Workshop on Virtual Reality Interaction and Physical Simulation VRIPHYS (2014)
J. Bender, C. Duriez, F. Jaillet, and G. Zachmann (Editors)

Parallel Particles (P^2): A Parallel Position Based Approach for Fast and Stable Simulation of Granular Materials

D. Holz

CM Labs Simulations, Canada

Abstract

Granular materials exhibit a large number of diverse physical phenomena which makes their numerical simulation challenging. When set in motion they flow almost like a fluid, while they can present high shear strength when at rest. Those macroscopic effects result from the material's microstructure: a particle skeleton with interlocking particles which stick to and slide across each other, producing soil cohesion and friction. For the purpose of Earthmoving equipment operator training, we developed Parallel Particles (P^2), a fast and stable position based granular material simulator which models inter-particle friction and adhesion and captures the physical nature of soil to an extend sufficient for training. Our parallel solver makes the approach scalable and applicable to modern multi-core architectures yielding the simulation speed required in this application. Using a regularization procedure, we successfully model visco-elastic particle interactions on the position level which provides real, physical parameters allowing for intuitive tuning. We employ the proposed technique in an Excavator training simulator and demonstrate that it yields physically plausible results at interactive to real-time simulation rates.

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Physically Based Modeling

1. Introduction

Recently, the training of operators of heavy machinery, such as cranes, excavators and bulldozers, with Virtual Reality (VR) training simulators has become more and more common. In general, simulations have to run at interactive to real-time rates and must be physically plausible to prevent negative training. Especially in the context of Earthmoving equipment simulation this is a challenging task, since the machine interacts with a deformable medium: soil. The non-linear physical nature of soil gives rise to a number of dynamic effects including fluid-like flows, high compressibility, non-linear cutting forces, sudden material slip, internal friction and cohesion yielding a wide range of emerging angles of repose as well as stable slope angles. These effects must be captured as accurately as possible in a training simulator and at simulation rates which are close to real-time. Furthermore, due to the human in the loop, interactions between machine and soil medium are arbitrary and must be stable at all times. Operators use their equipment to push, lift and dump soil. For each of these operations the operator needs to learn specific skills and techniques, devised to correctly interact with soil. For example, very cohesive soils require the operator to perform accelerating bucket motions to make the soil fully escape the bucket. A bulldozer blade must be lifted in a particular way to prevent the blade from being sucked downwards into the ground, creating a cutting force too high for the bulldozer to advance. The more precise and the faster the operator is able to perform those maneuvers, the more efficient is he in accomplishing his tasks. Apart from operator safety, improving the efficiency of operators in the field is one of the main goals of training. Consequently, a real-time granular material simulation should have the following qualities:

- simulation stability
- good performance
- can model frictious and cohesive materials
- provides plausible soil dynamics
- enables interaction with rigid bodies, such as a bucket

DOI: 10.2312/vriphys.20141232

- good scalability on modern architectures

For the purpose of successful training of Earthmoving equipment operators, we developed a fast and stable granular material simulator – Parallel Particles (P^2) – which is based on the framework of Position Based Dynamics (PBD) [MHHR06], and fulfills all of the aforementioned requirements. We employ a Discrete Element Method (DEM) which explicitly models inter-particle contacts as opposed to Continuum Mechanics approaches which treat the material as a continuum [LL03]. Our model considers particle positions only, omitting rotations. We find that this still allows to produce physically plausible results and at the same time provides potential for a reduction in processing time. Dealing only with positions leads to simpler formulations and a smaller memory footprint which results in better performance. We decided to use the PBD approach for several reasons. It is first of all unconditionally stable, fulfilling already one of the requirements. Second of all, omitting orientations and modeling particle positions only, suffices to create realistic cohesive and frictional effects. In fact, oriented particles, require modeling of either rolling friction or interlocking, e.g., via particle angularity, to allow for stable piles. This is not the case for purely positional particles. Sliding friction alone suffices to stabilize a particle on a slope. With a spherical, oriented particle, sliding friction merely leads to rolling which makes one of the above techniques necessary to prevent it from rolling down an even lightly inclined slope. Therefore, with positional particles, less parameters are required to achieve the desired soil dynamics effects which makes simulation tuning easier. Our contributions are

- a novel, parallel solver for general PBD problems
- a visco-elastic position-based contact constraint, equivalent to a spring-damper, used to model stable granular materials
- Coulomb friction formulated in the PBD framework combined with adhesion, enabling stable heaps of granular material
- simple error reduction techniques which improve solver convergence

In the following, we will first present related work. Section 3 then describes the original PBD approach and our modifications to it. The following sections provide details on the proposed method, including our parallel solver, the particle contact model and coupling with rigid bodies, as well as error reduction techniques to improve convergence. Section 8 provides simulation results. We conclude with Section 9 and discuss future work.

2. Related Work

Bell et al. propose a DEM method for the representation of granular materials with friction but no adhesion in [BYM05]. The authors use the concept of composite particles to simulate angular grains, and allow thus for particle interlocking in their simulations. Their integration scheme is different to ours and is not suitable for real-time applications.

The method presented in [IWT12] is a meshfree approach which uses predictive-corrective Smoothed Particle Hydrodynamics (PCISPH) to simulate granular material with friction and cohesion. The method yields very good results but is not used in a real-time context.

A granular material simulation using regularized Projected Gauss-Seidel is proposed by Servin et al. [SWLB14]. The authors show the benefits of non-smooth vs. smooth approaches, and propose a hybrid method. No timing results are provided in this work. Unfortunately, Gauss-Seidel is not efficiently parallelizable as pointed out by Tonge et al. [TBV12]. The graph coloring approach to split the work is a NP hard problem and requires use of a greedy algorithm for an acceptable complexity. This increases the number of needed colors, making the load balancing suboptimal and hence limiting the speed-up. This problem occurs especially with many constraints per rigid body, which is precisely the case in granular materials.

Tonge et al. suggest splitting of rigid bodies to derive a parallel solver for the solution of Rigid Body Linear Complementarity Problems (LCP) in [TBV12]. Their approach enables jitter-free, parallel rigid body dynamics. We inspired our parallel solver from this work.

The approach proposed by Macklin et al. in [MMCK14], not yet published at the time of writing, shows strong similarities to our method, but was developed independently. The authors also present a parallel Position Based Dynamics solver. They show that the approach is suitable for simulation of various materials, while we focus on contact modeling for granular materials only.

3. Position Based Dynamics

The proposed granular material simulation approach is based on the Position Based Dynamics (PBD) framework introduced to the Computer Graphics community by Müller et al. [MHHR06], which is outlined in this section. In this method, which is intended mostly for simulation of deformable bodies such as cloth but has also been successfully applied to the simulation of other materials [BMOT13] [MMCK14], a deformable body is represented by a set of point masses with positions $\mathbf{x}_i$, velocities $\mathbf{v}_i$ and masses m_i. The point masses are connected by constraints, such as distance, stretching and bending constraints to form, e.g., a cloth. A constraint between n points is represented by a function $C : \mathbb{R}^{3n} \rightarrow \mathbb{R}$, formulated on the position level, which can be either an equality or an inequality:

$$\begin{aligned} C(\mathbf{x}_{i_1}, \ldots. \mathbf{x}_{i_n}) &= 0 \\ C(\mathbf{x}_{i_1}, \ldots. \mathbf{x}_{i_n}) &\geq 0 \end{aligned} \tag{1}$$

Inequalities can be used to model unilateral contacts, while for a distance constraint an equality would be used.

The dynamics of the deformable body are simulated in a predictor-corrector fashion as follows. At the beginning of each step, the unconstrained positions $\mathbf{p}_i$ of all points are computed by performing a semi-implicit Euler integration based on current velocities and external forces, $\mathbf{f}_e$, such as gravity:

$$\mathbf{p}_i = \mathbf{x}_i + \Delta t \mathbf{v}_i + \Delta t^2 \frac{\mathbf{f}_e}{m_i} \tag{2}$$

Note that in the original approach the predicted velocities are first damped for stability reasons before they are used to produce the predicted positions, which is not necessary in our method.

The predicted, unconstrained positions are likely causing constraints to be violated at this point, and must therefore be corrected to viable target positions which do not violate the constraints at the end of the step. This is realized by a correction phase. Before any positions are corrected, collision detection is performed on the predicted positions yielding contact constraints which are represented by inequalities. All constraints, both permanent and contact constraints, are then processed by the solver. The original work suggests an iterative Gauss-Seidel style approach, with which the solver corrects the predicted positions by processing all constraints sequentially. This process is repeated for a fixed number of iterations yielding the final corrected positions $\mathbf{p}_i$. In a subsequent integration phase, the new velocities and positions are computed as follows:

$$\begin{aligned} \mathbf{v}_i &\leftarrow (\mathbf{p}_i - \mathbf{x}_i)/\Delta t \\ \mathbf{x}_i &\leftarrow \mathbf{p}_i \end{aligned} \tag{3}$$

As pointed out by Müller et al. [MHHR06], a big advantage of this approach is that the integration scheme is unconditionally stable. This makes it a very good fit for application in a VR training simulator. In the original approach, the integration phase follows a velocity update, in which velocities are modified in order to mimic contact dynamics effects such as friction. We omit this phase in our approach, since we directly embed friction into the position corrections performed by the solver (see Section 6.5.1).

3.1. Position Correction

During the correction phase, individual point position corrections, which are combined by the solver, are computed using constraint projection as follows. In a given solver iteration, let's assume that a n-point constraint C is violated, that is $C(\mathbf{p}_1,..\mathbf{p}_n) \neq 0$. For simplicity, we will assume that C is an equality constraint w.l.o.g. Inequality constraints are handled by the solver by simply not performing any position correction unless the inequality is violated. So, all concepts discussed below directly apply also to inequality constraints. Defining the vector of positions as $\mathbf{p} = (\mathbf{p}_1,...,\mathbf{p}_n)$, Müller et al. [MHHR06] want to find a vector of position corrections, $\Delta\mathbf{p}$, which no longer violates the constraint function:

$$C(\mathbf{p} + \Delta\mathbf{p}) = 0 \tag{4}$$

Using a Taylor series expansion and dropping the non-linear error terms, they linearize the constraint C around $\mathbf{p}$ as

$$0 = C(\mathbf{p} + \Delta\mathbf{p}) \approx C(\mathbf{p}) + \nabla C(\mathbf{p})\Delta\mathbf{p}, \tag{5}$$

where $\nabla C(\mathbf{p})$ denotes the gradient of C as a row vector. The authors of [MHHR06] restrict the position correction to lie along the constraint gradient, turning it into a scalar β, and weight the position corrections by the inverse masses of the individual points, $w_i = m_i^{-1}$, yielding

$$\Delta\mathbf{p} = \beta \mathbf{W} \nabla C(\mathbf{p})^T, \tag{6}$$

where $\mathbf{W} = diag(w_1,...,w_n)$ denotes the inverse mass matrix. By combining Equations 5 and 6, Müller et al. derive the following formula for the correction of the individual point $\mathbf{p}_i$ due to the effect of constraint C:

$$\begin{aligned} \Delta\mathbf{p}_i &= -s w_i \nabla_{\mathbf{p}_i} C(\mathbf{p}), \\ s &= \frac{C(\mathbf{p})}{\sum_{j=1}^{n} w_j |\nabla_{\mathbf{p}_j} C(\mathbf{p})|^2}, \end{aligned} \tag{7}$$

Here, $\nabla_{\mathbf{p}_i} C(\mathbf{p})$ denotes the gradient of C with respect to point $\mathbf{p}_i$.

For the simulation of compliant constraints, Müller et al. suggest a simple model in which the position correction $\mathbf{p}_i$ is scaled by a stiffness factor k_{pb}, reducing the magnitude of the position correction, and, thus relaxing the constraint. Applying the position correction yields the new, corrected position:

$$\mathbf{p}_i \leftarrow \mathbf{p}_i + k_{pb} \Delta\mathbf{p}_i \tag{8}$$

We replace this simple compliance model in Section 6. This section also describes how Equation 7 can be used to formulate contact constraints which model visco-elastic particle interactions together with friction and adhesion.

4. Method Overview

We adopt the Position Based Dynamics approach in our DEM soil simulation by considering the point masses as individual, spherical particles with radii r_i. Our collision detection produces inter-particle and rigid body/particle contact constraints using the predicted particle positions. Most importantly, we make the following modifications.

First, we do not use the original sequential Gauss-Seidel style solver but propose a parallel solver scheme. The main idea is to process particles independently, by computing all constraint projections of a particle and averaging them to obtain the final particle position for the iteration.

Second, we do not employ the simplified friction model proposed in the original approach, which is based on damping velocities. This model is not capable of simulating stable piles. We also drop the simplified original compliant contact model, which is purely elastic and has no potential for tuning contact dampness – a factor which is very important in a DEM simulation. Instead we model real Coulomb friction together with adhesion. Also, we handle inter-particle collisions via a relaxed, unilateral contact constraint, which formulates a spring-damper on the position level and creates visco-elastic particle interactions, allowing for tuning via real physical stiffness and damping coefficients.

Finally, we suggest error reduction techniques which create physically plausible results even with a low number of solver iterations. All these modifications are presented in the following sections.

5. Parallel Constraint Solver

The proposed parallel constraint solver is motivated by the mass-splitting approach taken by [TBV12], who split rigid bodies by the number of their interactions, creating split masses. All contacts involved in one interaction are solved in parallel using Projected Gauss Seidel; assuming a fixed joint between all split bodies, the final velocity is computed as the average of all split body velocities. We follow a similar approach. In a given solver iteration, we process all particles in parallel and compute the corrected particle positions at the end of the iteration through constraint averaging as follows. For each predicted particle position $\mathbf{p}_i$, we compute the n position corrections, $\Delta\mathbf{p}_{i,j}$, caused by the n constraints C_j which the particle violates in this iteration, and we set the final position correction for this iteration, $\Delta\mathbf{p}_i$, to the average of the individual corrections. Applying this correction to the current particle position yields the new, corrected particle position, $\mathbf{p}_i$, at the end of this iteration:

$$\Delta\mathbf{p}_i = \frac{1}{n}\sum_{j=1}^{n}\Delta\mathbf{p}_{i,j} \qquad (9)$$
$$\mathbf{p}_i \leftarrow \mathbf{p}_i + \Delta\mathbf{p}_i$$

This treatment produces a data-parallel problem and allows for processing particles in parallel with excellent load balancing, as follows. At the beginning of the correction phase, all particles are assigned to different threads. In a given solver iteration, all particles are then processed in parallel. A thread will process all its assigned particles one by one. For each particle i, it computes the individual position corrections $\Delta\mathbf{p}_{i,j}$ and combines them using Equation 9 to move the particle to the new, corrected position at the end of the given iteration. The solver will perform a fixed number of iterations in this way, usually between 5 and 10 for real-time applications, yielding the final particle positions and velocities at the end of the step through Equation 3. No convergence check is performed because with so few iterations it can not be reached. However, the results are physically plausible and suffice for our application. It is noted that the proposed solver is a generic solver which is not limited to the simulation of granular materials, but can be applied to other position-based problems as well.

The solver parallelization leads to a significant speed-up, as shown in Figure 1. We further reduce processing time by sorting particles by their occurrence in space, using their hash grid id (which becomes also the particle id) as the sort key. Each thread then processes contacts in lexicographical order of their two particle ids which produces a spatially and temporally local memory access pattern. Processing contacts in this way increases cache coherence and therefore reduces memory access times significantly.

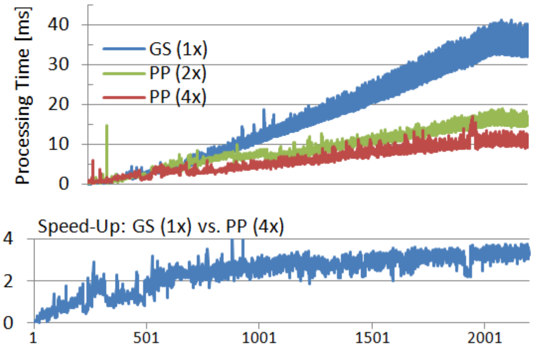

Figure 1: *Performance of our parallel solver (PP) with 2 and 4 threads compared to single-threaded Gauss-Seidel style solver (GS). 2000 particles are successively poured to form a pile (cf. Fig. 3), using 10 solver iterations. Timings include collision detection.*

6. Contact Model

The original constraint projection function, which employs a normalized stiffness parameter for relaxation, can yield very stiff interactions if the number of solver iterations is correspondingly high. It allows simulating only elastic interactions, which is problematic for the simulation of granular materials with large timesteps. Small penetration errors, e.g., due to tunnelling problems in the collision detection can cause high reaction forces, which in turn can lead to instabilities. Müller et al. circumvented this problem by applying global damping on the velocity level [MHHR06]. In this section, we will present a visco-elastic contact model with friction and adhesion, formulated as position-based constraints. A simple, yet effective regularization approach allows modeling the normal component of particle contacts as spring-damper, which enables intuitive and physically motivated relaxation and damping of particle interactions. This makes system-wide damping to achieve stable simulations unnecessary.

6.1. Elastic Constraint (Spring)

Following the recipe from Erin Catto in [Cat11], we will first show that a single distance constraint in the original PBD approach is equivalent to a spring, integrated with implicit Euler. W.l.o.g. we will continue the discussion in one dimension and examine a single mass particle attached by a spring with stiffness coefficient k to some fixed reference point under zero gravity. The equation of motion for this spring-mass system is

$$m\ddot{x} + kx = 0, \tag{10}$$

with particle position x and mass m. Assuming initial position x_0, velocity v_0 and step size Δt, and introducing velocity $v = \dot{x}$, this second-order ordinary differential equation (ODE) can be numerically approximated by performing two successive implicit Euler integrations, leading to the following sequence of positions and velocities

$$\begin{aligned} v_{r+1} &= v_r - \Delta t \frac{k}{m} x_{r+1} \\ x_{r+1} &= x_r + \Delta t v_{r+1} \end{aligned} \tag{11}$$

where r denotes the r'th element in the sequence. Solving for velocity, we obtain

$$v_{r+1} = \frac{1}{1+\Delta t^2 \frac{k}{m}} v_r - \frac{\Delta t \frac{k}{m}}{1+\Delta t^2 \frac{k}{m}} x_r. \tag{12}$$

Let's do the same for a position-based spring (a distance constraint with zero target distance), and assuming only one solver iteration. First, the method performs a prediction step, yielding the predicted position p via one semi-implicit Euler integration, which is then corrected. The spring constraint function in our example reduces to $C(p) = p \overset{!}{=} 0$, and according to Equation 7 the position correction is simply $-p$, and it is relaxed by the stiffness factor k_{pb} according to Equation 8. Finally, the new velocity is computed using Equation 3. Altogether this produces the following sequence:

$$\begin{aligned} p &= x_r + \Delta t v_r \\ x_{r+1} &= p - k_{pb} p = (1 - k_{pb}) p \\ v_{r+1} &= \frac{x_{r+1} - x_r}{\Delta t} \end{aligned} \tag{13}$$

Solving for velocity leads to

$$v_{r+1} = (1 - k_{pb}) v_r - \frac{k_{pb}}{\Delta t} x_r. \tag{14}$$

Performing coefficient matching between Equations 12 and 14, we obtain the following relationship between the spring coefficient k and the position-based stiffness factor k_{pb}:

$$k_{pb} = \frac{\Delta t^2 \frac{k}{m}}{1+\Delta t^2 \frac{k}{m}} \tag{15}$$

This shows that a single position-based spring constraint is equivalent to a spring integrated with implicit Euler, as also demonstrated in Figure 2. As pointed out by Müller et al. [MHHR06] the stiffness factor k_{pb} must be made independent of the number of solver iterations, and the authors provide the corresponding formula. We will face the same problem when deriving the combined position-based spring-damper constraint in Section 6.3.

6.2. Viscous Constraint (Damper)

We can observe that Equation 15 gives the simple, normalized stiffness factor proposed by Müller et al. a physical meaning, relating it to a spring. In the following, we will first introduce a viscous constraint in the position-based framework which will ultimately lead to a full spring-damper constraint for visco-elastic particle interactions.

A damper dissipates kinetic energy, therefore reducing the relative velocity between two constrained particles. Velocity in the position-based approach results from the difference between the previous particle position and the new, predicted and corrected particle position. That is, according to Equation 3, particle velocities emerge implicitly during the iterative constraint correction process performed by the solver as

$$\mathbf{v}_{i,j} = \frac{\mathbf{p}_{i,j} - \mathbf{x}_i}{\Delta t}, \tag{16}$$

where $\mathbf{v}_{i,j}$ denotes the velocity of particle i after a correction caused by constraint C_j which yields the corrected position $\mathbf{p}_{i,j}$. Here, $\mathbf{x}_i$ denotes the particle position from the last frame.

A viscous damper constraint between two colliding particles i and j, acting along the contact normal $\mathbf{n} = \frac{(\mathbf{p}_i - \mathbf{p}_j)}{|(\mathbf{p}_i - \mathbf{p}_j)|}$, can thus be formulated as

$$C_d(\mathbf{p}_i, \mathbf{p}_j) = ((\mathbf{p}_i - \mathbf{x}_i) - (\mathbf{p}_j - \mathbf{x}_j)) \cdot \mathbf{n} \overset{!}{=} 0. \tag{17}$$

Again, the effect of the constraint is relaxed by scaling the corresponding position correction (see Eq. 7) by a factor, referred to as c_{pb} to avoid confusion with k_{pb}, which we use exclusively for the spring constraint.

6.3. Visco-Elastic Constraint (Spring-Damper)

In order to model a spring-damper, we could now simply create both a damper and a spring constraint between each pair of contacting particles, which would model exactly the parallel nature of the Kelvin-Voigt spring-damper model. However, this would require processing more constraints in the solver and would therefore reduce performance and yield worse convergence.

To prevent this, we will instead combine the spring and damper constraint types in a single spring-damper constraint and make it again equivalent to the implicit Euler case for convenient parametrization, following the procedure in Section 6.1. However, we will re-utilize the damper constraint later to model viscous sliding friction and adhesion.

As before, w.l.o.g. we will proceed in one dimension. A single mass particle attached by a spring-damper with stiffness coefficient k and damping coefficient c under zero gravity has the following equation of motion:

$$m\ddot{x} + c\dot{x} + kx = 0 \tag{18}$$

The sequence of particle positions and velocities can be approximated using implicit Euler integration as

$$\begin{aligned} v_{r+1} &= v_r - \Delta t \frac{c}{m} v_{r+1} - \Delta t \frac{k}{m} x_{r+1} \\ x_{r+1} &= x_r + \Delta t v_{r+1} \end{aligned} \tag{19}$$

We now combine the position-based spring constraint and the damper constraint from Equation 17 to a single constraint by performing both the spring and the damper correction at the same time. We recall that p denotes the predicted position. According to Equation 7, the damper position correction in this simple case is $(p - x_r)$, which we scale by the damping factor c_{pb} from Section 6.2. Again assuming only one solver iteration, this yields the following time stepping scheme:

$$\begin{aligned} p &= x_r + \Delta t v_r \\ x_{r+1} &= p - k_{pb} p - c_{pb}(p - x_r) \\ &= (1 - k_{pb} - c_{pb})p + c_{pb} x_r \\ v_{r+1} &= \frac{x_{r+1} - x_r}{\Delta t} \end{aligned} \tag{20}$$

Solving for v_{r+1} in both the position-based and the implicit Euler time stepping scheme as in Section 6.1, and performing coefficient matching, we obtain the following relationship between the stiffness and damping coefficients k and c, and the position-based stiffness and damping factors k_{pb} and c_{pb}:

$$\begin{aligned} k_{pb} &= \frac{\Delta t^2 \frac{k}{m}}{1 + \Delta t \frac{c}{m} + \Delta t^2 \frac{k}{m}} \\ c_{pb} &= \frac{\Delta t \frac{c}{m}}{1 + \Delta t \frac{c}{m} + \Delta t^2 \frac{k}{m}} \end{aligned} \tag{21}$$

As we can see, the position-based relaxation factors k_{pb} and c_{pb} are now coupled. They both depend on the stiffness coefficient k and the damping coefficient c, which shows that the effect of the spring and the damper are in fact now combined. If we set the damping coefficient c to zero, we will obtain again Equation 15, i.e., a pure spring which does not depend on c. Analogously, we obtain the formula for a pure position-based damper if k is zero.

Now, that we know how to scale the position corrections of the position-based spring and damper constraints to obtain combined spring-damper behavior in the simple one-dimensional case, we can apply our knowledge to the general case of an inter-particle contact, modeled as spring-damper in the normal direction. The constraint function for a spring between particles i and j with radii r_i and r_j respectively is provided as

$$C_s(\mathbf{p}_i, \mathbf{p}_j) = (\mathbf{p}_i - \mathbf{p}_j) \cdot \mathbf{n} - (r_i + r_j) \overset{!}{=} 0. \tag{22}$$

We apply the position correction from both the spring constraint C_s and the damper constraint C_d from Equation 17 to particle i and scale it by the stiffness and damping factors k_{pb} and c_{pb}. The position correction for particle i due to a collision with particle j is therefore

$$\Delta\mathbf{p}_{i \leftarrow j} = k_{pb} \Delta\mathbf{p}_{i,s} + c_{pb} \Delta\mathbf{p}_{i,d}, \tag{23}$$

where $\Delta\mathbf{p}_{i,\alpha}$ denotes the position correction of $\mathbf{p}_i$ due to constraint C_α computed via Equation 7. Correction of particle position $\mathbf{p}_j$ follows analogously. In order to model the contact correctly, a position correction is only performed if the particles do in fact collide, that is, $C_s \leq 0$, which is the same as turning the constraint into an inequality. Note that the gradient of both constraints with respect to particle i, $\nabla_{\mathbf{p}_i} C_s$ and $\nabla_{\mathbf{p}_i} C_d$, is $\mathbf{n}$, which, according to Equation 7, means that this spring-damper is purely acting in normal direction. This allows us to easily add tangential contact effects such as friction and adhesion later on.

6.4. Solver Iteration Independence

Now that we have a fully configurable, single, spring-damper constraint, we need to address the issue of the number of solver iterations. As described in [MHHR06], the simple, normalized stiffness multiplier produces different results with different solver iteration counts, which is obvious. Müller et al. presented an effective way to produce independence from solver iterations by converting the stiffness multiplier into a *stiffness factor* via a conversion function which was based on the total number of solver iterations applied each frame. This works fine if a fixed number of iterations is used, which is usually the case in game-like interactive and real-time applications, as in our use case.

In the previous examples we have so far always assumed only a single iteration when performing an integration using the position-based method. Let's now examine the behavior of the simple, one-dimensional spring-damper system from Section 6.3 when integrated using the position-based approach with more than one iteration. In this case a single prediction step is followed by several constraint corrections, which leads to the following sequence:

$$\begin{aligned} p^{(0)} &= x_r + \Delta t v_r \\ p^{(1)} &= (1 - k_{pb} - c_{pb}) p^{(0)} + c_{pb} x_r \\ &\vdots \\ p^{(n)} &= (1 - k_{pb} - c_{pb}) p^{(n-1)} + c_{pb} x_r \\ x_{r+1} &= p^{(n)} \\ v_{r+1} &= \frac{x_{r+1} - x_r}{\Delta t} \end{aligned} \tag{24}$$

Here, n denotes the number of solver iterations and $p^{(i)}$ corresponds to the corrected position after the i'th iteration. As can be easily seen, with more than one iteration we no longer obtain the desired particle position which models a spring-damper. By examining the series of corrected positions $p^{(i)}$ more closely, we are able to bring the final corrected position $p^{(n)}$ into a more convenient form, writing it as the polynomial

$$\begin{aligned} f_n(\alpha) :=& p^{(0)}\alpha^n + c_{pb}x_r\alpha^{n-1} + \cdots + c_{pb}x_r\alpha^0 \\ =& p^{(0)}\alpha^n + c_{pb}x_r \sum_{k=0}^{n-1} \alpha^k = p^{(n)}, \end{aligned} \tag{25}$$

with $\alpha = (1 - k_{pb} - c_{pb})$. Under the assumption that $\alpha \neq 1 \Leftrightarrow k_{pb} + c_{pb} \neq 0$, which means that at least the spring or the damper must be active, we can replace the geometric series term by $\frac{1-\alpha^n}{1-\alpha}$ and obtain

$$\begin{aligned} f_n(k_{pb}, c_{pb}) :=& p^{(0)}(1 - k_{pb} - c_{pb})^n \\ &+ c_{pb}x_r \frac{1 - (1 - k_{pb} - c_{pb})^n}{1 - (1 - k_{pb} - c_{pb})} = f_n(\alpha). \end{aligned} \tag{26}$$

The function f_n defines the corrected position after n iterations. In order to obtain the desired spring-damper result as in Equation 20, the final corrected position $p^{(n)}$ must be equivalent to $(1 - k_{pb} - c_{pb})p + c_{pb}x_r$. To this end, we introduce solver-iteration independent, modified relaxation factors k'_{pb} and c'_{pb}, which replace k_{pb} and c_{pb} in the correction phase (see Eq. 24), and for which we must have

$$f_n(k'_{pb}, c'_{pb}) \overset{!}{=} p^{(0)}(1 - k_{pb} - c_{pb}) + c_{pb}x_r. \tag{27}$$

This condition can be solved to obtain the following relations

$$\begin{aligned} k'_{pb} &= 1 - c_{pb}\frac{1 - \alpha^{1/n}}{1 - \alpha} - \alpha^{1/n} \\ c'_{pb} &= c_{pb}\frac{1 - \alpha^{1/n}}{1 - \alpha} \\ \alpha &= 1 - k_{pb} - c_{pb} \end{aligned} \tag{28}$$

which allow us to convert the solver-iteration *dependent* stiffness and damping factors k_{pb} and c_{pb} from Equation 21 into solver-iteration *independent* stiffness and damping factors k'_{pb} and c'_{pb}. The factors k'_{pb} and c'_{pb} are used instead of k_{pb} and c_{pb} in Equation 23. With these modifications a single position-based spring-damper constraint simulated with multiple solver iterations is equivalent to a spring-damper integrated with implicit Euler, as demonstrated in Figure 2.

6.5. Friction and Adhesion

Soil shear strength, which resists collapse of soil piles, is governed by the soil's cohesion and internal friction. The internal friction is for the most part produced by the dry friction between individual soil particles, which form the particle skeleton of a soil formation. Interlocking between particles caused by roughness of the facing surfaces (particle angularity) also plays a role, further strengthening the soil [Das83]. In addition particle angularity creates the effect of rolling resistance. In our simulation model we omit the interlocking effect and model the internal friction and cohesion as pure dry friction and surface adhesion respectively. Our simulator is able to produce various different angles of repose with the proposed parameter set, allowing for the representation of materials ranging from purely frictious, cohesion-less materials such as sand and gravel to cohesive soils such as loam and clay. The following sections describe the proposed models in detail.

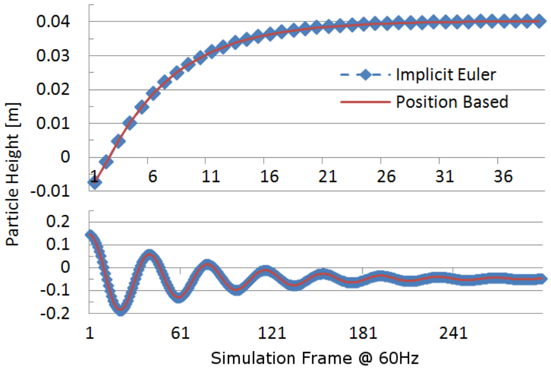

Figure 2: *Response of spring-damper integrated with our approach (5 solver iterations) and implicit Euler. A single particle with radius 0.05 m and 1 kg mass collides with a flat horizontal ground at height zero. Top: stiffness and damping coefficients are k=1000 $\frac{N}{m}$ and c=100 $\frac{Ns}{m}$. Bottom: k=100 $\frac{N}{m}$, c=0 $\frac{Ns}{m}$.*

6.5.1. Friction

For the simulation of friction and adhesion, which can be modelled as purely viscous effects, we can reuse the damper constraint from Section 6.2. Friction can be described by the Coulomb friction law, relating friction forces with normal forces acting at a contact point [TBV12] as

$$\begin{aligned} |\mathbf{v}_t| = 0 &\Rightarrow |\mathbf{f}_t| \leq \mu_s|\mathbf{f}_n| \\ |\mathbf{v}_t| \neq 0 &\Rightarrow |\mathbf{f}_t| = \mu_k|\mathbf{f}_n| \wedge \mathbf{f}_t \cdot \mathbf{v}_t = -|\mathbf{f}_t||\mathbf{v}_t| \end{aligned} \tag{29}$$

with relative tangent velocity vector $\mathbf{v}_t$, and friction and normal force vectors $\mathbf{f}_t$, $\mathbf{f}_n$ respectively. Intuitively this states that if the contact is sliding (relative tangent velocity is not zero), the friction force must always oppose the relative tangent velocity. Also, the friction force magnitude is linearly proportional to the normal force, scaled by one of two friction coefficients, the static and kinetic friction coefficients μ_s and μ_k respectively. As demonstrated in [SWLB14], the Projected Gauss-Seidel method for Rigid Body LCPs is an efficient way to numerically model this relationship. In this approach, for each contact constraint, first the normal force is computed. The resultant normal force is then used to cap the friction force via projection onto the friction cone. Our friction approach is inspired by this technique.

We use the following damper constraint (see Eq. 17) without relaxation ($c'_{pb} = 1$) to produce the tangential friction force between particles i and j:

$$\begin{aligned} C_f(\mathbf{p}_i, \mathbf{p}_j) &= \mathbf{v} \cdot \mathbf{t} \overset{!}{=} 0 \\ \mathbf{v} &:= (\mathbf{p}_i - \mathbf{x}_i) - (\mathbf{p}_j - \mathbf{x}_j) \end{aligned} \tag{30}$$

Here, $\mathbf{v}$ measures the relative velocity between the particles and $\mathbf{t} = \frac{\mathbf{v}_\perp}{|\mathbf{v}_\perp|}$ denotes its normalized projection onto the contact tangent plane with $\mathbf{v}_\perp = \mathbf{v} - (\mathbf{v} \cdot \mathbf{n})\mathbf{n}$. If the relative velocity has no tangent component (i.e., $|\mathbf{v}_\perp| = 0$) no friction correction needs to be applied. Otherwise, the position correction applied by this constraint to particle i can be written according to Equation 7 as

$$\Delta\mathbf{p}_{i,f} = \frac{w_i}{w_i + w_j} \lambda \mathbf{t}, \tag{31}$$

with $\lambda = C_f(\mathbf{p}_i, \mathbf{p}_j)$ and $\nabla_{\mathbf{p}_i} C_f = \mathbf{t}$. In order to model Coulomb friction we limit the constraint correction by projecting the displacement λ onto the friction cone as in the Projected Gauss-Seidel method, which yields the capped displacement λ_f. The cap is performed based on the normal correction λ_n, caused by the spring-damper constraint in Equation 23, which can be reformulated and combined with the friction correction, yielding

$$\Delta\mathbf{p}_{i \leftarrow j} = \frac{w_i}{w_i + w_j} (\lambda_f \mathbf{t} + \lambda_n \mathbf{n}), \tag{32}$$

where λ_n depends on the spring-damper factors k'_{pb} and c'_{pb}, and λ_f is defined as

$$\lambda_f = \begin{cases} \lambda & \text{, if } |\lambda| \leq \mu_s |\lambda_n| \\ \text{sgn}(\lambda)\mu_k |\lambda_n| & \text{, otherwise} \end{cases} \tag{33}$$

We derived this projected correction scheme as follows. By examining the difference in relative particle velocities induced by a position correction, we derived formulas which convert a scalar (or vectorial) position correction λ, applied to a particle with mass m, into an equivalent constraint force f and vice versa:

$$f = \frac{m}{\Delta t^2} \lambda, \quad \lambda = \Delta t^2 \frac{f}{m} \tag{34}$$

These relations can be used to extract the force f_n from λ_n, compute the tangential friction force $f_t = \mu_s |f_n|$ and transform it back into an equivalent tangent displacement λ_f. Putting everything together this yields Equation 33. Figure 3 demonstrates the presented friction model.

6.5.2. Adhesion

Cohesive forces in soil increase with the surface area in which they act when a soil is subject to shear stress. Motivated by this fact, we take the contact surface A_c into account when computing the adhesive force F_a acting on particles from the provided adhesion parameter a:

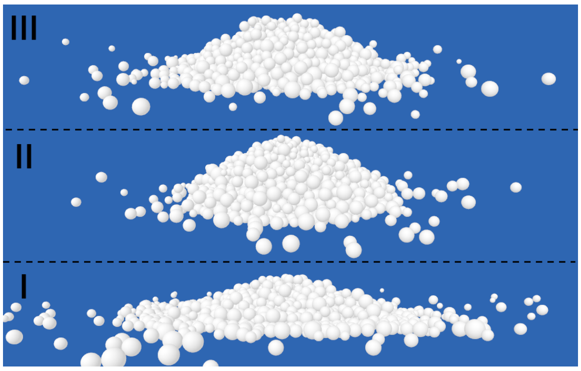

Figure 3: *Angle of repose experiment: 2000 particles are poured to form a pile and stabilize. Experiments performed with PP^2 solver with 10 iterations, $\varepsilon = 0.5$, 0.0003 m sleep threshold, and varying kinetic friction angle θ_k and adhesion a. I: $\theta_k = 5°$, $a = 0$ Pa. II: $\theta_k = 30°$, $a = 0$ Pa. III: $\theta_k = 5°$, $a = 1000$ Pa.*

$$F_a = a A_c \tag{35}$$

Favouring fast computation over accuracy, we use a quadratically interpolated contact surface. Assuming that the contact surface is maximized when the penetration ζ between two particles i and j is equal to the smaller of the two radii, $r_{min} = min\{r_i, r_j\}$, the particle contact surface can be approximated as

$$A_c = \pi r_{min}^2 (1 - (\frac{\zeta}{r_{min}} - 1)^2). \tag{36}$$

The computed adhesive force is applied both on the contact tangent plane as well as in contact normal direction. For the tangential component, we compute the scalar displacement λ_a equivalent to F_a using Equation 34 and add it to the friction displacement cap in Equation 33, thus combining friction and tangential adhesion in a single constraint. For the normal adhesion we use an additional damper constraint without any relaxation ($c'_{pb} = 1$) and cap the normal displacement by λ_a. The corresponding adhesive position correction is applied immediately after the spring-damper position correction in Gauss-Seidel style, which avoids having to process the additional damper constraint in the solver.

6.6. Rigid Body Coupling

A fundamental component of any interactive particle simulation is interaction with the environment. In particular, for training of excavator and bulldozer operators, forces applied to the cutting tool (the bucket or blade) are very important to correctly model the effects of the machine interacting with the simulated granular material. At the beginning of each frame we compute rigid body/particle contacts using the *predicted* particle positions. For each particle position correction computed for rigid body/particle contacts in the solver, we compute an equivalent force vector using Equation 34 and apply it to the rigid body as an external force at the contact position. Figure 4 shows a simulation with rigid body/particle interactions.

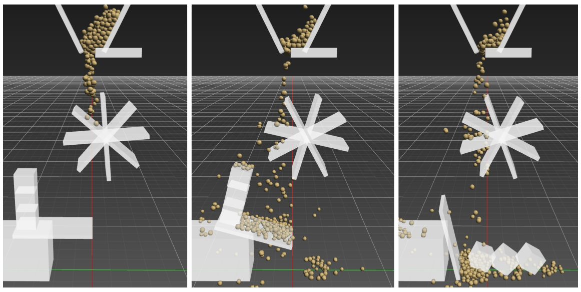

Figure 4: *Rigid body coupling: The wheel turns as a result of rigid body/particle contacts. Particles accumulate on the plank and make it tilt, causing weights to slide down.*

7. Error Reduction

The proposed relaxed Jacobi-style solver allows for an easy parallel implementation and high performance. On the downside, it requires a high number of solver iterations to converge to an equally stiff result as in the sequential Gauss-Seidel scheme (see Fig. 5). To improve convergence, we propose an intuitively motivated approach. Noticing that in a granular material simulation the error in the solution tends to increase with increasing depth in a pile, especially with few iterations, we suggest the following simple treatment to reduce the error. For every contact between particles i and j, we correct the position of the particle which lies on top by a larger quantity than the other, and weighting the effect by the angle between contact normal and gravity. This has the effect of artificially increasing the mass of the lower particle. The weighting factor e_i, applied to the particle position correction $\Delta\mathbf{p}_{i\leftarrow j}$, is defined as

$$e_i = 1 - \varepsilon(\mathbf{n} \cdot \hat{\mathbf{g}}), \tag{37}$$

where $\hat{\mathbf{g}}$ denotes the normalized gravity vector and $\varepsilon \in [0,1]$ denotes a global error reduction parameter which allows tuning in the effect. The weighting factor e_i can be seen as a way to model the increase in effective mass further down in a material pile. This very simple and efficient approach yields good results as can be seen in Figure 5.

The proposed approach is loosely related to the shock propagation technique proposed in [Erl05] and used to resolve inter-penetrations in stacks and piles. However, as opposed to the shock propagation technique, our approach is a local heuristic. A global heuristic could be applied here as well, which could model the effect of the increase in effective mass more accurately by looking at the graph distance of particles to some ground body in the contact network, as in [Erl05], and computing the weight as a function of the latter. The advantage of the local heuristic compared to the global heuristic is that the latter requires a contact graph analysis as well as the concept of *ground*, which is not the case for the local heuristic. It is thus more efficient and flexible. Also it provides satisfying results in our application which are visually feasible (see Fig. 5, bottom).

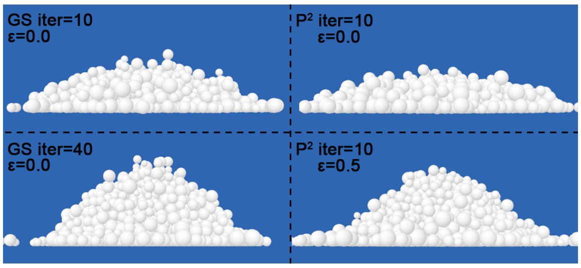

Figure 5: *Slope angles produced by Gauss-Seidel (GS) solver without error reduction and our P^2 solver. Top: As expected, GS performs slightly better with same iteration count (iter). Bottom: P^2 with error reduction ($\varepsilon = 0.5$) and iter $= 10$ produces a slope angle close to GS with iter $= 40$.*

As an additional error reduction technique we introduce particle sleeping. Once a particle almost comes to rest, that is, the corrected particle position $\mathbf{p}_i$ at the end of the step is very close to the previous particle position $\mathbf{x}_i$, we do not modify the particle position. This technique allows for the simulation of very stable heaps, since it eliminates drift even at a small number of solver iterations. However, it can not be used when particles are transported in, e.g., a truck bed or bucket, which is the case in our target application. Therefore, we do not make use of this technique in our training simulator. However, we are still able to simulate stable heaps, even at low solver iterations, because we integrate our Parallel Particles approach with a hybrid grid/particle soil simulation framework [HBK09]. In this method particles are removed and replaced by grid volume if they come to rest, therefore leading to stable, drift-free slope angles (see Fig. 6).

8. Results

In this section we present simulation results of the proposed granular material simulation approach Parallel Particles (P^2). We integrated P^2 with a Virtual Reality Earthmoving training simulator based on CM Labs' multi-body dynamics toolkit Vortex 6.2 [CM 14]. In the simulator we employ the hybrid particle/grid soil simulation approach from [HBK09] which adaptively introduces particles into the simulation, replacing portions of the grid as required by the digging operations. Particles are removed from the simulation and transformed back into grid volume once they settle.

All presented simulations ran at interactive to real-time simulation rates and were obtained on an Intel Core i7-3770 CPU @ 3.40GHz with 4 physical cores. Figure 6 shows various interactions of an excavator with a loam type soil.

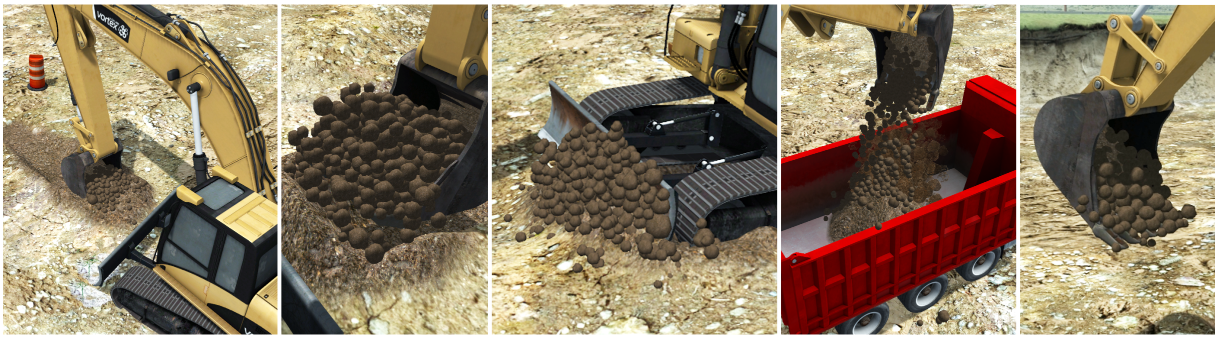

Figure 6: *Excavator interacting with a frictious and cohesive loam type soil. Rightmost picture shows adhesive particles stuck to the bucket, which need to be actively released by performing accelerating bucket motions.*

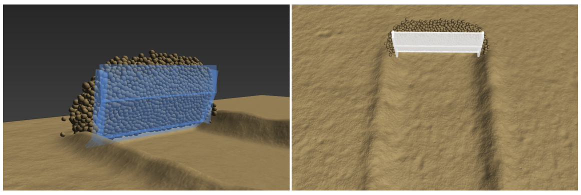

Figure 7: *Kinematic bulldozer blade pushing soil over long distance, which yields physically plausible soil ridges.*

It also depicts adhesive loam particles stuck to the bucket, which need to be actively released by performing accelerating bucket motions. We performed a bulldozing experiment showing the formation of ridges due to particles flowing sidewards out of the blade. The results can be seen in Figure 7. Figure 1 shows performance results of P^2 obtained during the soil pile experiments (see Fig. 3). As can be seen, the speed-up when using multiple threads is significant.

9. Conclusion and Future work

We presented a novel approach for the simulation of granular material which yields fast and stable results. Important soil dynamics effects, such as piling and sticking, are captured to a physically plausible extend. The proposed contact dynamics model supports kinetic and static Coulomb friction and adhesion. Through the use of a simple regularization method we were able to formulate a spring-damper constraint on the position level which we use to model visco-elastic contacts. We proposed a novel, parallel solver for Position Based Dynamics enabling the simulation of granular material at interactive to real-time frame rates. As such, the method proved practical for use in an Earthmoving training simulator.

In the future, we would like to explore a GPU implementation of the approach. This would allow us to use more particles and gain in realism both in terms of physical accuracy and visual fidelity.

The presented method proved sufficient for our application. However, with our current solver, if more than one contact is applied to a particle, the normal forces do not converge to the actual spring-damper forces, even with many iterations. This is likely caused by the too aggressive relaxation through constraint averaging. Convergence can probably be improved by performing actual mass-splitting as in Tonge et al. [TBV12], weighting the position corrections accordingly, which would produce a weighted instead of an ordinary average.

References

[BMOT13] BENDER J., MÜLLER M., OTADUY M. A., TESCHNER M.: Position-based methods for the simulation of solid objects in computer graphics. In *EG* (2013). 2

[BYM05] BELL N., YU Y., MUCHA P. J.: Particle-based simulation of granular materials. In *SCA* (2005), pp. 77–86. 2

[Cat11] CATTO E.: Soft constraints. In *GDC* (2011). 4

[CM 14] CM LABS SIMULATIONS: Vortex Dynamics 6.2. `http://www.cm-labs.com`, 2014. 9

[Das83] DAS B. M.: *Advanced soil mechanics*. Hemisphere Publishing Corporation, 1983. 7

[Erl05] ERLEBEN K.: *Stable, Robust, and Versatile Multibody Dynamics Animation*. PhD thesis, 2005. 9

[HBK09] HOLZ D., BEER T., KUHLEN T.: Soil Deformation Models for Real-Time Simulation: A Hybrid Approach. In *VRIPHYS* (2009), Eurographics Association, pp. 21–30. 9

[IWT12] IHMSEN M., WAHL A., TESCHNER M.: High-resolution simulation of granular material with sph. In *VRIPHYS* (2012), Eurographics Association, pp. 53–60. 2

[LL03] LIU G. R., LIU M. B.: *Smoothed Particle Hydrodynamics - A Meshfree Particle Method*. World Scientific, 2003. 2

[MHHR06] MÜLLER M., HEIDELBERGER B., HENNIX M., RATCLIFF J.: Position based dynamics. In *VRIPHYS* (2006), Eurographics Association, pp. 71–80. 2, 3, 4, 5, 6

[MMCK14] MACKLIN M., MÜLLER M., CHENTANEZ N., KIM T.-Y.: Unified particle physics for real-time applications. In *SIGGRAPH* (2014). To appear. 2

[SWLB14] SERVIN M., WANG D., LACOURSIÈRE C., BODIN K.: Examining the smooth and nonsmooth discrete element approaches to granular matter. *International Journal for Numerical Methods in Engineering 97*, 12 (2014), 878–902. 2, 7

[TBV12] TONGE R., BENEVOLENSKI F., VOROSHILOV A.: Mass splitting for jitter-free parallel rigid body simulation. *ACM Trans. Graph. 31*, 4 (July 2012), 105:1–105:8. 2, 4, 7, 10