-form expression to compute the continuous and smooth collision response. Our method has very small additional overhead compared to previous penalty methods, and can significantly improve the stability and robustness. We highlight its benefits on several benchmarks.

CR Categories: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Physically based modeling

Keywords: Contact force, penalty-based methods, continuous collision detection, robust collision handling

Links: DL PDF WEB

*e-mail:tang_m@zju.edu.cn
†e-mail:dm@cs.unc.edu
‡e-mail:miguel.otaduy@urjc.es
§e-mail:trf@cs.unc.edu

1 Introduction

Contact and collision modeling is an important problem in computer graphics, haptics and robotics. With the recent advances in processing power, many interactive applications such as games and virtual environments use physics-based simulations to generate realistic behaviors. Two important components of a physical simulation are reliable detection of collisions between objects, and responding to those collisions by modeling a contact force that keeps the objects well-separated.

There is extensive literature on methods for detecting and handling collisions. Our work deals with penalty forces, which offer a computationally simple, yet often effective solution for collision response. In a nutshell, penalty methods estimate the magnitude of penetration depth once objects have collided and compute a restoring force as a function of the penetration depth. Penalty methods are often used for real-time applications [Heidelberger et al. 2004] as well as complex scenarios involving deformable solids, cloth [Baraff and Witkin 1998], and articulated models. Besides computer graphics, penalty methods are widely used in haptics [Hasegawa and Sato 2004; Barbič and James 2008], robotics [Yamane and Nakamura 2006; Drumwright 2008; Rengifo et al. 2009], and engineering applications [Wriggers 2006].

There are many robustness and stability problems that arise due to use of penalty-based methods. Two key issues are computation of force magnitude and force direction. An appropriate choice of force should ensure penetration constraint resolution while maintaining continuity and smoothness, both in space and time. Inappropriate choices involving force discontinuities, underestimation or overestimation of force magnitude may lead to problems such as jitter, instability, or constraint violation in the form of excessive penetration or even pop-through.

Main Results: We present a simple algorithm to compute continuous penalty-based forces between rigid and deformable models. Our approach is general and decomposes a mesh into vertices, edges, and triangles. We use continuous collision detection (CCD) algorithms to estimate *penetration time intervals* between overlapping feature pairs, by approximating their trajectories using continuous formulations. We take into account both vertex-face and edge-edge feature pairs, and compute analytic expressions of con-

tact impulses by accumulating continuous penalty forces along the entire feature trajectories. The overall algorithm offers the following benefits:

- **Simplicity:** Our approach retains the simplicity and flexibility of penalty-based forces.
- **Efficiency:** The overall computation is very fast. It takes about 3 μsec to check for collisions for each penetrating feature pair and compute the response force, on a single core.
- **Integrated collision detection and response:** The underlying formulation does not miss any penetrations and computes the response force for each vertex-face and edge-edge feature pair.
- **Improved robustness:** As compared to previous penalty-based methods, we observe reduction in jitter and constraint violation, as well as improved stability.

The rest of the paper is organized as follows. We give a brief overview of prior work in Section 2. Section 3 introduces our notation and presents closed form expressions for continuous penalty-based forces. We highlight their performance on different benchmarks in Section 4 and compare with prior algorithms in Section 5.

2 Related Work

We give a brief overview of prior work in collision detection and contact force computation.

Collision Detection: There is extensive work on collision detection and contact computation [Teschner et al. 2005]. At a broad level, prior methods can be classified into discrete and continuous collision detection. Discrete methods check for collisions or penetration at a specific time instance. Continuous methods are regarded as more robust as they check for collisions between two discrete time instances and compute the first time of contact [Bridson et al. 2002; Redon et al. 2004; Ortega et al. 2007; Tang et al. 2009].

Contact Force Computation: At a broad level, collision response algorithms can be classified into three categories [Witkin and Baraff 1997]: constraint-based formulations [Bridson et al. 2002; Duriez et al. 2004; Pauly et al. 2004; Duriez et al. 2006; Otaduy et al. 2009], penalty-based methods [Terzopoulos et al. 1987; Moore and Wilhelms 1988; Wriggers et al. 1990], and impulse-based methods [Mirtich 1996]. In general, constraint-based methods result in a more plausible simulation at the cost of extra computation. In this paper, we limit ourselves to penalty-based methods.

Penalty-Based Methods: Penalty-based methods stem from the definition of a conservative force field that tries to restore a non-penetrating state when two or more objects have penetrated. Multiple definitions of penetration depth exist, and we use one of the simplest formulations. Given a point $\vec{x}$ belonging to object A and penetrating object B, the simplest definition of penetration depth $\delta(\vec{x})$ is the distance to the closest point to $\vec{x}$ on the surface of B. Based on the magnitude of penetration depth, the basic penalty energy is typically defined as:

$$E(\vec{x}) = \frac{1}{2} k\, \delta(\vec{x})^2, \tag{1}$$

where k is the stiffness constant. The penalty force can be computed as the gradient of the penalty energy, i.e.,

$$\vec{F}(\vec{x}) = -\nabla E(\vec{x}) = -k\, \delta(\vec{x})\, \nabla \delta(\vec{x}). \tag{2}$$

Given the above definition of penetration depth, the (negative) gradient of penetration depth equals the unit surface normal $\vec{n}$ at the closest point on the surface of the penetrated object. The penalty force can be computed simply as:

$$\vec{F}(\vec{x}) = k\, \delta(\vec{x})\, \vec{n}. \tag{3}$$

This basic formulation, *discrete penalty force*, suffers from several problems. One is that penetration depth is not differentiable at points lying on the medial axis of the penetrated object, due to a discontinuity in the definition of closest points on the boundary. This discontinuity translates into a discontinuous contact normal $\vec{n}$, and the resulting penalty forces at those points are discontinuous in direction. In a discrete-time setting, the discontinuity of the penalty force can be approximated with a large derivative. Unfortunately, a large derivative formulation may violate the passivity of the discrete-time simulation [Mahvash and Hayward 2005]. Non-passive contact adds spurious energy to colliding objects, and may result in undesired jitter (i.e., high-frequency deviation from the expected path) or even instability (i.e., unbounded growth of physical quantities such as momentum). To avoid discontinuities, Barbič and James [2008] redefine $\vec{n}$ as the surface normal at the penetrating point $\vec{x}$, but this approach is discrete and may miss collisions under large time steps.

Another difficulty with respect to penalty methods is that the simplest, local approach described above for defining penetration depth based on the closest point on the penetrated surface may not result in a globally consistent solution. Global inconsistency is characterized by the existence of multiple penetrating points with opposing penalty forces, and may prevent the resolution of inter-penetration or induce pop-through artifacts. Using only points of maximum penetration is not sufficient, as it may result in oscillatory behavior [Drumwright 2008]. Heidelberger et al. [2004] improve consistency by performing a global contact treatment. Other approaches include computation of global penetration depth, but this approach is suited mostly for rigid bodies. Moreover, it is difficult to compute global penetration depth for non-convex objects by taking into account rotational motion [Zhang et al. 2007].

Another issue with penalty methods is that, in order to avoid inter-penetrations, repulsive forces need to be stiff or non-linear [Terzopoulos et al. 1987]. Higher contact stiffness increases the risk of instability, in particular if the force direction is discontinuous. In such cases, a high stiffness could magnify the discontinuities.

Despite their difficulties, penalty-based methods are widely used, even on complex applications such as cloth simulation [Baraff and Witkin 1998; Choi and Ko 2002]. However, several authors have highlighted the challenges in computing appropriate contact forces that can prevent inter-penetrations [Bridson et al. 2002; Heidelberger et al. 2004; Drumwright 2008]. Many algorithms use distance fields to estimate inter-penetrations and compute response forces [Fisher and Lin 2001; Heidelberger et al. 2004; Teschner et al. 2005]. Harmon et al. [2009] introduce a new penalty-based approach that uses a barrier method instead of stiff springs, and use a mixed explicit-implicit integrator to simulate contact forces at a higher rate than the rest of the system [Harmon et al. 2011]. Ellis et al. [1997] propose a filtering method to partially solve instability problems due to discrete sampling of contact forces in haptic rendering. To overcome the stability problems of penalty forces based on penetration depth, Hasegawa and Sato [2004] propose a volume-based penalty method to compute contact forces between colliding rigid bodies for haptic rendering. Volume-based penalty methods have also been extended to deformable bodies [Faure et al. 2008; Teran et al. 2005]. Some of the challenges in these approaches include the appropriate partitioning of global penetration volumes into sub-volumes [Allard et al. 2010].

3 Continuous Contact Handling

In this section, we introduce our notation and present our continuous penalty force (CPF) formulation.

3.1 Notation

Our algorithm is applicable to rigid and deformable models, which are represented as triangulated meshes. We do not make any assumptions about the motion of any object or its deformation, though our current formulation is limited to explicit time integration. Furthermore, the objects may undergo topological changes, such as fracture. We use the symbols V, E, and F to represent vertices, edges, and faces, respectively. We use lower-case symbols v, e, and t to denote a specific vertex, edge, and triangle, respectively. Vector quantities are represented with a little arrow, e.g., $\vec{n}$ for the normal of a triangle. Moreover, we use the notation $\{v, f\}$ or $\{e, e\}$ to denote a pairwise relationship between two features.

3.2 CCD and Penetration Depth

We normalize each simulation time step to the unit interval $[0, 1]$. For collision response, we consider all possible pairwise features, $\{v, f\}$ and $\{e, e\}$, and use CCD algorithms to identify the first time of contact between these pairs [Tang et al. 2009]. The simplest algorithms for CCD assume constant vertex velocities during the time interval, which yield linear vertex trajectories. In this case, the problem of finding the times of collision reduces to computing the roots of a cubic polynomial [Provot 1997]. Other methods represent the trajectories using higher degree polynomials based on constant acceleration or rotational object motions. For example, finding the times of contact in the constant acceleration case requires computing the roots of a degree 6 polynomial. In the rest of the paper, we assume linear trajectories, although our formulation can be easily extended to higher order trajectories (see Section 3.6).

As discussed in Section 2, the simplest penalty methods compute penetration depth based on the distance between closest points. Instead, we use a different definition of penetration depth δ that leverages CCD. We compute all overlapping VF and EE pairs using a CCD algorithm. We use the symbols $\vec{p}$ and $\vec{q}$ to represent the *collision points* on each pair of contact features. These points, $\vec{p}$ and $\vec{q}$, correspond to the first-time-of-contact between those feature pairs and are traced along their deforming trajectories, as the features move (Fig. 2). We also define a contact normal $\vec{n}$ for each feature pair, given by the triangle's normal for a VF pair, and by the normalized cross-product of edge vectors for an EE pair. For subsequent simulation time steps, we define the penetration depth as the distance between the collision points projected onto the contact normal, i.e.,

$$\delta = \vec{n}^T(\vec{p} - \vec{q}). \tag{4}$$

Once a feature pair is identified by CCD, we consider it to be colliding until its penetration depth reduces to zero. Our definition of contact features prevents discontinuities in the definition of the contact normal, ensures a continuous and smooth penetration depth, and therefore ensure a continuous penalty force as defined by Equation (3). These properties largely improve stability and robustness of penalty methods as shown in our benchmarks.

For a colliding feature pair, we define a *penetration time interval* $[t_a^i, t_b^i] \in [0, 1]$ as a time interval during which their penetration depth is strictly positive. Even with linear trajectories, an elementary collision test for a VF or EE pair may return up to 3 real roots. Therefore, the interval $[0, 1]$ may be split into multiple penetration time intervals. Fig. 3 shows two cases with higher-degree trajectories and two penetration time intervals.

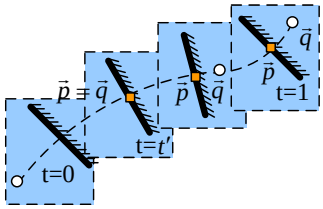


Figure 2: ***Collision points of a VF pair:*** *For a triangle and a vertex undergoing deformation, the collision points $\vec{p}$ and $\vec{q}$ are computed at the first-time-of-contact t' and traced along their deforming trajectories as the features move.*

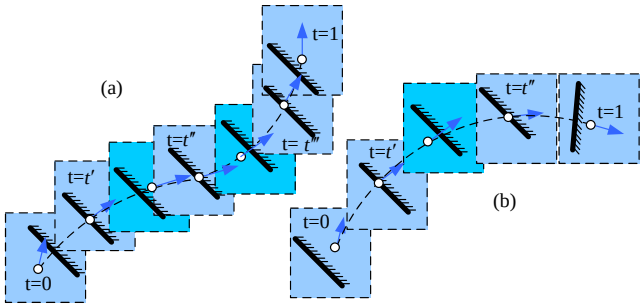


Figure 3: ***Deforming VF pair:*** *For the deforming VF pair in (a), there are three times of contact (t', t'', and t''') in [0, 1]. Two penetration time intervals $[t', t'']$ and $[t''', 1]$ are used to compute continuous penalty forces in this case. For the VF pair shown in (b), the features are penetrating at $t = 0$. Two times of contact (t' and t'') yield two penetration time intervals, $[0, t']$ and $[t'', 1]$.*

3.3 Continuous Penalty Forces

Our continuous penalty force formulation is based on the following observation: for any feature pair that is inter-penetrating and moves along a continuous path between two discrete positions, the contact force from Equation (3) varies continuously along the feature trajectories. Based on this observation, we present a continuous penalty force formulation that takes into account the overall trajectory of penetration features (see Fig. 4(b)).

To derive our force formulation, we pay attention to the impulse $\vec{I}$ produced by a penalty force $\vec{F}$ during a time interval of length Δt. From Newton's Second Law of Motion, the change in momentum produced by an impulse $\vec{I}$ is defined as

$$m\,\vec{v}(t + \Delta t) = m\,\vec{v}(t) + \vec{I}. \tag{5}$$

The impulse produced by a time-dependent force $\vec{F}$ between time instants t_a and t_b is defined using the following integral:

$$\vec{I} = \int_{t_a}^{t_b} \vec{F}(t)dt. \tag{6}$$

Given this formulation, the choice of integration method (e.g., explicit vs. implicit) determines the computation of impulse $\vec{I}$ in Equation (5). In our current implementation, we opt for an explicit Euler integrator. We first apply all other forces F_{other}, then we predict the velocity $\vec{v}^*(t + \Delta t) = \vec{v}(t) + \frac{\Delta t}{m}\vec{F}_{\text{other}}$ based on those forces, execute CCD, and finally correct the momentum based on the penalty impulse. Altogether, we compute the change in momentum as

$$m\,\vec{v}(t+\Delta t) = m\,\vec{v}(t) + \int_t^{t+\Delta t} \vec{F}(\vec{x}(t) + (\tau - t)\vec{v}^*(t + \Delta t))d\tau. \tag{7}$$

Theorem 1: Continuous Penalty Force Theorem. *For a pair of moving collision points $\vec{p}$ and $\vec{q}$, with time-varying contact normal $\vec{n}$, the impulse produced by a continuous time-dependent penalty force in the time interval $[0, 1]$ is defined as:*

$$\vec{I} = k \sum_{i=0}^{i<N} \int_{t_a^i}^{t_b^i} \vec{n}(t)^T \left(\vec{p}(t) - \vec{q}(t)\right) \vec{n}(t)\, dt, \tag{8}$$

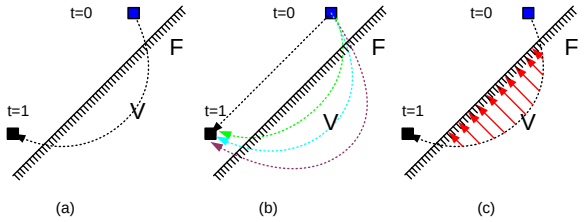


Figure 4: ***Trajectory based penalty forces:*** *Discrete penalty force computation tends to be independent of the actual trajectories of features between discrete time instances; therefore, it returns zero contact force for the colliding VF pair in (a). Our continuous penalty force, on the other hand, averages the force in (c) along the trajectory, and therefore produces different collision response for different trajectories shown in (b).*

where k is the stiffness constant, $[t_a^i, t_b^i] \in [0,1]$ the i-th penetration time interval, and N the number of penetration time intervals.

Proof. We substitute the definition of penalty force in Equation (3) into the definition of impulse produced by a time-dependent force in Equation (6) for the interval $[0,1]$:

$$\vec{I} = \int_0^1 k\,\delta(t)\,\vec{n}(t)dt. \tag{9}$$

The penetration depth is $\delta = 0$ when the objects are disjoint. Therefore, we split the integral into penetration time intervals,

$$\vec{I} = \sum_{i=0}^{i<N} \int_{t_a^i}^{t_b^i} k\,\delta(t)\,\vec{n}(t)dt. \tag{10}$$

Finally, substituting our definition of penetration depth from Equation (4), we obtain the expression in Equation (8). □

We use the formulation in Equation (8) to derive the forces for VF and EE feature pairs in Sections 3.4 and 3.5 respectively.

By analogy with Euler integrators, the impulse in Equation (5) can be interpreted as the integral of a constant force, $\vec{I} = \Delta t\,\vec{F}^*$. For our choice of explicit impulse in Equation (7), this constant force turns out to be simply the time-average of the continuous penalty force,

$$\vec{F}^* = \frac{1}{\Delta t}\int_t^{t+\Delta t} \vec{F}(t)dt. \tag{11}$$

This observation allows us to further interpret our formulation of continuous penalty forces in Equation (8). In the example shown in Fig. 4(a), discrete penalty methods produce no contact force, as they depend only on the end positions of penetrating features. Our continuous formulation, on the other hand, accumulates the penalty force along the trajectory to compute the total contact impulse. As we observe, this is equivalent to the integration of the average penalty force along the trajectory. As a result, the subtle trajectory variations in Fig. 4(b) result in different penalty forces.

3.4 VF Contact Force

Based on Equation (8), for VF pairs under linear interpolating motion (Fig. 5), we derive the following formulation for penalty force computation:

Corollary 1: Continuous VF Contact Force Corollary. *The impulses produced by a time-varying penalty force on a vertex p and*

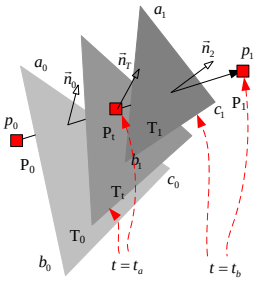


Figure 5: ***Continuous penalty force for VF contact:*** *For a triangle T and a vertex p undergoing linear motion, the accumulated impulse can be computed by evaluating a polynomial of degree 6 (Equation (12)).*

the vertices a, b and c of a triangle T, over N penetration time intervals $[t_a^i, t_b^i] \in [0,1]$, can be expressed as

$$\vec{I}_p = k\sum_{i=0}^{i<N}\int_{t_a^i}^{t_b^i} (\vec{n}_T)^T(\vec{p} - w_a\,\vec{a} - w_b\,\vec{b} - w_c\,\vec{c})\,\vec{n}_T\,dt, \tag{12}$$

$$\vec{I}_a = -w_a\,\vec{I}_p,\quad \vec{I}_b = -w_b\,\vec{I}_p,\quad \vec{I}_c = -w_c\,\vec{I}_p. \tag{13}$$

The point $q = w_a\,\vec{a} + w_b\,\vec{b} + w_c\,\vec{c}$ corresponds to a collision point on the triangle T, expressed as a Barycentric combination of vertex positions; $\vec{n}_T$ is the normal of the triangle; and k is the contact stiffness. The symbols $\vec{p}$, $\vec{a}$, $\vec{b}$, $\vec{c}$ and $\vec{n}_T$ correspond to time-varying functions.

For vertices with constant velocities and end positions $(\vec{p}_0, \vec{p}_1)$, $(\vec{a}_0, \vec{a}_1)$, $(\vec{b}_0, \vec{b}_1)$ and $(\vec{c}_0, \vec{c}_1)$, the vertex velocities can be computed as $\vec{v}_p = \vec{p}_1 - \vec{p}_0$, and similarly for $\vec{v}_a$, $\vec{v}_b$ and $\vec{v}_c$. Then the time-varying contact normal can be expressed as

$$\vec{n}_T(t) = \frac{\vec{n}_0\,B_0^2(t) + \vec{n}_1\,B_1^2(t) + \vec{n}_2\,B_2^2(t)}{L(t)}, \tag{14}$$

with $B_i^2(t) = \frac{2!}{i!(2-i)!}\,t^i\,(1-t)^{2-i}$ the Bernstein polynomials of degree 2; (constant) basis coefficients $\vec{n}_0 = (\vec{b}_0 - \vec{a}_0) \times (\vec{c}_0 - \vec{a}_0)$, $\vec{n}_2 = (\vec{b}_1 - \vec{a}_1) \times (\vec{c}_1 - \vec{a}_1)$, and $\vec{n}_1 = \frac{\vec{n}_0 + \vec{n}_2 - (\vec{v}_b - \vec{v}_a)\times(\vec{v}_c - \vec{v}_a)}{2}$, respectively; and a (time-varying) normalization factor $L(t) = \sqrt{(L_0\;L_1\;\ldots\;L_4)\cdot(B_0^4(t)\;B_1^4(t)\;\ldots\;B_4^4(t))^T}$.

The exact coefficients $\{L_0 \ldots L_4\}$ are given in the supplementary document. The computation of the exact accumulated impulse requires the integral of rational functions, due to the normalization factor $L(t)$. We use a numerical integration scheme based on quadrature to compute the exact integral. However, we found that the approximation of $L(t)$ given as

$$L(t) \approx L_k = \sqrt{\frac{L_0 + L_1 + L_2 + L_3 + L_4}{5}} \tag{15}$$

produces no noticeable difference in simulation results, as shown in Fig. 6. On the other hand, the use of the constant normalization term L_k largely simplifies the computation of the accumulated impulse in Equation (12), providing large performance speedups in all our benchmarks, up to an order of magnitude.

The evaluation of the impulse follows from Equation (12), where t is the integration variable. $\vec{n}_T$ is approximated by a quadratic

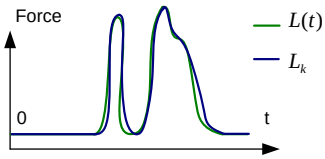


Figure 6: ***Result comparison between exact and approximate normal vector:*** *By approximating $L(t)$ with L_k, the magnitude of contact force differs by $4.2\% - 8.1\%$ for the Chain Benchmark (Fig. 1(a)).*

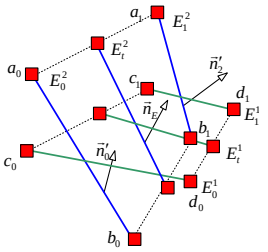


Figure 7: ***Continuous penalty force for EE contact:*** *For two deforming edges E^1 and E^2 undergoing linear motion, the accumulated impulse can be computed by evaluating a polynomial of degree 6 (Equation (16)).*

polynomial (by replacing $L(t)$ with L_k); $\vec{p}$, $\vec{a}$, $\vec{b}$, and $\vec{c}$ are linear polynomials; w_a, w_b, and w_c are scalars. Therefore, the integrand corresponds to a degree-five polynomial, and the total impulse can be computed as a degree-six polynomial by summing over all penetration time intervals. The exact terms of this degree-six polynomial are given in the supplementary document.

3.5 EE Contact Force

Similarly, for a given EE pair undergoing linear interpolating motion (Fig. 7), we derive the following formulation for penalty force computation:

Corollary 2: Continuous EE Contact Force Corollary. *The impulses produced by a time-varying penalty force on the vertices a and b of an edge E^1 and the vertices c and d of an edge E^2, over N penetration time intervals $[t_a^i, t_b^i] \in [0,1]$, can be expressed as*

$$\vec{I}_E = k \sum_{i=0}^{i<N} \int_{t_a^i}^{t_b^i} \vec{n}_E^T (w_a\,\vec{a} + w_b\,\vec{b} - w_c\,\vec{c} - w_d\,\vec{d})\,\vec{n}_E\,dt, \quad (16)$$

$$\vec{I}_a = w_a\,\vec{I}_E,\ \vec{I}_b = w_b\,\vec{I}_E,\ \vec{I}_c = -w_c\,\vec{I}_E,\ \vec{I}_d = -w_d\,\vec{I}_E. \quad (17)$$

The points $p = w_a\,\vec{a} + w_b\,\vec{b}$ and $q = w_c\,\vec{c} + w_d\,\vec{d}$ correspond to the collision points on edges E^1 and E^2, respectively, expressed as Barycentric combinations of vertex positions; $\vec{n}_E$ is the normalized cross product of edge vectors; and k is the contact stiffness. The symbols, $\vec{a}$, $\vec{b}$, $\vec{c}$, $\vec{d}$ and $\vec{n}_E$ correspond to time-varying functions.

As in the VF case, we assume constant velocities and linear trajectories between the vertex end-positions. The time-varying contact normal can be expressed as

$$\vec{n}_E(t) = \frac{\vec{n}'_0\,B_0^2(t) + \vec{n}'_1\,B_1^2(t) + \vec{n}'_2\,B_2^2(t)}{L'(t)}, \quad (18)$$

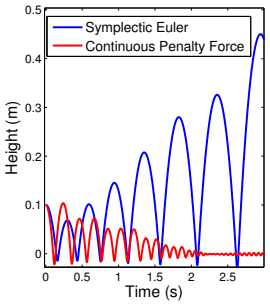
with Bernstein basis coefficients $\vec{n}'_0 = (\vec{b}_0 - \vec{a}_0) \times (\vec{d}_0 - \vec{c}_0)$, $\vec{n}'_2 = (\vec{b}_1 - \vec{a}_1) \times (\vec{d}_1 - \vec{c}_1)$, and $\vec{n}'_1 = \frac{\vec{n}'_0 + \vec{n}'_2 - (\vec{v}_b - \vec{v}_a) \times (\vec{v}_d - \vec{v}_c)}{2}$, respectively; and a (time-varying) normalization factor $L'(t)$ similar to the one in the VF case.

3.6 Discussion

Theoretical Smoothness Analysis: We present a theoretical derivation to show that our method (CPF) can attenuate jitter and produce smoother motion. Our method compares stability conditions and robustness to perturbations of a 1D linear contact scenario, for Symplectic Euler (SE) and CPF. We show that CPF exhibits increased robustness to perturbations and nonlinearities.

For a simple 1D particle with mass m dropped onto a plane with penalty stiffness k, symplectic Euler (SE) is stable for time steps $\Delta t < 2\sqrt{\frac{m}{k}}$. This stability condition can be reached through eigenvalue analysis of the linear state update rule (See the supplementary document for a derivation). For this 1D case, our CPF method is linear, and stable for time steps $\Delta t < \sqrt{2\frac{m}{k}}$ (worse than SE due to the filtering introduced by integration). However, SE is critically stable for the pure linear system, with eigenvalues exactly $|\lambda| = 1$, and may hence suffer jitter under nonlinearities or perturbations. With CPF, the eigenvalues are always $|\lambda| < 1$ in the stable regime. In other words, similar to e.g., implicit Euler, CPF adds numerical damping, which attenuates jitter.

We have confirmed this analysis empirically too. We have simulated the particle's motion with SE and CPF, while applying sinusoidal perturbations to the penalty stiffness. The plot on the left shows the motion of a 1kg particle, as it is dropped from a height of 0.1m onto a plane with stiffness 10kN/m $\pm 25\%$, simulated with a time step of 2ms. The motion is damped with CPF, but the same situation becomes unstable with SE. Under these conditions, we found SE to be robust only to stiffness perturbations below 15%, while CPF is robust to stiffness perturbations up to 70%. Our method's numerical damping could be regarded as a limitation in some situations, and the design of a penalty method that is both free of numerical damping and robust to jitter, for general 3D contacts, is an exciting area for future research.



Initializing the Penetrating State: Penetrations may occur at the beginning of a time step, i.e., $t = 0$. This is not an issue, as the contact impulse in Equation (8) can naturally handle such situations by accumulating the continuous penalty force during penetration time intervals. An example of initialization in penetrating state is shown in Fig. 3(b). In this example, two penetration time intervals, $[0, t']$ and $[t'', 1]$, are used to compute continuous penalty forces.

Contact Normal: $\vec{n}_t$ is defined as the contact normal of the penetrating features. These features are identified at the first time of contact and are tracked until the penetration is resolved, but $\vec{n}_t$ properly rotates as the features deform between discrete time instances. Our approach could be extended by continuously redefining feature pairs during sliding contact (similar to [Irving et al. 2004]).

Extension to Higher Order Deformations: The VF and EE contact force theorems can easily generalize to cases where vertices deform based on higher degree functions, i.e., $d > 1$. With the constant approximation of normalization factors $L(t)$ and $L'(t)$, the time-varying contact normals $\vec{n}_T$ and $\vec{n}_E$ can be approximated using Bernstein basis functions of degree $2d$. Therefore, the accumulated contact impulses can be computed by evaluating polynomials of degree $5d+1$. For example, with constant-acceleration deformations (i.e., with vertex positions defined by polynomials of degree 2), the complexity for CCD and continuous force computation will correspond to polynomials of degree 6 and degree 11, respectively.

Friction Force: A friction force orthogonal to the contact force, both static and kinetic, can be defined using Coulomb's law [Mirtich 1996]. Even though one could formulate a friction impulse by trajectory integration (similar to the continuous penalty force), we use a simple and efficient formulation to compute the friction force, which is independent of the contact force, with a single friction factor μ [Yamane and Nakamura 2006; Drumwright 2008].

4 Results

Our algorithm can be used with explicit time integration schemes and any dynamic simulation algorithm that employs penalty meth-

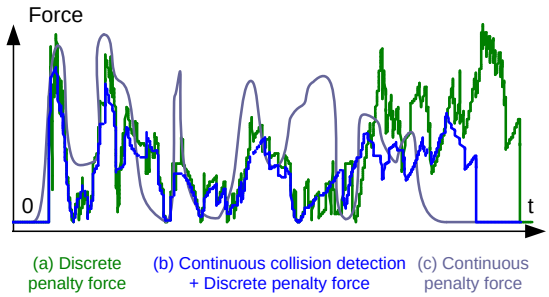


Figure 8: ***Force stability in Chain Benchmark:*** *We plot the magnitude of the contact forces between the swinging chain and the rigid box shown in Fig. 1(a). The continuous forces (c) result in more stable behavior over discrete penalty forces (a), and show smoother changing than discrete penalty forces with CCD (b). The continuous penalty force is computed using penetration time intervals, while the discrete penalty force is computed based on the penetration depth at t=1.*

ods for collision response. Specifically, we have implemented our CPF computation for mesh-based models in the SOFA system, which is freely available for download at http://www.sofa-framework.org/. All the experiments were carried out on a standard PC (Windows/XP 32bits, Q6600 CPU@2.4GHz, 2GB RAM), and the timings reported here were generated using a single core.

In all examples, we assume that the vertices of the mesh deform according to linear trajectories (i.e., with constant velocities) during each time step, and we perform the VF/EE elementary collision tests by solving cubic equations to compute the penetration time intervals. We use the Self-CCD [1] package to perform collision computations. In practice, computing penetration time intervals for each pair takes about 2 μsec. Our approximation of continuous penalty forces defines contact impulses as polynomials of degree 6. In practice, the cost of computing a penalty impulse is under 1 μsec.

Chain Benchmark (Fig. 1(a)) highlights the benefit of continuous contact forces as the swinging chain collides against a rigid box. We use a symplectic Euler integrator with a large time step ($> 0.4s$). Under such a large time step, standard discrete penalty methods suffer from contact leaking, leading to inter-penetrating configurations that cannot be resolved, and producing a 'sticking' behavior. These problems are alleviated using our continuous penalty force formulation that is based on penetration time intervals. Fig. 8 compares the profile of contact forces during the simulation under three different settings. (a) Standard discrete penalty methods produce contact forces with high-frequency noise. (b) The combination of discrete penalty forces with our definition of penetration depth based on C-CD (see Section 3.2) reduces noise because of the continuity of contact normals. (c) CPF results in smooth force computation.

Stick Benchmark (Fig. 9) demonstrates the different results generated with (a) continuous penalty force computation and (b) discrete penalty force computation. In this case, the ball bounces back after hitting the stick. We compute a more accurate trajectory with continuous contact forces, i.e., the ball should bounce back upwards. At $t = 0.45$ the stick and the ball have already collided (with an angle > 90 degree). With discrete penalty force computation, an incorrect result is obtained, i.e., the ball bounces back horizontally.

Octopus Benchmark (Fig. 10 & Fig. 11) and Cloth Benchmark (Fig. 12 & Fig. 13) demonstrate the application of continuous penalty forces on deformable objects. The Dragons Benchmark (Fig. 14) highlights the scalability of the continuous formulation in

[1] http://gamma.cs.unc.edu/SELFCD/

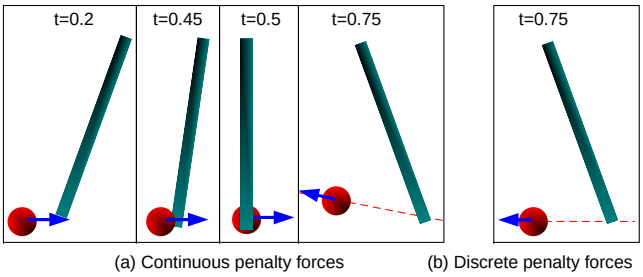


Figure 9: ***Stick Benchmark:*** *Different trajectories (shown at $t = 0.75$) computed using continuous vs. discrete penalty forces: the rigid ball bounces back at different velocities (shown with arrows) after colliding with a stick. ($3k$ triangles, 60fps)*

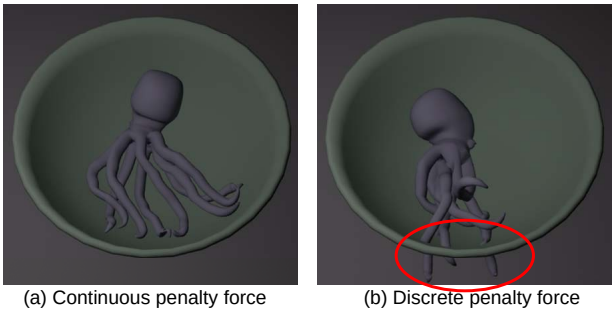


Figure 10: ***Octopus Benchmark:*** *A soft deforming octopus in a rigid bowl. The benchmark exhibits many inter-object & intra-object collisions, but continuous penalty forces result in a robust simulation and no penetration despite the thin features. The model has $14.7k$ triangles and the frame rate of the simulation is $10fps$ on a single core.*

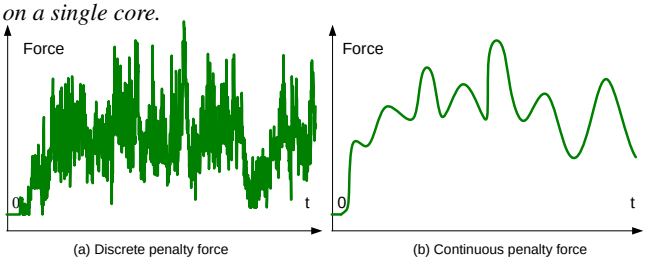


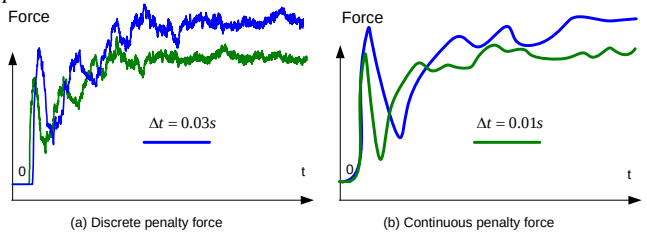
Figure 11: ***Force stability in Octopus Benchmark:*** *The magnitude of contact forces between the soft octopus and the rigid bowl shows improved stability with continuous vs. discrete penalty force computation. The continuous penalty force is computed using penetration time intervals, while the discrete penalty force is computed using penetration depth values at $t = 1$.*

terms of the number of geometric features. The Rings Benchmark (Fig. 1(b)) demonstrates the benefit of continuous penalty forces in terms of alleviating over-penetration problems.

Our approach is not limited to closed objects, as shown in the cloth (Fig. 12) and bucket examples (Fig. 14). We perform CCD tests and compute penetration depth for VF or EE features (Eq. (4)) without any inside/outside information. Our method handles both inter- and intra-cloth collisions, but it may not resolve all tangled scenarios.

We use SOFA scene files to define the settings for all benchmarks: stiffness constants, friction models, time-step sizes, damping, integration methods, collision detection tolerances, etc.. For the Chain Benchmark, we have used a time step of 0.02s, a collision tolerance of 0.4mm, and a stiffness constant of 10N/m. For the Cloth Benchmark, we have used a time step of 0.0333s, a collision tolerance

Figure 12: ***Cloth Benchmark:*** *Multiple deforming tori fall on the cloth and generate many inter-object collisions (65k triangles, 6fps). Using discrete penalty forces, the simulation suffers severe penetrations between the tori and the cloth.*



(a) Discrete penalty force (b) Continuous penalty force

Figure 13: ***Force stability in Cloth Benchmark:*** *The magnitude of contact forces between the deforming tori and the cloth shows improved stability with continuous forces. Using discrete penalty forces, the contact force tends to be noisy with a large time step ($\Delta t = 0.03s$, blue curve) as well as a small time step ($\Delta t = 0.01s$, green curve). With continuous penalty forces, the contact force is smooth with the large and small time steps.*

of 0.2mm, and a stiffness constant of 2N/m. We used the friction model implemented in SOFA, which is orthogonal to contact force computation. We set the viscous friction coefficient to 0.005, and Coulomb friction coefficient to 0.8. We observe good energy behavior in our benchmarks. From an implementation point of view, our continuous penalty force approach can be easily integrated into the SOFA framework by simply replacing the penalty force computation method, and by changing the collision detection routine from "NewProximityIntersection" to "ContinuousIntersection". In the examples, we use the same sets of parameters to compute both discrete and continuous penalty forces.

5 Analysis & Comparisons

5.1 Benefits

There are two main reasons for the various benefits of using continuous penalty forces. First, contact primitives are detected using CCD and tracked along their trajectories. This feature provides valid contact primitives even under large time steps, alleviates force inconsistency issues, and prevents excessive inter-penetration. Second, collision impulses are computed by accumulating continuous penalty forces along the trajectories of contact primitives. This feature provides a low-pass filter of penalty forces that reduces jitter and improves stability.

In practice, our continuous penalty force formulation can alleviate many of the robustness issues associated with discrete penalty forces, such as:

Jitter: The Chain Benchmark (Fig. 1(a)) exhibits jitter under discrete penalty forces, as shown in Fig. 8 (also in Fig. 11 & Fig. 13).

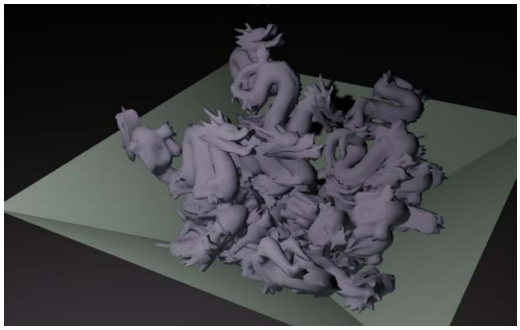


Figure 14: ***Dragons Benchmark:*** *40 deforming dragons fall into a rigid bucket (103k triangles, 3fps), and generate many inter-object and intra-object collisions. Using discrete penalty methods, the simulation fails to resolve object inter-penetrations.*

The filtering effect of our continuous penalty force formulation dramatically reduces jitter and produces smooth forces. The supplementary video shows an example of a box falling on a plane that comes to rest fine with continuous forces, but the jitter present with discrete forces prevents it from stopping completely.

Inter-Penetration: In the Chain (Fig. 1(a)), Octopus (Fig. 10), and Rings Benchmarks (Fig. 1(b)), the use of traditional penalty methods based on discrete collision detection leads to excessive, often irreversible, inter-penetrations. In the Octopus Benchmark, the video clearly shows how two of the tentacles penetrate the bowl. In the Rings Benchmark, penetrations with the thin pins are not resolved. In all these benchmarks, our continuous penalty force formulation resolves interactions satisfactorily and significantly alleviates inter-penetration problems. These improvements are possible because we detect contact features based on CCD, and track these features along their trajectories to compute penetration time intervals.

Inconsistent Forces: As a consequence of excessive inter-penetration, typical penalty methods may enter situations with inconsistent forces that fail to resolve collisions correctly, producing sticking behavior or pop-through effects. The Chain Benchmark (Fig. 1(a)) highlights a clear example of sticking behavior with discrete penalty methods, which is resolved with our continuous penalty force formulation. The Stick Benchmark (Fig. 9) illustrates another limitation of discrete penalty methods. For two moving objects, discrete penalty forces are computed at a particular point in their trajectories, resulting in inaccurate force directions. Again, our continuous penalty force formulation, which computes the first time of contact, alleviates force direction inaccuracies.

Fast Moving Objects: When objects move rapidly and collide, the continuous penalty force tends to produce a repulsive force based on contact features at the first-time-of-contact, and will restore the objects to a non-penetrating state accordingly.

5.2 Parameter and Performance Analysis

Fig. 15 and Fig. 16 highlight the contact forces between the falling rings and the base in the Ring Benchmark (Fig. 1(b)), under various stiffness and time step values. Our continuous penalty forces demonstrate good behavior for all these cases. We also note that with a very small time step ($\delta t = 0.0001$), the discrete penalty force tends to be smooth.

CCD and penalty force computation (CPF) take about 3 μsec per feature pair. The CCD+CPF computations are about $3 - 4$ times slower than discrete collision detection (DCD) & force computation (DPF). Most of this slowdown is due to the use of CCD, to guarantee that no collisions are missed. The additional overhead of

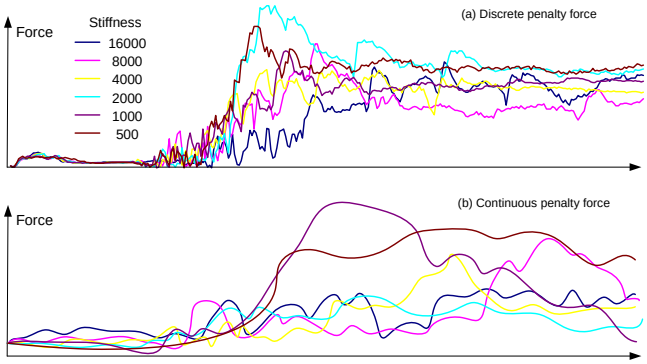


Figure 15: ***Contact forces with different stiffness:*** *We plot the magnitude of the contact forces between the falling rings and the base shown in Fig. 1(b) ($\Delta t = 0.03333$).*

CPF corresponds to evaluating a degree-6 polynomial vs. a linear polynomial. Even if we reduce the size of the time step for discrete force computation to one third of the original time step, the resulting DPF is not smooth (see Fig. 16). In the Cloth Benchmark (Fig. 12), we need a step size $< \delta t = 0.0002s$ (i.e. $50 - 100$x smaller) to produce smooth DPFs.

5.3 Comparison

Unlike distance field based approaches [Fisher and Lin 2001; Barbič and James 2008], our algorithm performs no pre-processing to construct the distance field and computes a more accurate measure of penetration. Our formulation of continuous contact forces is based on local penetration time intervals between the overlapping features. Compared to volume-based approaches [Faure et al. 2008], which use intersection volumes to compute contact forces and distribute the forces to vertices with appropriate weights, our formulation computes the contact force for each vertex using an analytic formulation. Most prior penalty-based algorithms calculate the contact forces using only the end positions of the object. Although Ortega et al. [2007] used CCD, along with constraint-based methods, for haptic interaction between rigid bodies, it is difficult to extend his approach to deformable models. Hasegawa and Sato [2004] performed integration on the intersection volume of convex polytopes to compute contact forces for haptics. This formulation is prone to degeneracies and is relatively expensive.

Many techniques have been proposed in the literature to overcome the robustness problems, which include jitter and lack of stability. The techniques proposed are based on adaptive time steps, checking relative velocity of objects, use of signed distance fields, different integration methods (e.g. implicit integrators), etc. Our formulation is orthogonal to these approaches and the main benefit arises from the continuous formulation of the trajectory and computation of contact forces by accumulating the contact force along the entire trajectory. No additional parameters or other forms of tweaking are used, and our continuous formulation can be easily combined with other methods.

5.4 Limitations

Our approach has some limitations. Degenerate collision scenarios, such as edge-edge collisions corresponding to parallel edges, may require special treatment. Under large rotational motions between discrete time instances, our approximation of penetration depth based on linear vertex trajectories may not be accurate. Furthermore, our algorithm computes only local penetration depth, as opposed to global penetration depth [Zhang et al. 2007], and we

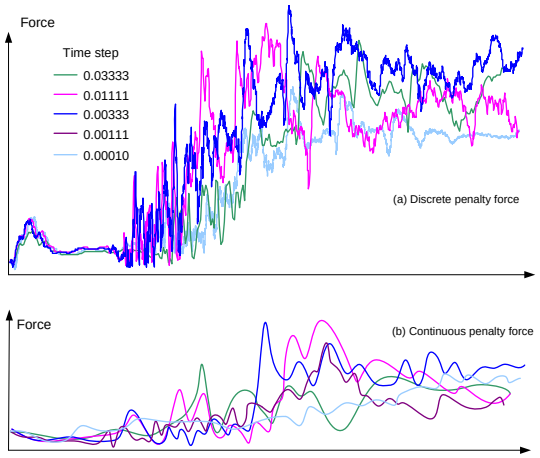


Figure 16: ***Contact forces with different time steps:*** *We plot the magnitude of the contact forces between the falling rings and the base shown in Fig. 1(b) (Stiffness $= 8000$).*

cannot rigorously guarantee global resolution of inter-penetrations. Finally, our current formulation of contact impulses uses explicit integration of continuous penalty forces, and could suffer instability issues under very large time steps or large contact stiffness values. Stability could be improved by extending our formulation to handle implicit integration methods. They are typically nonlinear and solved with Newton's method, which would not be a problem for our continuous force definition, which is differentiable. Finding times of contact implicitly, on the other hand, would probably require some approximation.

6 Conclusion and Future Work

We present a novel continuous penalty force computation algorithm between rigid and deformable objects that estimates penetration time intervals between overlapping features. The contact force is expressed as a closed-form polynomial expression, and this force is integrated along feature trajectories to produce a smooth contact impulse. We highlight the improved robustness of penalty-based methods on many benchmarks.

There are many avenues for future work. We would like to further improve the robustness of our approach against all type of inter-penetrations. It may be possible to obtain more accurate force computations by using higher order trajectories for deep penetrations. Finally, we would like to explore the combination of our approach with constraint-based methods to perform fast and robust collision handling between complex models.

Acknowledgements: We would like to thank François Faure and the SOFA team for their support, and Jianfei Chen for useful discussions. This research is supported in part by NSFC (61170140), the National Basic Research Program of China (2011CB302205), the National Key Technology R&D Program of China (2012BAD35B01), NSFZC (Y1100069). Manocha is supported in part by ARO Contract W911NF-10-1-0506, NSF awards 0917040, 0904990, 1000579 and 1117127, and Intel. Otaduy is supported in part by the Spanish Ministry of Science and Innovation (TIN2009-07942) and by the European Research Council (ERC-2011-StG-280135 Animetrics). Tong is partly supported by NSFC (61170141).

References

ALLARD, J., FAURE, F., COURTECUISSE, H., FALIPOU, F.,

DURIEZ, C., AND KRY, P. G. 2010. Volume contact constraints at arbitrary resolution. *ACM Transactions on Graphics 29*, 3.

BARAFF, D., AND WITKIN, A. 1998. Large steps in cloth simulation. In *Proc. of ACM SIGGRAPH*, 43–54.

BARBIČ, J., AND JAMES, D. L. 2008. Six-DoF haptic rendering of contact between geometrically complex reduced deformable models. *IEEE Transactions on Haptics 1*, 1, 39–52.

BRIDSON, R., FEDKIW, R., AND ANDERSON, J. 2002. Robust treament for collisions, contact and friction for cloth animation. In *Proc. of ACM SIGGRAPH*, 594–603.

CHOI, K.-J., AND KO, H.-S. 2002. Stable but responsive cloth. In *Proc. of ACM SIGGRAPH*, 604–611.

DRUMWRIGHT, E. 2008. A fast and stable penalty method for rigid body simulation. *IEEE Transactions on Visualization and Computer Graphics 14* (January), 231–240.

DURIEZ, C., ANDRIOT, C., AND KHEDDAR, A. 2004. Signorini's contact model for deformable objects in haptic simulations. In *Proc. of IEEE Int'l Conf. Intelligent Robots and Systems*, 3232–3237.

DURIEZ, C., DUBOIS, F., KHEDDAR, A., AND ANDRIOT, C. 2006. Realistic haptic rendering of interacting deformable objects in virtual environments. *IEEE Transactions on Visualization and Computer Graphics 12*, 1, 36–47.

ELLIS, R. E., SARKAR, N., AND JENKINS, M. A. 1997. Numerical methods for the force reflection of contact. *ASME Transactions on Dynamic Systems Measurement and Control 119*, 4, 768–774.

FAURE, F., BARBIER, S., ALLARD, J., AND FALIPOU, F. 2008. Image-based collision detection and response between arbitrary volumetric objects. In *Proc. of ACM Siggraph/Eurographics Symp. on Computer Animation*, 155–162.

FISHER, S., AND LIN, M. C. 2001. Deformed distance fields for simulation of non-penetrating flexible bodies. In *Proc. of Eurographic workshop on Computer animation & simulation*, 99–111.

HARMON, D., VOUGA, E., SMITH, B., TAMSTORF, R., AND GRINSPUN, E. 2009. Asynchronous contact mechanics. In *Proc. of ACM SIGGRAPH*, 87:1–87:12.

HARMON, D., ZHOU, Q., AND ZORIN, D. 2011. Asynchronous integration with phantom meshes. In *Proceedings of the 2011 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '11, 247–256.

HASEGAWA, S., AND SATO, M. 2004. Real-time rigid body simulation for haptic interactions based on contact volume of polygonal objects. *Computer Graphics Forum 23*, 3, 529–538.

HEIDELBERGER, B., TESCHNER, M., KEISER, R., MÜLLER, M., AND GROSS, M. 2004. Consistent penetration depth estimation for deformable collision response. In *Proc. of Vision, Modeling, Visualization*, 339–346.

IRVING, G., TERAN, J., AND FEDKIW, R. 2004. Invertible finite elements for robust simulation of large deformation. In *Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation*, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, SCA '04, 131–140.

MAHVASH, M., AND HAYWARD, V. 2005. High-fidelty passive force-reflecting virtual environments. *IEEE Transactions on Robotics 21*, 1, 38–46.

MIRTICH, B. 1996. *Impulse-based Dynamic Simulation of Rigid Body Systems*. Ph.D. thesis, Dept. Elec. Engin. Comput. Sci., Univ. California, Berkeley, CA.

MOORE, M., AND WILHELMS, J. 1988. Collision detection and response for computer animation. In *Proc. of ACM SIGGRAPH*, 289–298.

ORTEGA, M., REDON, S., AND COQUILLART, S. 2007. A six degree-of-freedom god-object method for haptic display of rigid bodies with surface properties. *IEEE Transactions on Visualization and Computer Graphics 13*, 458–469.

OTADUY, M. A., TAMSTORF, R., STEINEMANN, D., AND GROSS, M. 2009. Implicit contact handling for deformable objects. In *Proc. of Eurographics*, 559–568.

PAULY, M., PAI, D. K., AND GUIBAS, L. J. 2004. Quasi-rigid objects in contact. In *Proc. of the 2004 ACM SIGGRAPH/Eurographics Symp. Computer animation*, 109–119.

PROVOT, X. 1997. Collision and self-collision handling in cloth model dedicated to design garment. *Graphics Interface*, 177–189.

REDON, S., KIM, Y. J., LIN, M. C., AND MANOCHA, D. 2004. Fast continuous collision detection for articulated models. In *Proceedings of the ninth ACM symposium on Solid modeling and applications*, 145–156.

RENGIFO, C., AOUSTIN, Y., CHEVALLEREAU, C., AND PLESTAN, F. 2009. A penalty-based approach for contact forces computation in bipedal robots. In *Proc. of IEEE-RAS Int'l Conf. on Humanoid Robots*, 121 – 127.

TANG, M., CURTIS, S., YOON, S.-E., AND MANOCHA, D. 2009. ICCD: Interactive continuous collision detection between deformable models using connectivity-based culling. *IEEE Transactions on Visualization and Computer Graphics 15*, 4, 544–557.

TERAN, J., SIFAKIS, E., IRVING, G., AND FEDKIW, R. 2005. Robust quasistatic finite elements and flesh simulation. In *Proc. of the ACM SIGGRAPH/Eurographics Symp. on Computer animation*, 181–190.

TERZOPOULOS, D., PLATT, J., BARR, A., AND FLEISCHER, K. 1987. Elastically deformable models. In *Proc. of ACM SIGGRAPH*, 205–214.

TESCHNER, M., KIMMERLE, S., HEIDELBERGER, B., ZACHMANN, G., RAGHUPATHI, L., FUHRMANN, A., P. CANI, M., FAURE, F., MAGNENAT-THALMANN, N., STRASSER, W., AND VOLINO, P. 2005. Collision detection for deformable objects. *Computer Graphics Forum 24*, 61–81.

WITKIN, A., AND BARAFF, D. 1997. Physically based modeling: Principles and practice. *SIGGRAPH 1997 Course*.

WRIGGERS, P., VU VAN, T., AND STEIN, E. 1990. Finite element formulation of large deformation impact-contact problems with friction. *Computers & Structures 37*, 3, 319–331.

WRIGGERS, P. 2006. *Computational Contact Mechanics, 2nd Edition*. Springer.

YAMANE, K., AND NAKAMURA, Y. 2006. Stable penalty-based model of frictional contacts. In *Proc. of IEEE Int'l Conf. on Robotics and Automation*, 1904 –1909.

ZHANG, L., KIM, Y. J., VARADHAN, G., AND MANOCHA, D. 2007. Generalized penetration depth computation. *Comput. Aided Des. 39*, 625–638.

Continuous Penalty Forces
Supplementary Material

Min Tang*
State Key Lab of CAD&CG, Zhejiang University

Dinesh Manocha†
University of North Carolina at Chapel Hill

Miguel A. Otaduy‡
URJC Madrid

Ruofeng Tong§
State Key Lab of CAD&CG, Zhejiang University

http://gamma.cs.unc.edu/CPF/

In this supplementary document, we include:

- Proofs of the continuous normal theorems given in the paper.
- A detailed derivation of the polynomial corresponding to the continuous force for linear trajectories.
- A derivation of stability conditions of 1D particle-on-plane contact for Symplectic Euler and continuous penalty forces.

A Time-Varying Contact Normals for VF and EE Pairs

Theorem 1: Continuous Normal Theorem for a Deforming Triangle. *Given the start and end positions of the vertices of a triangle during the interval* $[0,1]$*, whose positions are linearly interpolated in the interval with respect to the time variable, t. The unit normal vector,* $\vec{n}_T(t)$*, of the triangle, at time t, is given by the equation:*

$$\vec{n}_T(t) = \frac{\vec{n}_0\, B_0^2(t) + \vec{n}_1\, B_1^2(t) + \vec{n}_2\, B_2^2(t)}{L(t)}, \tag{1}$$

where

- $B_i^2(t) = \frac{2!}{i!(2-i)!}\, t^i\, (1-t)^{2-i}$.
- $a_0, a_1, b_0, b_1, c_0, c_1$ *are the start and end positions of the three vertices of the deforming triangle, respectively.*
- $\vec{v}_a = a_1 - a_0$, $\vec{v}_b = b_1 - b_0$, *and* $\vec{v}_c = c_1 - c_0$.
- $\vec{n}_0 = (b_0 - a_0) \times (c_0 - a_0)$, $\vec{n}_2 = (b_1 - a_1) \times (c_1 - a_1)$, $\vec{n}_1 = \frac{\vec{n}_0 + \vec{n}_2 - (\vec{v}_b - \vec{v}_a) \times (\vec{v}_c - \vec{v}_a)}{2}$, *respectively.*
- $L_0 = (\vec{n}_0^T\, \vec{n}_0)$, $L_1 = (\vec{n}_0^T\, \vec{n}_1)$, $L_2 = \frac{2\,(\vec{n}_1^T\, \vec{n}_1) + (\vec{n}_0^T\, \vec{n}_2)}{3}$, $L_3 = (\vec{n}_1^T\, \vec{n}_2)$, $L_4 = (\vec{n}_2^T\, \vec{n}_2)$, *respectively.*
- $B_i^4(t) = \frac{4!}{i!(4-i)!}\, t^i\, (1-t)^{4-i}$.
- $L(t) = \sqrt{(L_0\ L_1\ \ldots\ L_4) \cdot (B_0^4(t)\ B_1^4(t)\ \ldots\ B_4^4(t))^T}$.

Proof. We define the following terms: $\vec{a}_t = \vec{a}_0 + \vec{v}_a\, t$, $\vec{b}_t = \vec{b}_0 + \vec{v}_b\, t$, and $\vec{c}_t = \vec{c}_0 + \vec{v}_c\, t$. The normal vector of triangle $\triangle a_t b_t c_t$ is given as:

$$\begin{aligned}
\vec{m}_t &= (\vec{b}_t - \vec{a}_t) \times (\vec{c}_t - \vec{a}_t) \\
&= [(b_0 - a_0) + (\vec{v}_b - \vec{v}_a)\, t] \times [(c_0 - a_0) + (\vec{v}_c - \vec{v}_a)\, t] \\
&= (b_0 - a_0) \times (c_0 - a_0) + (\vec{v}_b - \vec{v}_a) \times (c_0 - a_0)\, t + \\
&\quad (b_0 - a_0) \times (\vec{v}_c - \vec{v}_a)\, t + \\
&\quad (\vec{v}_b - \vec{v}_a) \times (\vec{v}_c - \vec{v}_a)\, t^2.
\end{aligned} \tag{2}$$

Let $\vec{n}_0$ and $\vec{n}_2$ be the normal vectors of triangle $\triangle a_0 b_0 c_0$ and $\triangle a_1 b_1 c_1$, respectively. Then:

$$\begin{aligned}
\vec{n}_0 &= (b_0 - a_0) \times (c_0 - a_0), \\
\vec{n}_2 &= (b_1 - a_1) \times (c_1 - a_1) \\
&= (b_0 + \vec{v}_b - a_0 - \vec{v}_a) \times (c_0 + \vec{v}_c - a_0 - \vec{v}_a) \\
&= (b_0 - a_0) \times (c_0 - a_0) + (\vec{v}_b - \vec{v}_a) \times (c_0 - a_0) + \\
&\quad (b_0 - a_0) \times (\vec{v}_c - \vec{v}_a) + (\vec{v}_b - \vec{v}_a) \times (\vec{v}_c - \vec{v}_a).
\end{aligned} \tag{3, 4}$$

Based on the above equations, we obtain:

$$\begin{aligned}
\vec{n}_2 - \vec{n}_0 &= (\vec{v}_b - \vec{v}_a) \times (c_0 - a_0) + (b_0 - a_0) \times (\vec{v}_c - \vec{v}_a) + \\
&\quad (\vec{v}_b - \vec{v}_a) \times (\vec{v}_c - \vec{v}_a).
\end{aligned} \tag{5}$$

We define:

$$\vec{\omega} = (\vec{v}_b - \vec{v}_a) \times (\vec{v}_c - \vec{v}_a). \tag{6}$$

Then from equations (5) and (6):

$$\vec{n}_2 - \vec{n}_0 - \vec{\omega} = (\vec{v}_b - \vec{v}_a) \times (c_0 - a_0) + (b_0 - a_0) \times (\vec{v}_c - \vec{v}_a) \tag{7}$$

By plugging the equations (3), (6), and (7) into equation (2), $\vec{m}_t$ can be represented as:

$$\begin{aligned}
\vec{m}_t &= \vec{n}_0 + (\vec{n}_2 - \vec{n}_0 - \vec{\omega})\, t + \vec{\omega}\, t^2 \\
&= \vec{n}_0\, (1-t)^2 + \frac{\vec{n}_0 + \vec{n}_2 - \vec{\omega}}{2}\, 2\, t\, (1-t) + \vec{n}_2\, t^2 \\
&= \vec{n}_0\, B_0^2(t) + \frac{\vec{n}_0 + \vec{n}_2 - \vec{\omega}}{2}\, B_1^2(t) + \vec{n}_2\, B_2^2(t) \\
&= \vec{n}_0\, B_0^2(t) + \vec{n}_1\, B_1^2(t) + \vec{n}_2\, B_2^2(t).
\end{aligned} \tag{8}$$

And:

$$\begin{aligned}
||\vec{m}_t||^2 &= \vec{m}_t^T\, \vec{m}_t \\
&= (\vec{n}_0^T\, \vec{n}_0)\, B_0^2(t)\, B_0^2(t) \\
&+ 2\, (\vec{n}_0^T\, \vec{n}_1)\, B_0^2(t)\, B_1^2(t) \\
&+ (\vec{n}_1^T\, \vec{n}_1)\, B_1^2(t)\, B_1^2(t) \\
&+ 2\, (\vec{n}_0^T\, \vec{n}_2)\, B_0^2(t)\, B_2^2(t) \\
&+ (\vec{n}_2^T\, \vec{n}_2)\, B_2^2(t)\, B_2^2(t) \\
&+ 2\, (\vec{n}_1^T\, \vec{n}_2)\, B_1^2(t)\, B_2^2(t).
\end{aligned} \tag{9}$$

Based on the properties of the Bernstein basis functions, we have:

$$\begin{aligned}
B_0^2(t)\, B_0^2(t) &= B_0^4(t) \\
B_0^2(t)\, B_1^2(t) &= \frac{B_1^4(t)}{2} \\
B_1^2(t)\, B_1^2(t) &= \frac{2\, B_2^4(t)}{3} \\
B_0^2(t)\, B_2^2(t) &= \frac{B_2^4(t)}{6} \\
B_1^2(t)\, B_2^2(t) &= \frac{B_3^4(t)}{2} \\
B_2^2(t)\, B_2^2(t) &= B_4^4(t).
\end{aligned} \tag{10}$$

*e-mail:tang_m@zju.edu.cn
†e-mail:dm@cs.unc.edu
‡e-mail:miguel.otaduy@urjc.es
§e-mail:trf@cs.unc.edu

By plugging the equation (10) into equation (9), we have:

$$\begin{aligned} ||\vec{m}_t||^2 &= (\vec{n}_0^T\,\vec{n}_0)\,B_0^4(t) + (\vec{n}_0^T\,\vec{n}_1)\,B_1^4(t) \\ &+ \frac{2\,(\vec{n}_1^T\,\vec{n}_1) + (\vec{n}_0^T\,\vec{n}_2)}{3}\,B_2^4(t) \\ &+ (\vec{n}_1^T\,\vec{n}_2)\,B_3^4(t) + (\vec{n}_2^T\,\vec{n}_2)\,B_4^4(t) \\ &= L(t)\,L(t). \end{aligned} \tag{11}$$

Based on equation (8) and (11), the normalized normal vector, $\vec{n}_T(t)$, will be:

$$\begin{aligned} \vec{n}_T(t) &= \frac{\vec{m}_t}{||\vec{m}_t||} \\ &= \frac{\vec{n}_0\,B_0^2(t) + \vec{n}_1\,B_1^2(t) + \vec{n}_2\,B_2^2(t)}{L(t)} \end{aligned} \tag{12}$$

□

Theorem 2: Continuous Normal Theorem for Two Deforming Edges. *Given the start and end positions of the vertices of two edges during the interval* $[0,1]$*, whose positions are linearly interpolated in the interval with respect to the time variable, t, the normal vector,* $\vec{n}_E(t)$*, between the two edges, at time t, is given by the equation:*

$$\vec{n}_E(t) = \frac{\vec{n}_0'\,B_0^2(t) + \vec{n}_1'\,B_1^2(t) + \vec{n}_2'\,B_2^2(t)}{L'(t)}, \tag{13}$$

where

- $B_i^2(t) = \begin{pmatrix} 2 \\ i \end{pmatrix}(1-t)^i\,t^{2-i}$.
- $a_0, a_1, b_0, b_1, c_0, c_1, d_0, d_1$ *are the start and end positions of the four vertices of the two deforming edges, respectively.*
- $\vec{v}_a = a_1 - a_0$, $\vec{v}_b = b_1 - b_0$, $\vec{v}_c = c_1 - c_0$, *and* $\vec{v}_d = d_1 - d_0$.
- $\vec{n}_0' = (b_0 - a_0) \times (c_0 - d_0)$, $\vec{n}_2' = (b_1 - a_1) \times (c_1 - d_1)$, $\vec{n}_1' = \frac{\vec{n}_0' + \vec{n}_2' - \vec{\omega}'}{2}$, *and* $\vec{\omega}' = (\vec{v}_b - \vec{v}_a) \times (\vec{v}_c - \vec{v}_d)$, *respectively.*
- $L_0' = (\vec{n}_0'^T\,\vec{n}_0')$, $L_1' = (\vec{n}_0'^T\,\vec{n}_1')$, $L_2' = \frac{2\,(\vec{n}_1'^T\,\vec{n}_1') + (\vec{n}_0'^T\,\vec{n}_2')}{3}$, $L_3' = (\vec{n}_1'^T\,\vec{n}_2')$, $L_4' = (\vec{n}_2'^T\,\vec{n}_2')$, *respectively.*
- $B_i^4(t) = \frac{4!}{i!(4-i)!}\,t^i\,(1-t)^{4-i}$.
- $L'(t) = \sqrt{(L_0'\;L_1'\;\ldots\;L_4') \cdot (B_0^4(t)\;B_1^4(t)\;\ldots\;B_4^4(t))^T}$.

Proof. We define the following terms: $\vec{a}_t = \vec{a}_0 + \vec{v}_a\,t$, $\vec{b}_t = \vec{b}_0 + \vec{v}_b\,t$, $\vec{c}_t = \vec{c}_0 + \vec{v}_c\,t$, and $\vec{d}_t = \vec{d}_0 + \vec{v}_d\,t$. The normal vector of the two edges defined by a_t, b_t and c_t, d_t, respectively, is given as:

$$\begin{aligned} \vec{m}_t' &= (\vec{b}_t - \vec{a}_t) \times (\vec{c}_t - \vec{d}_t) \\ &= [(b_0 - a_0) + (\vec{v}_b - \vec{v}_a)\,t] \times [(c_0 - d_0) + (\vec{v}_c - \vec{v}_d)\,t] \\ &= (b_0 - a_0) \times (c_0 - d_0) + (\vec{v}_b - \vec{v}_a) \times (c_0 - d_0)\,t + \\ &\quad (b_0 - a_0) \times (\vec{v}_c - \vec{v}_d)\,t + \\ &\quad (\vec{v}_b - \vec{v}_a) \times (\vec{v}_c - \vec{v}_d)\,t^2. \end{aligned} \tag{14}$$

Let $\vec{n}_0'$ and $\vec{n}_2'$ be the normal vectors of the two edges defined by a_0, b_0, c_0, d_0 and a_1, b_1, c_1, d_1, respectively. Then:

$$\vec{n}_0' = (b_0 - a_0) \times (c_0 - d_0), \tag{15}$$

$$\begin{aligned} \vec{n}_2' &= (b_1 - a_1) \times (c_1 - d_1) \\ &= (b_0 + \vec{v}_b - a_0 - \vec{v}_a) \times (c_0 + \vec{v}_c - d_0 - \vec{v}_d) \\ &= (b_0 - a_0) \times (c_0 - d_0) + (\vec{v}_b - \vec{v}_a) \times (c_0 - d_0) + \\ &\quad (b_0 - a_0) \times (\vec{v}_c - \vec{v}_d) + (\vec{v}_b - \vec{v}_a) \times (\vec{v}_c - \vec{v}_d). \end{aligned} \tag{16}$$

Based on the above equations, we obtain:

$$\begin{aligned} \vec{n}_2' - \vec{n}_0' &= (\vec{v}_b - \vec{v}_a) \times (c_0 - d_0) + (b_0 - a_0) \times (\vec{v}_c - \vec{v}_d) + \\ &\quad (\vec{v}_b - \vec{v}_a) \times (\vec{v}_c - \vec{v}_d). \end{aligned} \tag{17}$$

We define:

$$\vec{\omega}' = (\vec{v}_b - \vec{v}_a) \times (\vec{v}_c - \vec{v}_d). \tag{18}$$

Then from equations (17) and (18):

$$\vec{n}_2' - \vec{n}_0' - \vec{\omega}' = (\vec{v}_b - \vec{v}_a) \times (c_0 - d_0) + (b_0 - a_0) \times (\vec{v}_c - \vec{v}_d) \tag{19}$$

By plugging the equations (15), (18), and (19) into equation (14), $\vec{m}_t'$ can be represented as:

$$\begin{aligned} \vec{m}_t' &= \vec{n}_0' + (\vec{n}_2' - \vec{n}_0' - \vec{\omega}')\,t + \vec{\omega}'\,t^2 \\ &= \vec{n}_0'\,(1-t)^2 + \frac{\vec{n}_0' + \vec{n}_2' - \vec{\omega}'}{2}\,2\,t\,(1-t) + \vec{n}_2'\,t^2 \\ &= \vec{n}_0'\,B_0^2(t) + \frac{\vec{n}_0' + \vec{n}_2' - \vec{\omega}'}{2}\,B_1^2(t) + \vec{n}_2'\,B_2^2(t) \\ &= \vec{n}_0'\,B_0^2(t) + \vec{n}_1'\,B_1^2(t) + \vec{n}_2'\,B_2^2(t) \end{aligned} \tag{20}$$

Analogous to the deduction in Theorem 1, the normalized normal vector, $\vec{n}_E(t)$, will be:

$$\begin{aligned} \vec{n}_E(t) &= \frac{\vec{m}_t'}{||\vec{m}_t'||} \\ &= \frac{\vec{n}_0'\,B_0^2(t) + \vec{n}_1'\,B_1^2(t) + \vec{n}_2'\,B_2^2(t)}{\sqrt{(L_0'\;L_1'\;\ldots\;L_4') \cdot (B_0^4(t)\;B_1^4(t)\;\ldots\;B_4^4(t))^T}} \\ &= \frac{\vec{n}_0'\,B_0^2(t) + \vec{n}_1'\,B_1^2(t) + \vec{n}_2'\,B_2^2(t)}{L'(t)} \end{aligned} \tag{21}$$

□

B Computation of Continuous Penalty Force

In this section, we give the exact formula for the degree six polynomial used for contact force computation. We use Corollary I to derive this formula for VF contact force.

Coefficients of the Degree-Six Polynomial. For the evaluation of the following equation:

$$\vec{I}_p = k \sum_{i=0}^{i<N} \int_{t_a^i}^{t_b^i} (\vec{n}_T)^T (\vec{p} - w_a\,\vec{a} - w_b\,\vec{b} - w_c\,\vec{c})\,\vec{n}_T\,dt,$$

where t is the unknown. $\vec{n}_T$ is approximated by a quadratic polynomial (by replacing $L(t)$ with L_k). $\vec{p}$, $\vec{a}$, $\vec{b}$, $\vec{c}$ are linear polynomials; w_a, w_b, and w_c are scalars.

From Theorem 1, let:

$$\begin{aligned} \vec{n}_T &= \hat{a}t^2 + \hat{b}t + \hat{c}, \\ \vec{p} - w_a\,\vec{a} - w_b\,\vec{b} - w_c\,\vec{c} &= \hat{d}t + \hat{e}, \end{aligned}$$

where:

$$\begin{aligned} \hat{a} &= \frac{\vec{n}_0 - 2\vec{n}_1 + \vec{n}2}{L_k}, \\ \hat{b} &= \frac{2(\vec{n}_1 - \vec{n}_0)}{L_k}, \\ \hat{c} &= \frac{\vec{n}_0}{L_k}, \\ \hat{d} &= p_0 - w_a a_0 - w_b b_0 - w_c c_0, \\ \hat{e} &= (p_1 - p0) - w_a(a_1 - a_0) - w_b(b_1 - b_0) - w_c(c_1 - c_0). \end{aligned}$$

The inner term of the integral corresponds to a degree-five polynomial:

$$(\vec{n}_T)^T(\vec{p}-w_a\,\vec{a}-w_b\,\vec{b}-w_c\,\vec{c})\,\vec{n}_T = \hat{a}'t^5+\hat{b}'t^4+\hat{c}'t^3+\hat{d}'t^2+\hat{e}'t+\hat{f}',$$

where:

$$\begin{aligned}
\hat{a}' &= (\hat{a}^T\,\hat{d})\,\hat{a},\\
\hat{b}' &= (\hat{a}^T\,\hat{d})\,\hat{b} + (\hat{a}^T\,\hat{e} + \hat{b}^T\,\hat{d})\,\hat{a},\\
\hat{c}' &= (\hat{a}^T\,\hat{d})\,\hat{c} + (\hat{a}^T\,\hat{e} + \hat{b}^T\,\hat{d})\,\hat{b} + (\hat{b}^T\,\hat{e} + \hat{c}^T\,\hat{d})\,\hat{a},\\
\hat{d}' &= (\hat{a}^T\,\hat{e} + \hat{b}^T\,\hat{d})\,\hat{c} + (\hat{b}^T\,\hat{e} + \hat{c}^T\,\hat{d})\,\hat{b},\\
\hat{e}' &= (\hat{b}^T\,\hat{e} + \hat{c}^T\,\hat{d})\,\hat{c} + (\hat{c}^T\,\hat{e})\,\hat{b},\\
\hat{f}' &= (\hat{c}^T\,\hat{e})\,\hat{c}.
\end{aligned}$$

Then we get the coefficients of the degree-six polynomial:

$$\int_{t_a^i}^{t_b^i} (\vec{n}_T)^T(\vec{p} - w_a\,\vec{a} - w_b\,\vec{b} - w_c\,\vec{c})\,\vec{n}_T\,dt =$$
$$(\frac{\hat{a}'t^6}{6} + \frac{\hat{b}'t^5}{5} + \frac{\hat{c}'t^4}{4} + \frac{\hat{d}'t^3}{3} + \frac{\hat{e}'t^2}{2} + \hat{f}'t)|_{t_a^i}^{t_b^i}.$$

Analog to VF contact force, we can similarly derive the coefficients of the degree six polynomial for EE contact force.

C Stability Analysis of 1D Particle-on-Plane Contact

The (continuous) motion of a particle with mass m under the action of gravity and a penalty force with stiffness k centered at $x = 0$ is described, through Newton's 2^{nd} Law, as:

$$m\,\dot{v} = -m\,g - k\,x. \tag{22}$$

To analyze the stability of an integration method, we discretize the equation above with a time step Δt, discard the gravity force term, and write an iterative update rule of the form

$$\begin{pmatrix} x(t+\Delta t) \\ v(t+\Delta t) \end{pmatrix} = A \begin{pmatrix} x(t) \\ v(t) \end{pmatrix}. \tag{23}$$

The integration method is stable for time steps for which all eigenvalues $\|\lambda(A)\| < 1$.

With Symplectic Euler (SE), the velocity and position updates can be written as

$$\begin{aligned}
v(t+\Delta t) &= v(t) - \Delta t\,\frac{k}{m}\,x(t), &(24)\\
x(t+\Delta t) &= x(t) + \Delta t\,v(t+\Delta t). &(25)
\end{aligned}$$

In the form of equation 23, the update rule is

$$\begin{pmatrix} x(t+\Delta t) \\ v(t+\Delta t) \end{pmatrix} = \begin{pmatrix} 1-\Delta t^2\,\frac{k}{m} & \Delta t \\ -\Delta t\,\frac{k}{m} & 1 \end{pmatrix} \begin{pmatrix} x(t) \\ v(t) \end{pmatrix}. \tag{26}$$

And the eigenvalue analysis yields $\Delta t < 2\sqrt{\frac{m}{k}}$ for stability.

With Continuous Penalty Forces (CPF), following the formulation in Section 3.3 in the paper, we first predict the velocity at the end of the time step, which in this case is simply $v^*(t+\Delta t) = v(t)$. Then, we integrate the penalty force, and we obtain the following average force:

$$F^* = \frac{1}{\Delta t}\int_0^{\Delta t} -k\,(x(t) + \tau\,v(t))\,d\tau = -k\,x(t) - \Delta t\frac{k}{2}\,v(t). \tag{27}$$

The velocity and position updates with CPF can be written as

$$\begin{aligned}
v(t+\Delta t) &= v(t) - \Delta t\,\frac{k}{m}\,x(t) - \Delta t^2\,\frac{k}{2\,m}\,v(t), &(28)\\
x(t+\Delta t) &= x(t) + \Delta t\,v(t+\Delta t). &(29)
\end{aligned}$$

In the form of equation 23, the update rule is

$$\begin{pmatrix} x(t+\Delta t) \\ v(t+\Delta t) \end{pmatrix} = \begin{pmatrix} 1-\Delta t^2\,\frac{k}{m} & \Delta t - \Delta t^3\,\frac{k}{2m} \\ -\Delta t\,\frac{k}{m} & 1-\Delta t^2\,\frac{k}{2m} \end{pmatrix} \begin{pmatrix} x(t) \\ v(t) \end{pmatrix}. \tag{30}$$

And the eigenvalue analysis yields $\Delta t < \sqrt{2\,\frac{m}{k}}$ for stability.