Multi-contact Frictional Rigid Dynamics using Impulse Decomposition

Sheng Li[1,2] Tianxiang Zhang[1] Guoping Wang[1] Hanqiu Sun[3] Dinesh Manocha[2]
[1]Peking University [2]University of North Carolina at Chapel Hill [3]Chinese University of HongKong

***Abstract*— We present an interactive and stable multi-contact dynamic simulation algorithm for rigid bodies. Our approach is based on fast frictional dynamics (FFD) [14], which is designed for large sets of non-convex rigid bodies. We use a new friction model that performs velocity-level multi-contact simulation using impulse decomposition. Moreover, we accurately handle friction at each contact point using contact distribution and frictional impulse solvers, which also account for relative motion. We evaluate our algorithm's performance on many complex multi-body benchmarks with thousands of contacts. In practice, our dynamics simulation algorithm takes a few milliseconds per timestep and exhibits more stable behaviors.**

I. Introduction

Rigid body dynamics is widely studied in different domains, including robotics, haptics, and physically-based modeling. There is extensive literature in the field on collision detection and contact resolution. Some of the main challenges include handling complex contact scenarios with non-penetration constraints and performing such simulations at interactive rates.

There is considerable work on collision detection between rigid models and current methods can efficiently check for collisions using discrete or continuous methods. Many techniques have been proposed for collision response. The simplest algorithms are based on penalty forces, which compute a response force as a function of the penetrating distance. However, these methods may not be stable, especially when there are multiple contacts. Other sets of algorithms are based on impulse velocity approaches, which are more accurate and more stable than penalty-based methods [18]. Some of the most popular methods for rigid body dynamics are developed based on applying Linear Complementarity Programming (LCP) [17], [2]. The LCP model has been widely studied and used for rigid simulation [3]. However, LCP-based methods can be expensive, especially for interactive applications.

A key issue in rigid body dynamics is accurate resolution of contacts. Current methods can be very sensitive to small variations in the configuration and position of the contacts. In scenarios with multiple contacts, this sensitivity can result in high variation in the resulting simulation [5]. Moreover, the accurate computation of frictional forces also remains a major challenge.

Fast frictional dynamics (FFD) is an alternative method for rigid body dynamics that is used to simulate large sets of non-convex objects[14]. It uses a different friction model in the configuration space of rigid bodies along with quadratic programming (QP) to model these contacts. It is quite fast in practice and can handle thousands of non-convex objects with multiple contacts. However, due to the approximation of normal impulse distribution by a projection operation, the modeling of friction may not be robust. Moreover, the algorithm may not be able to correctly model friction caused by relative motion.

Main Results: We present an improved FFD algorithm for multi-body contacts. Our formulation is based on a fast and accurate algorithm that computes the normal impulse at each contact point using decomposition, which is computed using a contact distribution solver. Next, the frictional impulse for each contact point is computed using a frictional impulse solver. Finally, we compute the stable frictional impulse by coupling the frictional impulse and the normal contact impulse into a convex space. We have applied our algorithm to many complex rigid body simulations with thousands of contact points and can resolve these contacts robustly at interactive rates. We highlight its improved stability by comparing the performance between our method, FFD, and LCP using PGS solver. Overall, our algorithm provides a realtime solution for complex rigid body simulations with good accuracy.

The rest of the paper is organized as follows. We give a brief overview of prior work in Section 2. We introduce our notation and give a background on FFD in Section 3. Our new algorithm is described in Section 4. We highlight its performance on different benchmarks in Section 5.

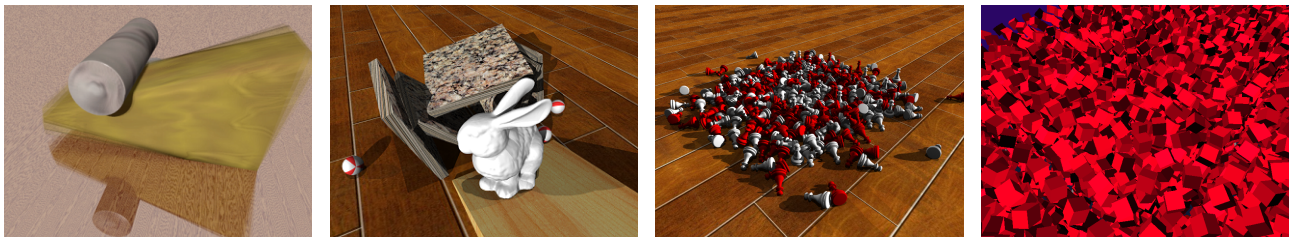
II. Related Work

Rigid body dynamics has been well-studied in different areas for more than three decades [30]. In this section, we give a brief overview of prior techniques for contact resolution in multi-contact configurations [25].

Multi-body contacts

The penalty force method is one of the simplest models used for contact resolution [12]. In practice, the penalty force model suffers from stability problems and can be extremely sensitive to the stiffness of rigid bodies [24]. Many techniques have been proposed to improve its stability, such as continuous penalty forces [27] and handling thin shell rigid bodies with interpenetration-free guarantees [7]. These techniques are also used in robotics [13].

The LCP model is widely used for rigid body simulations [2] and it can compute feasible solutions for contact resolution. The LCP regards the force at each contact point as an



(a) (b) (c) (d)

Fig. 1: *Four rigid body benchmarks with multi-body frictional contacts: (a) Bar: A rotating cylinder rolling on a wedge with friction; (b) Stack: Multiple contacts between rolling balls, a bunny, and stacked boards; (c) Chess: 320 chess pieces falling onto the ground; (d) Cube:* 5000 *cubes dropped into a box result in* $60K$ *contact points. In all these benchmarks, our algorithm can accurately resolve the contacts with high stability.*

unknown and uses appropriate equations to model all the non-penetration constraints. Overall, LCP corresponds to complementary problem formulation because it is impossible for two contacting bodies to have both positive contact forces and positive relative velocities (i.e. separating from each other) at the same time. This is referred to as the *Signorini Condition* [25], [29]. Many improvements have been proposed based on the LCP model, e.g., the generalized reflection (GR) model [23] has been proposed to simulate the phenomenon of body-detaching after impact. The quadratic contact energy (QCE) model [33] can be used to evaluate the energy variations during multi-body impact. In practice, the LCP model can result in high-quality simulation results.

Given the fact that LCP tends to model a non-linear combinatorial problem with a linear structure [25], it is expensive to solve LCP. Some faster techniques have been proposed based on iterative LCP solvers [21], [11], [6]. Recently Gauss-Siedel-based splitting methods have been used [8], [29]. With these methods, the simulation of a large scene with over 10K contact points can be handled. In practice, the accuracy of these iterative LCP solvers depends on the number of iterations. Fewer iterations can result in stability problems and may result in abnormalities such as jitters or crashing.

Friction

Friction is important in terms of contact response in a multi-contact system with non-smooth modeling [16], [4]. Overall, frictional force and normal contact force are coupled and influence each other. Theoretically they should be solved simultaneously, but in most cases they are processed separately because solving a coupled problem can be costly and difficult. For simplicity, the Coulomb friction model and maximal dissipation law [10] are widely used. Recently, Todorov proposed a new optimization-based variant of the maximal dissipation law, which can be solved using the interior point method [28]. A frictional model for penalty-based simulation is presented in [32] and can be improved using an implicit approach to simulate stable penalty-based frictional contact [31]. A staggered projection method is described for generating realistic results [15], though its computational cost is high.

Moreau et al. [20] proposed a model for frictionless multi-body dynamics according to Gauss's principle of least constraint [19]. Kaufman et al. [14] extended this work and proposed a new friction computation model by computing normal impulses that solve the constraints in the configuration space. In their approach, the distribution of contact impulses on each contact point is approximated by a simple projection operation. Therefore, the impulse distribution can only be considered as a rough approximation and may not be accurate. Because of this approximation, it can be difficult to simulate detailed frictional phenomena, which require an accurate impulse distribution. Moreover, solving the frictional impulse for each object independently by the maximal dissipation law implies that its relative motion with respect to other objects is ignored. That may lead to abnormalities in the resulting simulation. We describe an approach to address these problems.

III. Background

In this section, we introduce the notation and basic concepts used in the rest of the paper and give an overview of the FFD algorithm. The details about FFD and the Lie algebra of $SE(3)$ space are described in [14] and we follow their notation.

Given a rigid body B, we represent its configuration as $q = (P, R)$, where P represents the translation of B's centroid and R represents the rotation. The term *velocity* is called *twist* in $se(3)$ space, where $se(3)$ is the tangent space to $SE(3)$. The contact impulses are called *wrenches* in $se^*(3)$ space, where $se^*(3)$ is the cotangent space to $SE(3)$ We summarize the symbols used in the paper in Table II.

In this table, ${}_i f = (f_r, f_t)^T$, where f_r and f_t represent the impulses that drive the rigid body to rotate and translate in $\mathbb{R}^3$ space, respectively. The variable ${}^i\Gamma_k$ is introduced to connect $SE(3)$ space with $\mathbb{R}^3$ space: ${}^i\Gamma_k = (-\hat{x}_k, I)$, where I is a unit matrix. A variable in $\mathbb{R}^3$ space with a hat on top of it represents a skew-symmetric matrix, e.g. $\forall v \in \mathbb{R}^3$,

Symbol	Description
i	i-th object in the multi-body system
k	k-th contact point on a rigid body
$A(q)$	an optimized set of contact points
T	a constraint space of a rigid body
M	inertia tensor of a rigid body
ω	an object's rotation velocity in $\mathbb{R}^3$
v	an object's spatial velocity in $\mathbb{R}^3$
x_k	k-th contact point's position in $\mathbb{R}^3$
$\dot{x}_k$	k-th contact point's velocity in $\mathbb{R}^3$
f_k	k-th contact point's impulse in $\mathbb{R}^3$
${}^i\phi = (\omega, v)^T$	i-th object's twist
${}_if$	i-th object's wrench (contact impulse)
α_k	k-th contact point's normal impulse
${}^ir$	twist variation imposed by normal impulse
${}^if$	twist variation imposed by total impulse
${}^i\Gamma_k$	equivalent transformation $\mathbb{R}^3$ and $SE(3)$
${}^in_k$	normal of k-th contact point on i-th object

TABLE I: *Definitions of basic symbols.*

$v = (v_0, v_1, v_2)$,

$$\hat{v} = \begin{pmatrix} 0 & -v_2 & v_1 \\ v_2 & 0 & -v_0 \\ -v_1 & v_0 & 0 \end{pmatrix}. \tag{1}$$

Suppose an impulse f_k in $\mathbb{R}^3$ is applied on x_k of a body; the corresponding wrench of this impulse is given as:

$${}_if = {}^i\Gamma_k^T \ f_k. \tag{2}$$

The velocity in $\mathbb{R}^3$ of any point x_k of this body can be obtained from the twist based on

$${}^i\Gamma_k {}^i\phi = \dot{x}_k. \tag{3}$$

The transformation between the wrench and the twist can be:

$${}^if = M^{-1}{}_if, \tag{4}$$

where M is the inertia tensor of a rigid body. In $SE(3)$, two arbitrary vectors' inner product are calculated as

$$a \cdot b = a^T M b. \tag{5}$$

A. Contact Distribution

The FFD algorithm has two main steps. First the contact impulses are computed without friction. Next, the frictional impulses are calculated based on the results from the first step.

In order to ensure that each contact pair satisfies the non-penetration constraint, a space T (which is composed of all feasible twists that have positive relative velocities along the contact normals) is constructed for each object. According to the contact model based on Gauss's principle of least constraint [20], the value of the contact impulse should be the minimum that can satisfy the constraint at each contact point. The twist variation ${}^ir$ is computed as:

$${}^ir = {}^i\phi^+ - {}^i\phi^-, \tag{6}$$

where ${}^i\phi^+ = proj_T({}^i\phi^-) \in \partial T$, and ${}^i\phi^-$ and ${}^i\phi^+$ represent the twist corresponding to pre-collision and post-collision, respectively. ${}^i\phi^+$ is computed by using the projection operation. Because T corresponds to a a convex space, the twist ${}^i\phi^+$ after the projection is on ∂T, which represents the boundary of T.

In order to compute the distribution of normal contact impulses, an approximation is used [14], which applies the projection of the total contact impulse along normal ${}^in_k$:

$$\alpha_k = {}^in_k^T \ M \ {}^ir. \tag{7}$$

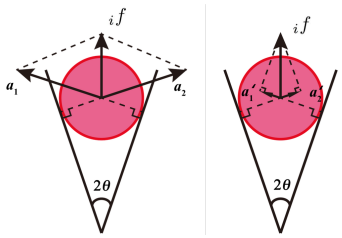


Fig. 2: *(a) Correct normal impulse distribution; (b) Approximation using projection along two normals respectively [14].* ${}_if$ *is the composition of contact impulse, and* α_1 *and* α_2 *are the normal impulses from the two supporting boards while* α_1' *and* α_2' *are the approximations.*

This approximation works well in some scenarios. However, it may not be accurate in many cases [22]. A simple illustration is given in Fig. 2, where a ball is in a V-shaped slot composed of two planes in equilibrium under gravity. Given a resulting contact impulse ${}_if$, the normal impulses α_1 and α_2 from the two boards can be decomposed according to the laws of physics, as shown in Fig. 2(a). The approximation error can be computed using the projection scheme. The more the normal deviates from the direction of the total contact impulse, the larger the error.

IV. Multi-Contact Frictional Rigid Dynamics

In this section, we present our multi-contact frictional rigid dynamics algorithm.

A. Frictional Impulse

To compute the friction impulse, we use a set S_k consisting of all possible directions of the frictional impulse. The direct sum of S_k at each contact point in $se^*(3)$ forms the set ${}^iS$ of directions for total frictional impulse and can be calculated as:

$$\begin{aligned} S_k &= \{s_k | s_k \in \mathbb{R}^3, s_k^T \ n_k = 0\}, k \in A(q), \\ {}^iS_k &= M^{-1} \ {}^i\Gamma_k^T \ S_k, \\ {}^iS &= \bigoplus_{k \in A(q)} {}^iS_k, \end{aligned} \tag{8}$$

where $\bigoplus$ indicates the direct sum. Theoretically, there are infinite s_k in S_k. In practice, we use a sampling strategy

to compute S_k and typically use 4 pairs. The resulting friction computation is based on the principle of maximal dissipation [10], the Coulomb friction law, and the non-penetration constraint as follows:

$$\begin{aligned} &{}^i f = proj_{{}^i S}({}^i\phi^-), \\ &{}^i s_k^T \; M \; {}^i f \leq \mu_k \; {}^i n_k^T \; M \; {}^i r, \forall {}^i s_k \in {}^i S_k, \forall k \in A(q), \\ &{}^i n_k^T \; M \; {}^i f \geq 0, \forall k \in A(q). \end{aligned} \tag{9}$$

FFD solves the frictional impulse using Eq. (9), where ${}^i\phi^-$ is an independent quantity. Although the frictional force is not related to the relative motion, the friction computation needs to consider the relative motion at the contact point between the two objects. This is because the multi-contact solver needs to treat them as an integrated system, and not individually. If the relative motion of object i with respect to other colliding objects is neglected, this may result in incorrect results.

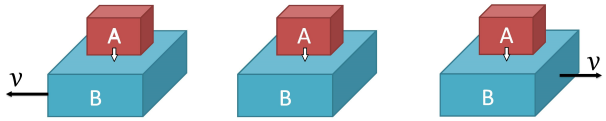


Fig. 3: *Illustrations of different frictions between two objects. Object A falls vertically onto B, but the moving status of B is quite different in each case as shown in: (a) moving left; (b) static; (c) moving right. In these cases, A and B have different relative motions.*

A scenario is illustrated in Fig. 3. A falls vertically down on B under 3 different conditions; the only difference among them is the relative motion of B. B moves horizontally to the left (a) and right (c), respectively, and is static in (b). In these cases, the frictional impulses imposed on A should be different in each case. However, since the projection scheme used in FFD is based on Eq. (8) and Eq. (9), the friction impulses that are applied to A will be the same in each case. This is because A has the same value of ${}^i\phi^-$ in each case. We present a modified algorithm to address this problem.

The overall velocity-level contact simulation of our approach is shown in Fig. 4. The method comprises of two stages: collision detection and collision response. After collision detection, the *Contact Impulse Solver* can compute the total contact impulse of each rigid body, followed by three components that are needed for our approach. First, an impulse decomposition algorithm (*Contact Distribution Solver*) is used to resolve the accurate distribution of normal contact impulse α_k at each contact point. Next, the *Frictional Impulse Solver* can compute the frictional impulse of each contact point. Finally, the *Coupling Solver* is used to compute the frictional impulse by coupling it into a convex space. Finally, the object's status is updated by the velocity integral and position integral.

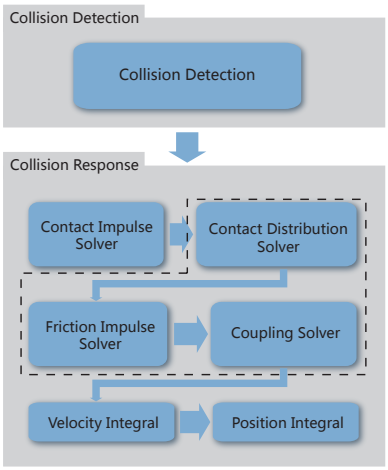


Fig. 4: *Overview of our multi-contact algorithm. The novel components are shown in dashed lines.*

B. Contact Decomposition

By using the contact impulse solver, the total contact impulse ${}^i r$ of a rigid body is computed. In order to compute the frictional impulse, we first decompose the total impulse and accurately compute its distribution on the contact points. Our approach is different from prior LCP and FFD algorithms. In the LCP method, a set of distributions is considered according to the non-penetration constraints. The complementary equations are constructed and used for resolving the distributions [2], [1]. However, in the FFD algorithm the non-penetration constraints are satisfied implicitly, as described in Section III-A, and the resulting algorithm uses the projection strategy for approximation, which may not be accurate.

In order to compute the contact distribution, we use a contact distribution solver to compute the accurate distribution of normal contact impulses using an optimization method. Given an x that includes a set of normal impulses, if the twist variation ${}^i r'$ acted by the sum of x equals the twist variation of a known total impulse ${}^i r$, then x can be treated as an accurate normal distribution of the total contact impulse.

A scalar α_k is used to represent the magnitude of the twist variation acted by the normal impulse. We convert the contact impulse into $SE(3)$ space:

$$\begin{aligned} {}^i r_k = M^{-1}{}_i r_k = M^{-1 i}\Gamma_k^T F = M^{-1 i}\Gamma_k^T n_k \alpha_k, \\ \sum_{k \in A(q)} {}^i r_k = {}^i r'. \end{aligned} \tag{10}$$

Here, ${}^i r'$ is the twist variation imposed by the distribution of normal contact impulses. Theoretically, ${}^i r'$ should be equal to ${}^i r$, which is obtained from Eq. (6). Overall, the difference between ${}^i r'$ and ${}^i r$ should be minimized:

$$x = argmin(({}^i r' - {}^i r)^T \; ({}^i r' - {}^i r)), \tag{11}$$

where $x = \begin{pmatrix} \alpha_1 & \alpha_2 & \dots & \alpha_{n_c} \end{pmatrix}^T$, n_c is the number of contact points. This equation ensures that x closely

matches the theoretical distribution by minimizing the error. According to the definition of ${}^{i}r'$, a matrix C is constructed to separate the known and unknown variables:

$$C = M^{-1} \begin{pmatrix} {}^{i}\Gamma_1^T n_1 & {}^{i}\Gamma_2^T n_2 & \dots & {}^{i}\Gamma_{n_c}^T n_{n_c} \end{pmatrix}, \tag{12}$$

where C is known and x is unknown. This minimization equation can be expressed as:

$$x = argmin(Cx - {}^{i}r)^T \ (Cx - {}^{i}r). \tag{13}$$

By solving this optimization equation, we obtain an accurate distribution of normal impulses on an object.

C. Friction at the Contact Point

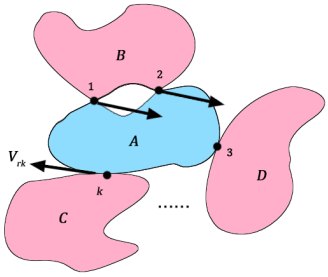


Fig. 5: *Illustration of multi-contact handling with relative motion. 1,2,...k denote the multiple contact points on object A, respectively. An arrow-line on the contact point indicates the motion (relative to its neighbor) at that point on A.*

According to the Coulomb friction law and the maximal dissipation law [10], the direction of friction tends to oppose in the tangential direction with relative motion. The magnitude of friction is no more than the product of a normal contact impulse with the friction coefficient. A complex multi-contact example is illustrated in Fig. 5, where the object A is impacted by several objects with multiple contact points, with each contact point having a different relative motion (even including the zero speed at the 3rd contact point). In the original FFD algorithm, each friction is resolved within the object itself. In this example, only A is treated as a single object, following [14]. In contrast, we treat A and its neighbors (B,C,D) as a combined system and resolve the friction at each contact point by using the relative velocity of each point.

Given a contact point k, its relative velocity V_{r_k} is used to compute the friction's direction s_k. s_k should be orthogonal to the contact normal, n_k, and opposite to the tangential relative velocity. This constraint can be satisfied when s_k is in the opposite direction of V_{r_k}'s projection on the tangent plane, which is orthogonal to the contact normal n_k. Thus s_k can be computed using the cross product operations:

$$s_k = (V_{r_k} \times n_k) \times n_k. \tag{14}$$

It turns out that directly using the mass matrix of the entire body at a contact point may not result in an accurate answer. As a result, we compute the frictional response on each contact point in a different manner. An impulse in $\mathbb{R}^3$ could affect the twist of an object, and also could change the velocity of each point on the object via:

$$\begin{aligned} {}_{i}f = {}^{i}\Gamma_k^T \ f_k, \quad {}^{i}f = M^{-1} \ {}_{i}f, \\ \Delta\dot{x}_k = {}^{i}\Gamma_k \ \Delta^{i}\phi = {}^{i}\Gamma_k \ {}_{i}f, \end{aligned} \tag{15}$$

where f_k is the external force imposed on contact point k and M is the inertia tensor. $\Delta\dot{x}_k$ represents the velocity difference after applying f_k. According to the laws of physics:

$$m_k \ s_k^T \ \Delta\dot{x}_k = f_k / s_k, \tag{16}$$

where f_k/s_k on the right side of the equation represents a pairwise division of each element in the vector f_k and s_k. We obtain an equivalent mass for a point on the object with the given force direction:

$$m_k = \frac{1}{s_k^T \ {}^{i}\Gamma_k \ M^{-1} \ {}^{i}\Gamma_k^T \ s_k}. \tag{17}$$

This equivalent mass m_k represents the velocity change of the contact point when it receives an impulse. For a mass point, we have the impulse equation $I = m_k \Delta\dot{x}_k$ that is used to compute the impulse through velocity change, where I is impulse. Similarly, we use the equivalent mass to compute the maximum frictional impulse that can counteract the relative motions:

$$f_k = -s_k^T V_{r_k} m_k. \tag{18}$$

This is the frictional impulse that can maximally dissipate the energy of motion with tangent relative velocity. According to the Coulomb friction law, the magnitude of twist variation f_k enacted by friction should be no more than the product of α_k (from contact impulse) and the friction coefficient μ. Friction formulation can be divided into kinetic friction and static friction. In kinetic friction, the magnitude of friction is equal to $\mu\alpha_k$, which can not dissipate the energy of motion with relative velocity in the tangential direction. With static friction, the friction maximally dissipates the energy of the motion and is still less than $\mu\alpha_k$. In summary, the twist variation from a frictional impulse corresponds to the smaller value between f_k and $\mu\alpha_k$. The magnitude of the frictional impulse of contact point k in object i can be given as:

$$\left\| f_k \right\|_{real} = min(\left\| f_k \right\|, \mu \ \left\| \alpha_k \right\|). \tag{19}$$

Along with the frictional impulse on each contact point, the total frictional impulse can be obtained by the following summation:

$$\begin{aligned} {}^{i}f_k = M^{-1\,i}\Gamma_k^T s_k \left\| f_k \right\|_{real}, \\ {}^{i}f' = \sum_{k \in A(q)} {}^{i}f_k. \end{aligned} \tag{20}$$

D. Coupling

Frictional impulses and normal contact impulses should be coupled in a simulation. The well-known Painleve Paradox is a good example of this coupling problem [25]. To compute the exact friction, a coupled implicit equation needs to be solved, which has a large computational cost. Approximate methods are often used to satisfy the non-penetration constraints for efficiency reasons. As it is difficult to solve the coupling problem exactly and efficiently, we present an approximate solution by using a fast projection.

In order to ensure the stability of the simulation, the total impulses should be restricted in T; as discussed in subsection III-A . ${}^i\phi^+$ is the projection on T. Therefore, it's necessary to make sure that the total frictional impulse is also restricted in T. ${}^if'$ can be projected onto T, and we obtain the twist variation ${}^if$ acted by the final frictional impulse as:

$$ {}^if = proj_T({}^if') \tag{21}$$

The final twist can be computed by the summation:

$$ {}^i\phi^+ = {}^i\phi^- + {}^ir + {}^if + \varepsilon {}^ir, \tag{22}$$

where $\varepsilon \in [0, 1]$ is the collision coefficient. According to Eq. (6), ${}^i\phi^-$ and ${}^i\phi^+ \in \partial T$, therefore ${}^ir = ({}^i\phi^+ - {}^i\phi^-) \in \partial T$, and ${}^if \in \partial T$. According to Eq. (22), all variables lie in he convex space T. As a result, the twist ${}^i\phi^+$ is restricted in T.

V. Results and Discussion

We describe our implementation and highlight the results on different benchmarks in terms of runtime performance, accuracy, and stability. All the simulation results are obtained by running the algorithm on a PC with a 3.00GHz Intel i5-2320 CPU with 4G RAM, though we also highlight GPU-based performance for some complex scenarios.

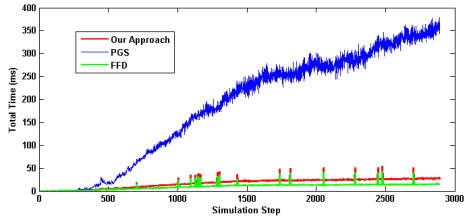
We use six challenging benchmarks to evaluate our approach. The six benchmarks and their simulation statistics are shown in Table II. Moreover, different friction phenomena are classified into static friction, kinetic friction (including slipping friction, rolling friction, rotating friction), and their coupling. These benchmarks are: Bar (Fig. 1(a)), $Stack$ (Fig. 1(b)), $Chess$ (Fig. 1(c)), $Cube$ (Fig. 1(d)), $Truck$(Fig. 7), and $Basin$ (Fig. 8). The collision detection implementation is the same as in [14], though faster algorithms are available for GPU-based proximity queries [26].

Our approach can simulate many detailed motions that occur due to friction. In the Bar benchmark, a quickly rotating cylinder moves towards the wedge and follows a parabolic trajectory after contact due to friction. Both static friction and slipping friction are simulated in our approach. In the $Stack$ benchmark, the bunny slides on a sloping surface. Five stacked boards that are in equilibrium are placed on the floor near the lower end of the slope. Three balls with different rotating orientations fall down the slope simultaneously and hit this stack of boards. $Stack$ is a rather complicated benchmark that is used to evaluate the stability of a friction solver, as discussed in [29]. Without accurate static friction, the stack may be in unstable equilibrium and collapse. In our simulation, the equilibrium of this structure is maintained at the beginning, and the stack only collapses after the balls and the bunny hit on it.

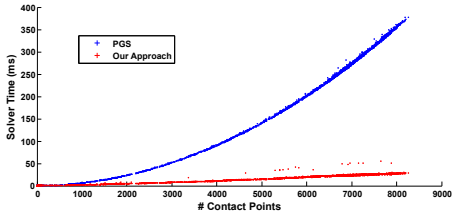
A. Performance Analysis

A key issue is the computational cost of the contact response algorithm. In particular, many applications such as haptics or virtual environments desire interactive performance. Some iterative solvers like PGS can achieve a higher speed with fewer iterations, but the accuracy can be low. FFD can be fast [14], but it fails to compute accurate contact impulse distribution. Our approach retains the advantages of both PGS and FFD with high accuracy and low runtime cost. The overall running time is comparable to FFD, but our contact resolution algorithm is more stable.

We use the $Chess$ benchmark to compare the efficiency because it has a large number of contact points during each timestep. We compare the performance of three algorithms based on FFD, LCP with PGS solver, and our new algorithm on the same configurations.



(a) Total timecost



(b) Solver timecost

Fig. 6: *Performance comparisons between FFD, LCP using PGS solver (labeled as PGS), and our approach on the Chess benchmark.*

The difference of the running times is shown in Figure 6(a). PGS is far costlier than our approach and FFD. A typical iteration count for PGS is five iterations [29]. In the same benchmark, FFD is much faster than LCP using PGS. Our approach is slightly slower than FFD, but still offers interactive performance on these complex benchmarks. Compared to FFD, our algorithm spends extra time in contact distribution computation, which improves the accuracy. In order to simulate such a scenario (with 8K contacts), PGS based on five iterations takes about 360 ms per timestep, while our approach only takes 28 ms per timestep.

Scenario	#Objects	#Triangles	# Contacts	Running Time	Static Friction	Kinetic Friction		
						Slipping	Rolling	Rotating
Bar	3	2K	120	2 ms	•	•	×	×
Stack	11	13K	344	5ms	•	•	•	×
Chess	321	236K	8K	28 ms	•	•	•	×
Cube	5001	120K	60K	50 ms	•	•	×	×
Truck	7	19K	52	4 ms	•	•	•	×
Basin	685	1750K	2K	39 ms	•	•	•	•

TABLE II: *Benchmarks. #Objects refers to the number of 3D rigid objects; # Triangles refers to the number of triangles in the scene; # Contacts refers to the maximum number of simultaneous contact points that occur during the simulation; Running Time refers to the computation time for the configuration corresponding to the maximum number of contact points. The circle denotes that those phenomena are present in that benchmark, while the cross implies that they are not.*

We use the $Cube$ benchmark to evaluate the performance in more complex contact configurations. The number of contact points is about 60K. In order to simulate at interactive rates, we use GPU acceleration based on simple parallelization. We run our algorithm on an NVIDIA GTX550 TI with 192 cores and our solver takes 50 ms per frame. These benchmarks demonstrate that our approach has the advantages of both PGS and FFD. By maintaining the speed advantages of FFD, our approach has a small runtime overhead. However, ours is much faster than LCP using PGS.

B. Stability Analysis

Stability is another important feature for rigid dynamics. Accuracy can partially reflect stability, but is not a sufficient criterion. Stability often means that the method will not crash or result in jitters during the simulation of a complex scene with a large number of objects and contacts. Many techniques work well on simple benchmarks, e.g. penalty-based methods, but may not work in complex scenarios with a high number of contact points. Therefore, penalty-based methods are less stable.

The $Cube$ benchmark is a challenging task because 5000 cubes are stacked and pressed against each other. The figure and accompanying video show that our approach is stable on this benchmark and can correctly handle all the contacts.

C. Detailed Frictional Phenomena

Fig. 7: *Truck: Chess pieces fall onto a moving truck with a sudden stop to demonstrate the relative motion. Ours works well on this scenario, though FFD would not.*

In the $Truck$ benchmark, we demonstrate the improvement obtained by our algorithm over FFD in terms of frictional response due to the relative motion, shown in Fig. 7 and the accompanying video. The scenario is designed as follows: chess pieces fall onto a moving truck model that has a sudden stop. In this case, the chess pieces have backward velocities relative to the truck, which generate forward frictional impulses with a friction coefficient $\mu = 0.4$. With the exertion of frictional force, these pieces will turn over, rotate, or slip on the truck. These chess pieces will continue to move when the truck suddenly stops due to inertia, and all of them finally come to rest after their kinetic energies dissipate due to friction. The FFD algorithm cannot reproduce correct simulation results because it fails to handle the friction caused by the relative motion. This results in the frictional impulse being zero because a chess piece has no horizontal velocity.

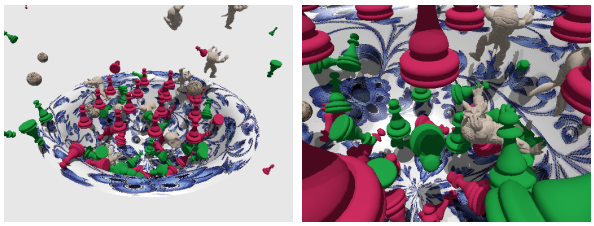


Fig. 8: *Basin: a snap shot and close-up view.*

In the $Basin$ benchmark, hundreds of 3D objects such as chess pieces, balls, and elephant and armadillo models (undergoing self-rotations) successively fall into the spin basin with varying angular velocities and bounce back. Moreover, they bounce due to frictional collision with the spin basin or other objects. Several types of friction phenomena can be observed in this benchmark. This benchmark also demonstrates that the relative motions and the angular velocities between objects are modeled and handled accurately using our approach. This way, we can simulate the entire motion in which the falling objects collide with the spin basin, rotate along with the basin with friction, and are thrown out of the basin when the friction forces cannot hold these objects any more.

VI. Conclusions and Future Work

We present a novel approach for modeling multi-contact friction for rigid dynamics using impulse decomposition. Our approach first computes the distribution of normal contact impulses using decomposition, evaluates the exact frictional impulse at each contact point, and finally couples them together. We have evaluated our approach's performance on many complex rigid body simulations with thousands of contact points. Its runtime performance is almost comparable

to FFD and it can be used for interactive applications. We observe higher accuracy over FFD.

There are many avenues for future work. We would to handle breaking objects and articulated models. If we can extend the approach to satisfy Newton's 3rd law of motion, we would like to use it for haptic rendering. We could also develop improved parallel algorithms and propose a more complete GPUs-based solution [9] for contact handling in large models.

ACKNOWLEDGEMENTS

The work is supported by Grants Nos. 61232014, 61421062, 61472010 from NSFC of China and Grant No. 2017YFB0203002 from National Key Research and Development Program of .

REFERENCES

[1] Mihai Anitescu and Florian A Potra. Formulating dynamic multi-rigid-body contact problems with friction as solvable linear complementarity problems. *Nonlinear Dynamics*, 14(3):231–247, 1997.

[2] David Baraff. Analytical methods for dynamic simulation of non-penetrating rigid bodies. In *ACM SIGGRAPH Computer Graphics*, volume 23, pages 223–232. ACM, 1989.

[3] Jan Bender, Kenny Erleben, and Jeff Trinkle. Interactive simulation of rigid body dynamics in computer graphics. In *Computer Graphics Forum*, volume 33, pages 246–270. Wiley Online Library, 2014.

[4] Bernard Brogliato. *Nonsmooth mechanics: models, dynamics and control*. Springer, 2016.

[5] A. Chatterjee and A. L. Ruina. Two interpretations of rigidity in rigid body collisions. *Journal of Applied Mechanics*, 65(4):894–900, 1998.

[6] Christian Duriez, Frederic Dubois, Abderrahmane Kheddar, and Claude Andriot. Realistic haptic rendering of interacting deformable objects in virtual environments. *IEEE transactions on visualization and computer graphics*, 12(1):36–47, 2006.

[7] R Elliot English, Michael Lentine, and Ron Fedkiw. Interpenetration free simulation of thin shell rigid bodies. *IEEE transactions on visualization and computer graphics*, 19(6):991–1004, 2013.

[8] Kenny Erleben. Velocity-based shock propagation for multibody dynamics animation. *ACM Transactions on Graphics (TOG)*, 26(2):12, 2007.

[9] Naga K Govindaraju, Ming C Lin, and Dinesh Manocha. Quick-cullide: Fast inter-and intra-object collision culling using graphics hardware. In *Virtual Reality, 2005. Proceedings. VR 2005. IEEE*, pages 59–66. IEEE, 2005.

[10] Suresh Goyal, Andy Ruina, and Jim Papadopoulos. Planar sliding with dry friction part 1. limit surface and moment function. *Wear*, 143(2):307–330, 1991.

[11] Eran Guendelman, Robert Bridson, and Ronald Fedkiw. Nonconvex rigid bodies with stacking. In *ACM Transactions on Graphics (TOG)*, volume 22, pages 871–878. ACM, 2003.

[12] James K Hahn. Realistic animation of rigid bodies. In *ACM SIGGRAPH Computer Graphics*, volume 22, pages 299–308. ACM, 1988.

[13] Kris Hauser. Robust contact generation for robot simulation with unstructured meshes. In *Robotics Research*, pages 357–373. Springer, 2016.

[14] Danny M Kaufman, Timothy Edmunds, and Dinesh K Pai. Fast frictional dynamics for rigid bodies. *ACM Transactions on Graphics (TOG)*, 24(3):946–956, 2005.

[15] Danny M Kaufman, Shinjiro Sueda, Doug L James, and Dinesh K Pai. Staggered projections for frictional contact in multibody systems. *ACM Transactions on Graphics (TOG)*, 27(5):164, 2008.

[16] Caishan Liu, Zhen Zhao, and Bernard Brogliato. Frictionless multiple impacts in multibody systems. i. theoretical framework. In *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, volume 464, pages 3193–3211. The Royal Society, 2008.

[17] Per Lötstedt. Numerical simulation of time-dependent contact and friction problems in rigid body mechanics. *SIAM journal on scientific and statistical computing*, 5(2):370–393, 1984.

[18] Brian Vincent Mirtich. *Impulse-based dynamic simulation of rigid body systems*. PhD thesis, University of California at Berkeley, 1996.

[19] Jean Jacques Moreau. Quadratic programming in mechanics: dynamics of one-sided constraints. *SIAM Journal on control*, 4(1):153–158, 1966.

[20] Jean Jacques Moreau and Panagiotis D Panagiotopoulos. *Nonsmooth mechanics and applications*, volume 302. Springer, 2014.

[21] Katta G Murty and Feng-Tien Yu. *Linear complementarity, linear and nonlinear programming*. Citeseer, 1988.

[22] Valentin Popov. *Contact mechanics and friction: physical principles and applications*. Springer Science & Business Media, 2010.

[23] Breannan Smith, Danny M Kaufman, Etienne Vouga, Rasmus Tamstorf, and Eitan Grinspun. Reflections on simultaneous impact. *ACM Transactions on Graphics (TOG)*, 31(4):106, 2012.

[24] Jonas Spillmann, Markus Becker, and Matthias Teschner. Non-iterative computation of contact forces for deformable objects. 2007.

[25] David E Stewart. Rigid-body dynamics with friction and impact. *SIAM review*, 42(1):3–39, 2000.

[26] A. Sud, N. Govindaraju, R. Gayle, I. Kabul, and D. Manocha. Fast proximity computation among deformable models using discrete voronoi diagrams. *ACM Transactions on Graphics (TOG)*, 25(3):1144–1153, 2006.

[27] Min Tang, Dinesh Manocha, Miguel A Otaduy, and Ruofeng Tong. Continuous penalty forces. *ACM Trans. Graphics*, 31(4):107:1–107:9, 2012.

[28] Emanuel Todorov. A convex, smooth and invertible contact model for trajectory optimization. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 1071–1076. IEEE, 2011.

[29] Richard Tonge, Feodor Benevolenski, and Andrey Voroshilov. Mass splitting for jitter-free parallel rigid body simulation. *ACM Transactions on Graphics (TOG)*, 31(4):105, 2012.

[30] Tamer M Wasfy and Ahmed K Noor. Computational strategies for flexible multibody systems. *Applied Mechanics Reviews*, 56(6):553–613, 2003.

[31] Hongyi Xu, Yili Zhao, and Jernej Barbic. Implicit multibody penalty-baseddistributed contact. *IEEE transactions on visualization and computer graphics*, 20(9):1266–1279, 2014.

[32] Katsu Yamane and Yoshihiko Nakamura. Stable penalty-based model of frictional contacts. In *Robotics and Automation. Proceedings IEEE International Conference on*, pages 1904–1909. IEEE, 2006.

[33] Tianxiang Zhang, Sheng Li, Dinesh Manocha, Guoping Wang, and Hanqiu Sun. Quadratic contact energy model for multi-impact simulation. In *Computer Graphics Forum*, volume 34, pages 133–144. Wiley Online Library, 2015.