Interactive High-Resolution Simulation of Granular Material

Alexander Sommer[1]
alexander.sommer@hs-rm.de

Ulrich Schwanecke[1]
ulrich.schwanecke@hs-rm.de

Elmar Schoemer[2]
schoemer@uni-mainz.de

[1] Computer Vision and Mixed Reality Group, RheinMain University of Applied Sciences
Wiesbaden Rüsselsheim, Germany
[2] Institute of Computer Science, Johannes Gutenberg University Mainz, Germany

ABSTRACT

We introduce a particle-based simulation method for granular material in interactive frame rates. We divide the simulation into two decoupled steps. In the first step, a relatively small number of particles is accurately simulated with a constraint-based method. Here, all collisions and the resulting friction between the particles are taken into account. In the second step, the small number of particles is significantly increased by an efficient sampling algorithm without creating additional artifacts. The method is particularly robust and allows relatively large time steps, which makes it well suited for real-time applications. With our method, up to 500k particles can be computed in interactive frame rates on consumer CPUs without relying on GPU support for massive parallel computing. This makes it well suited for applications where a lot of GPU power is already needed for render tasks.

Keywords: position-based dynamics, position-based simulation, real-time simulation, animation

1 INTRODUCTION

Granular materials are composed of many, small bodies that can be clearly separated from each other. The individual components of the granular material can be very different, for example grains, sand, gravel, rubble, but also beans, rice and much more.

The simulation of such materials is particularly challenging, since the macroscopic behavior of the material is determined by the microscopic interactions between the individual grains. A complete, physically correct description of all interactions is virtually impossible, so simplified models are used that accurately reflect reality to some degree. For this purpose, various methods have been developed in the field of civil engineering and later also in the field of computer graphics. While methods from the engineering field must be able to simulate acting forces as accurately as possible, the main focus of computer graphics methods is on generating visually plausible results.

A common problem with current simulation methods in computer graphics is that, despite considerable simplification, it is not possible to achieve interactive frame rates at higher particle counts that are necessary for plausible visualization. In this work we use a position-based simulation [4] with a low particle count that can be easily computed within interactive frame rates and then refine it with an efficient upsampling algorithm to obtain a large particle count. The two steps of the simulation are decoupled from each other and can be computed at different temporal resolutions. The accurate behavior of the continuum is determined by calculating the individual collisions in the first step of the simulation. The behavior of the high-resolution particles in the second step is ensured by interpolating the underlying velocity field and by partially blending in external forces.

2 RELATED WORK

In the field of computer science, the physical simulation of granular material plays an important role, besides physical computing, especially in computer graphics. A distinction is made between continuum methods and purely particle-based discrete methods.

Continuum methods are particularly well suited to simulate granular flow in the most time-effective manner, since there is a decoupling between grain size and resolution of the simulation. This decoupling is at the expense of finer details in areas where the motion is in free flow, for example at surfaces, free-falling grains or in the formation of spatter. Zhu and Bridson [29] used a hybrid Euler-Lagrangian formulation of the Fluid-Implicit (FLIP) method, treating sand as a fluid. Narain et. al. [22] developed a continuum-based model that efficiently calculates internal pressures and frictional stresses with an unilateral incomprehensibility constraint. Their method also allows two-way coupling with rigid-bodies. Lenaerts and Dutré [17] make

use of a pure Lagrangian approach by simulating granular material with the Smoothed Particle Hydrodynamics (SPH) method [19]. Alduán and Otaduy [1] incorporated Narain et. al.'s unilateral incomprehensibility within the predictive-corrective incompressible SPH (PCISPH) method [24]. Klar et. al. use the Drucker-Pager plastic flow model to simulate sand in the Material Point Method (MPM) [26][16]. Hu et. al. introduce the Moving Least Squares Material Point Method (MLS-MPM) [13] to enable the simulation of new phenomena in MPM like two-way coupling with rigid-bodies.

Discrete methods, in contrast to continuum methods, simulate the macroscopic behavior of the material based on contacts and collisions between individual grains or particles. This enables realistic modeling of various physical phenomena. However, accurate simulation of granular media often requires a very small grain size or particle radius, and thus a large number of particles. Collision detection of a large number of particles is computationally intensive. A small particle radius usually also requires an increase of the time steps in the simulation. In practice, therefore, often unnaturally large grain sizes have to be used. The first approaches for simulating granular material with discrete methods by Cundall and Strack [8] stem from Discrete Element Method (DEM) theory in molecular dynamics. Later Bell et. al. [3] model granular material as non-spherical particles following DEM principles with contact and shear forces for animation purposes. Müller et. al. [20] developed position-based dynamics (PBD), a particle-based simulation framework that applies positional changes of particles directly to the position layer without calculating forces between individual particles. Macklin et. al. [18] developed a static and dynamic friction model for PBD to mimic granular material behavior in this unified framework specifically tailored for real-time applications. Frâncu and Moldoveanu [10] formulated an accurate contact and Coulomb friction model suitable for rigid and flexible bodies in PBD.

Recent advances in the field of machine learning have also produced new AI-based approaches to predicting the behavior of granular materials. The neural networks used in these approaches are trained with classically simulated data. Again, there are approaches that use continuum methods like Coombs and Augarde [7] who use MPM and Sanchez-Gonzalez et. al [23] who use SPH and MPM and approaches that use DEM like Wallin and Servin [27]. Furthermore, hybrid techniques exist that combine the strengths of continuum and discrete methods, like the recently proposed method by Yue et. al. [28].

In this work we focus on discrete methods. We try to overcome the weaknesses of discrete methods, namely the size of the individual particles, by splitting our simulation into two steps. First, an accurate PBD simulation that takes into account collisions between individual particles but is computed at a low resolution with large particle radii. Second, we use an efficient refinement algorithm that replaces the results of the actual simulation with a much higher number of particles with a smaller radius. The decoupling of the two simulation parts makes it possible to calculate both parts with different time steps. This way, our method is very efficient, since the upsampling in the second part can be done with much larger time steps than the contact calculation in the low resolution part. The idea of this dual partitioning is not new. It has been already applied by Alduán et. al. [2], who compute their low resolution (LR) guide particles using the continuum method of Bell et. al. [3]. They move their high resolution (HR) visualization particles using the flow of LR particles as well as external forces. Ihmsen et. al. [15] took up this idea. They calculated their LR particles using another continuum based method, namely the friction model in SPH developed by Alduán and Otaduy [1]. Furthermore, they optimized the algorithm to interpolate the motion of the HR particles by superimposing external forces on the velocity field of the LR particles depending on the density of the LR particles. In contrast to the previously mentioned work, we use a discrete method with PBD. This allows real-time simulation of granular media thanks to the speed advantages of PBD over SPH. In addition, we modify the algorithm for advection of HR particles to achieve better interaction with domain boundaries and prevent particles from sticking to rigid bodies.

3 LR SIMULATION

In this section, we first describe the simulation of LR guiding particles. As mentioned before, our LR simulation is based on PBD [20], respectively the modification described by Macklin et. al. [18] for parallel execution by constraint averaging. This allows interactive frame rates as needed in real-time applications even for larger numbers of particles.

PBD and other discrete simulation methods use particles to represent each individual discrete element of the material being simulated. All particles have the same radius r_{LR}. In general these particles can have arbitrary attributes. For our purposes it is sufficient to assign each particle a position $\boldsymbol{x}$, a velocity $\boldsymbol{v}$ and a mass m.

In other discrete simulation methods, the particle motion is determined by time integration of all occurring internal and external forces. In PBD, however, position changes $\Delta\boldsymbol{x}$ due to internal forces are expressed directly by so-called constraints. The resulting velocity of individual particles is then calculated from the position difference between two time steps, i.e.

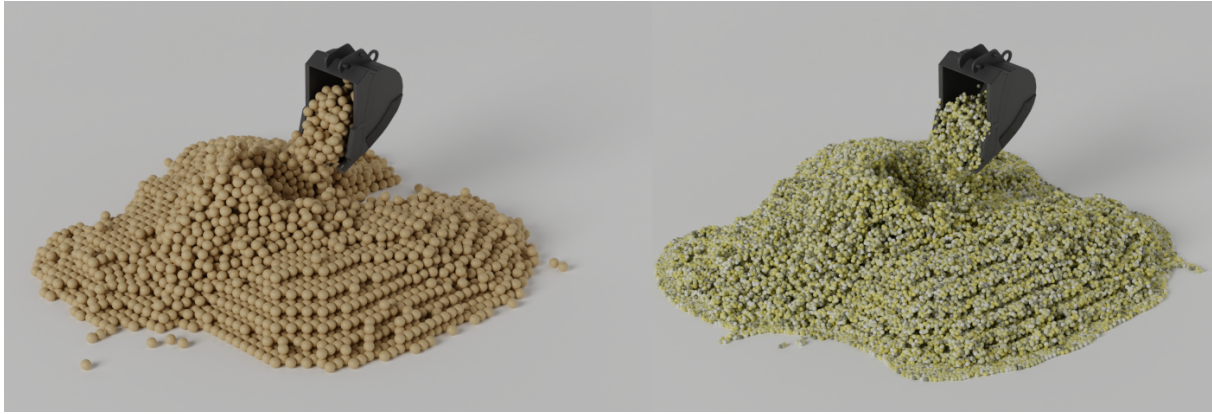$$\Delta\boldsymbol{v} = \frac{\Delta\boldsymbol{x}}{\Delta t_{LR}}.$$

Figure 1: An excavator lifts up sand. left: LR Simulation with 6.8k Particles. right: HR Simulation with 147k.

Each of the constraints used in PBD can affect k particles. In general a constraint can be an equality constraint $C(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k) = 0$ or an inequality constraint $C(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k) \geq 0$.

For our granular LR simulation we use the friction model proposed by Macklin et. al. [18]. Their model consists of two different contact constraints between a collision pair $\boldsymbol{x}_i$, $\boldsymbol{x}_j$. First, interpenetrations between the two collision partners are solved by displacing the two particles along the collision vector $\boldsymbol{x}_{ij} = \boldsymbol{x}_j - \boldsymbol{x}_i$. The displacement is proportional to the mass ratio:

$$\Delta \boldsymbol{x}_i = -\hat{m}_{ij} \left(2r_{LR} - |\boldsymbol{x}_{ij}| \right) \cdot \frac{\boldsymbol{x}_{ij}}{|\boldsymbol{x}_{ij}|} \quad (1)$$

$$\Delta \boldsymbol{x}_j = \hat{m}_{ji} \left(2r_{LR} - |\boldsymbol{x}_{ij}| \right) \cdot \frac{\boldsymbol{x}_{ij}}{|\boldsymbol{x}_{ij}|} \quad (2)$$

with

$$\hat{m}_{ij} = \frac{m_i^{-1}}{m_i^{-1} + m_j^{-1}} = \frac{m_j}{m_i + m_j}$$

After resolving collisions the particles i, j have new temporary positions $\tilde{\boldsymbol{x}}_i = \boldsymbol{x}_i + \Delta \boldsymbol{x}_i$, $\tilde{\boldsymbol{x}}_j = \boldsymbol{x}_j + \Delta \boldsymbol{x}_j$. The relative displacement between original positions and these new positions

$$\Delta \boldsymbol{x}_{ij} = (\tilde{\boldsymbol{x}}_i - \boldsymbol{x}_i) - (\tilde{\boldsymbol{x}}_j - \boldsymbol{x}_j) = \Delta \boldsymbol{x}_i - \Delta \boldsymbol{x}_j$$

is used to calculate a frictional position delta. It is based on the tangential component

$$\Delta \boldsymbol{x}_{ij_\perp} = \Delta \boldsymbol{x}_{ij} - (\Delta \boldsymbol{x}_{ij} \cdot \boldsymbol{x}_{ij}) \frac{\boldsymbol{x}_{ij}}{|\boldsymbol{x}_{ij}|^2}$$

relative to the collision vector $\boldsymbol{x}_{ij}$. With this tangential component the positional delta for particle i, j is calculated as

$$\Delta \boldsymbol{x}_i = -\hat{m}_{ij} \begin{cases} \Delta \boldsymbol{x}_{ij_\perp} & \text{if } |\Delta \boldsymbol{x}_{ij_\perp}| < 2r_{LR}\mu_s \\ \Delta \boldsymbol{x}_{ij_\perp} \min_{\text{fric}} & \text{else} \end{cases} \quad (3)$$

$$\Delta \boldsymbol{x}_j = \hat{m}_{ji} \begin{cases} \Delta \boldsymbol{x}_{ij_\perp} & \text{if } |\Delta \boldsymbol{x}_{ij_\perp}| < 2r_{LR}\mu_s \\ \Delta \boldsymbol{x}_{ij_\perp} \min_{\text{fric}} & \text{else} \end{cases} \quad (4)$$

with

$$\min_{\text{fric}} = \min \left(\frac{2r_{LR} \cdot \mu_k}{|\Delta \boldsymbol{x}_{ij_\perp}|}, 1 \right).$$

The parameters μ_s and μ_k are the dry friction coefficients for static and kinetic friction. For most cases, we use $\mu_s = 0.35$ and $\mu_k = 0.3$ in our simulation. For a collision pair with differing friction coefficients the cross friction coefficients

$$\mu_{ij} = \sqrt{\mu_i \mu_j}$$

has to be calculated, where μ_i and μ_j are the static respectively kinetic friction coefficients for particle i and j. The static and kinetic cross friction coefficients are used in this case to calculate the positional deltas $\Delta \boldsymbol{x}_i$ and $\Delta \boldsymbol{x}_j$.

While Eq. (1) and (2) resolve only the interpenetration of the particles, Eq. (3) and (4) model the friction effects. In the first case of Eq. (3) respectively (4), static friction effects are modeled by preventing any tangential motion if the particle velocity is below a traction threshold. Kinetic friction effects are treated in the second case by limiting the positional delta depending on the penetration depth.

The friction model described previously is used to calculate the friction effects between two particles. To model the interaction between particles and rigid-bodies or domain boundaries we sample them with particles as well. This allows us to use a unified collision handling and to work with the same constraints for all types of contacts. In the [25], a method is described to sample arbitrary closed volumes of 3D triangle meshes with particles for the needs of particle-based simulation. For stationary objects, the particles are assigned an infinitely large mass or an inverted mass of $m^{-1} = 0$, respectively. In this case it is sufficient to only sample the surface of the object. A method to sample arbitrary surfaces of 3D triangle meshes based on the uniform sampling algorithm of Bowers et. al. [5] can also be found in [25]. For moveable rigid-bodies the individual particles are first treated

as if they were unconnected. Then, a shape-matching constraint [21] is applied to the particles of the rigid body to find the particle configuration corresponding to a transformed resting state. The position delta from this constraint is given by

$$\Delta \boldsymbol{x}_i = (\boldsymbol{R}\boldsymbol{x}_{o_i} + \boldsymbol{c}) - \tilde{\boldsymbol{x}}_i$$

where $\boldsymbol{c}$ corresponds to the center of gravity of the new deformed particles of the rigid body. $\boldsymbol{x}_{o_i}$ corresponds to the offset of the i-th particle from the center of gravity of the undeformed particle positions. $\boldsymbol{R}$ is a rotation matrix which is given by the polar decomposition [12] of the covariance matrix

$$\boldsymbol{C} = \sum_i^n (\tilde{\boldsymbol{x}}_i - \boldsymbol{c}) \cdot \boldsymbol{x}_{o_i}^T$$

of the deformed shape. This enables a two-way coupling between the granular medium and rigid-bodies.

All of these previously mentioned constraints, namely contact, friction and shape-matching constraints, can be calculated in parallel. A particle i can receive a position delta $\Delta \boldsymbol{x}_{i_k}$ from several constraints k. Therefore, after all constraints have been calculated, the sum of all positions delta belonging to a particle i is divided by n_i the number of constraints involved:

$$\Delta \boldsymbol{x}_i = \frac{1}{n_i} \sum_k^{n_i} \Delta \boldsymbol{x}_{i_k}.$$

The procedure of constraint solving is repeated a couple of times until a solution is found that satisfies all the constraints. For our calculations between 3 and 5 iterations were sufficient. However, it can happen that no convergence was achieved and particles are in an invalid position at the start of a simulation step. This can lead to particles experiencing an unwanted acceleration in the next time step due to the next constrain to solve. This effect is especially strong the smaller the time step is. To avoid this, 1-2 stabilization iterations of the pure contact constraints, Eq. (1) and (2), are executed per time step. The resulting position deltas are applied to the temporary positions as well as to the regular positions. This prevents unnatural kinetic energy from being added to the system due to these irregularities when calculating new velocities. The complete algorithm for performing one time-step is shown in Algorithm 1.

To achieve a more stable piling of particles, e.g. in sand piles, and to prevent dissolution, a further improvement is made. As described in [18] a scaled mass

$$m_i^* = m_i \cdot e^{-h(\boldsymbol{x}_i)}$$

is assigned to each particle i based on the relative height $h(\boldsymbol{x}_i)$ with respect to the ground plane. These scaled masses m^* are used to solve the contact and friction constraints. Since higher particles exert less pressure on the levels below, the system converges faster and the piles are more stable.

Algorithm 1 One simulation time-step
|	
1:	**for all** particles i **do**
2:	$\quad \boldsymbol{v}_i \leftarrow \boldsymbol{v}_i + \Delta t_{LR}\boldsymbol{g}$ $\quad\quad\quad\quad \triangleright \lvert\boldsymbol{g}\rvert = 9.81\frac{m}{s^2}$
3:	$\quad \tilde{\boldsymbol{x}}_i \leftarrow \boldsymbol{x}_i + \Delta t_{LR}\boldsymbol{v}_i$
4:	$\quad m_i^* \leftarrow m_i e^{-h(\boldsymbol{x}_i)}$
5:	find_neighboring_particles()
6:	generate_stabilization_constraints()
7:	**for** stabilization iterations $it = 1, 2$ **do**
8:	$\quad \Delta\boldsymbol{x} \leftarrow 0, n \leftarrow 0 \quad\quad\quad\quad \triangleright \forall$ particle i
9:	$\quad \Delta\boldsymbol{x}, n \leftarrow$ solve_stabilization_constraints()
10:	$\quad$ **for all** particles i **do**
11:	$\quad\quad \tilde{\boldsymbol{x}}_i \leftarrow \tilde{\boldsymbol{x}}_i + \Delta\boldsymbol{x}_i / n_i$
12:	$\quad\quad \boldsymbol{x}_i \leftarrow \boldsymbol{x}_i + \Delta\boldsymbol{x}_i / n_i$
13:	generate_constraints()
14:	**for** solving iterations $it = 1, \ldots, 5$ **do**
15:	$\quad \Delta\boldsymbol{x} \leftarrow 0, n \leftarrow 0 \quad\quad\quad\quad \triangleright \forall$ particle i
16:	$\quad \Delta\boldsymbol{x}, n \leftarrow$ solve_constraints()
17:	$\quad \tilde{\boldsymbol{x}}_i \leftarrow \tilde{\boldsymbol{x}}_i + \Delta\boldsymbol{x}_i / n_i$
18:	**for all** particles i **do**
19:	$\quad \boldsymbol{v}_i \leftarrow (\tilde{\boldsymbol{x}}_i - \boldsymbol{x}_i) / \Delta t_{LR}$
20:	$\quad \boldsymbol{x}_i \leftarrow \tilde{\boldsymbol{x}}_i$
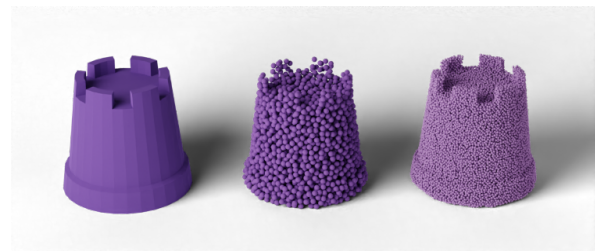
Figure 2: Exemplary illustration of the sampling process. left: Inital sandcastle mesh. middle: LR sampling. right: HR sampling

4 HR UPSAMPLING

In the previous section we described the simulation of LR guide particles taking into account all collisions between particles. We now describe how a finer resolution simulation result can be generated on the basis of the LR particles with the help of an upsampling. Figure 1 shows a comparison between the LR simulation on the left and the result of the HR upsampling on the right. This is done without having to perform complex collision queries between HR particles. Thus this upsampling is extremely effective and the size of the LR time step is decoupled from the size of the HR particles. Each LR Particle can be seen as a representation of several HR Particles.

It is especially important to create a good initial sampling to avoid artifacts. For example, a uniform sampling can lead to repetitive patterns during the simulation. Furthermore, if the HR particles are placed symmetrically around the LR particles, aliasing or staircase patterns can occur already in the initial state. Therefore,

we use the randomized volume sampling algorithm described in [25] for both the LR and the HR particles. We use a triangle mesh as a hull in which the particles should be located. First the bounding box delimiting the mesh is divided into grid cells with a side length of $\frac{2r}{\sqrt{3}}$, with r being the radius of the LR particles r_{LR} respectively HR particles r_{HR}. Then, a large number of randomly but uniformly distributed candidate positions are generated within the mesh and assigned to the respective grid cells. After that, for each grid cell within the mesh, an attempt is made to select a candidate position that does not overlap with already sampled positions. In this fashion the whole volume of the mesh is sampled. For a more detailed description we refer the reader to [25]. This algorithm is executed for both LR particles with radius r_{LR} and HR particles with radius r_{HR} before starting the simulation. Figure 2 shows an example of the sampling process. The initial mesh is displayed on the left, while in the middle the sampling for the LR simulation and on the right the corresponding sampling for the HR upsampling is shown.

Once the simulation is started, the LR particles are set in motion by the LR simulation described in the previous section. The HR particles trace the movement of these LR guide particles. HR particles should follow the flow of the LR simulation but still move individually to avoid the formation of clumps. For this we use the advection method described by Ihmsen et. al. [15]. For this, gravity is smoothly faded in as an external force depending on how densely a HR particle is surrounded by LR particles. If a HR particle is in the vicinity of many LR particles, the velocity field generated by them dominates. For this, distance-based weights w between HR particles i and LR particles j are calculated as

$$w_{ij} = \max\left(0, \left(1 - \frac{|\boldsymbol{x}_{ij}|^2}{9 \cdot r_{LR}^2}\right)^3\right).$$

These weights are used to determine the average velocity $\boldsymbol{v}_i$ at a HR particle i with

$$\tilde{\boldsymbol{v}}_i = \frac{1}{\sum_j w_{ij}} \sum_j w_{ij} \boldsymbol{v}_j.$$

It shall be noted that LR particles j can be granular material as well as the particle representation of rigid-bodies or boundaries. The blending in of external forces is controlled by a parameter

$$\alpha_i = \begin{cases} 1 - \max w_{ij} & \text{if } \max w_{ij} \leq c_1 \text{ or } \frac{\max w_{ij}}{\sum_j w_{ij}} \geq c_2 \\ 0 & \text{else} \end{cases}$$

which differs from zero only in sparse regions. The constant $c_1 = \frac{512}{729}$ is the distance-based weight for a distance equal to the LR particle radius r_{LR} and $c_2 = 0.6$ is
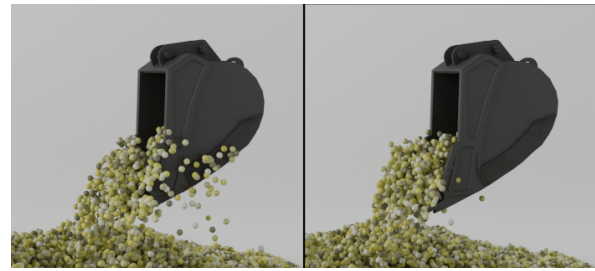

Figure 3: left: Ihmsen et. al.'s original upsampling with sticking artifacts. right: HR Simulation with our modification.

an empirically tested value. With this blending parameter the resulting velocity for a HR particle i is calculated as

$$\boldsymbol{v}_i^{t+1} = (1 - \alpha_i)\tilde{\boldsymbol{v}}_i^{t+1} + \alpha_i\left(\boldsymbol{v}_i^t + \Delta t_{HR}\boldsymbol{g}\right)$$

where $\boldsymbol{g}$ is the acceleration due to gravity and Δt_{HR} denotes the time-step size from time t to time $t+1$. The new position $\boldsymbol{x}_i$ for the i-th HR particle is then trivially obtained by time integration through

$$\boldsymbol{x}_i^{t+1} = \boldsymbol{x}_i^t + \Delta t_{HR}\boldsymbol{v}_i^{t+1}.$$

In contrast to Ihmsen et. al. we ignore the LR particles of rigid bodies if there are no LR granular particles in the vicinity. This prevents HR particles from sticking to rigid bodies that are moved externally (see Figure 3).

5 RESULTS

In contrast to previous works, our work focuses on runtimes in the range of interactive frame rates. Therefore, in this section we present some examples of our approach to illustrate the different possible applications.

5.1 Implementation

We have implemented our simulation framework in C++ 14. For more complex mathematical operations and mathematical data structures like vectors and matrices we use the Eigen3 [11] library. We parallelized our algorithms on the CPU using OpenMP [9]. For an efficient neighborhood search we use the CompactNSearch library based on the Compact Hashing approach by Ihmsen et al. [14]. Furthermore, we use the LEAVEN [25] library for sampling volumes and surfaces. All renderings in this work were created with Cycles in Blender [6].

5.2 Experiments & Performance

In this section we would like to underline and evaluate the usefulness of our method by suitable experiments (scenes). For this purpose, we have selected four different test scenarios. For all scenarios we use a default time step $\Delta t_{LR} = 0.005$s for the LR simulation, which is further constrained by the CFL condition if necessary. The time step for upsampling is $\Delta t_{HR} = 0.0167$s.

	Particles		Time per Frame	
	LR	HR	LR	HR
Sandcastle	2.4k	90k	9.75ms	7.31ms
Excavator	6.8k	147k	21.4ms	11.2ms
Hourglass	10k	460k	47.4ms	48.5ms
Squirrel	10k	207k	33.0ms	15.8ms

Table 1: Timing results for four different scenes

The density of the granular medium is $\rho = 1600\frac{\text{kg}}{\text{m}^3}$ and the kinetic and static friction coefficients are $\mu_k = 0.3$ and $\mu_s = 0.35$, respectively. In general, we use LR simulation radii r_{LR} between 0.005m and 0.05m. For the upsampling we use a HR particle radius between $r_{HR} = 0.2 \cdot r_{LR}$ and $r_{HR} = 0.4 \cdot r_{LR}$. This leads to an upscaling factor between 15 and 125.

Scene Sandcastle (see Figure 4) consists of a excavator controllable by user interaction and a sand castle with a relatively small number of particles. The LR particle radius is 0.03m and the upsample radius is 0.01m, resulting in 2.4k LR particles to 90k HR particles. This scene serves to demonstrate the real-time capability of our method, in which relatively few elaborately simulated particles can be used to obtain visually pleasing results by upsampling.

Scene Excavator , shown in Figure 1, also contains an excavator. In this example, a radius of 0.02m generates almost three times as many LR particles (6.8k) as in Sandcastle . The upsampling radius 0.008m generates 147k HR particles. This scene is intended to illustrate that our method can be used to simulate even complicated interactions such as the lifting and lowering of granular material with an excavator.

The Hourglass example should on one hand clarify the behavior and runtime of larger particle numbers and on the other hand, more importantly, give an impression of the stable piling of our method. For this purpose 10k LR particles in the hourglass are upsampled to 460k HR particles. Radii of 0.028m and 0.008m are used.

Scene Squirrel shows the two-way coupling between rigid bodies and the granular material. For this purpose, a squirrel is hurled into the sandcastle with an initial velocity. The squirrel is represented by 1.4k particles with a density of $\rho = 5000\frac{\text{kg}}{\text{m}^3}$, which are simulated in the LR simulation. Afterwards, the original squirrel shape is reconstructed by shape matching as described in Section 3, which accomplishes the two-way coupling. All LR simulation particles have a radius of 0.02m and the upsampling radius of the granular particles is 0.0075m. In addition to the 1.4k particles of the squirrel, there are 10k LR granular particles in the simulation, which are upsampled to 207k.

All scenes were calculated on an Intel Core i9-9980HK with 8 cores on the CPU alone. Table 1 shows the per frame timing results for the four different scenes. For the timing of the LR simulation, the

Figure 4: An excavator is moved by user input through a sand castle

Figure 5: An hourglass with sand containing 460k particles.

Figure 6: A squirrel hurled into a sand castle.

run-times of as many simulation steps as necessary to calculate a time difference $t = \Delta t_{HR} = 0.0167$s were combined. It shows that even for relatively high particle counts of 500k a computation within interactive frame rates is possible with our method on consumer hardware without exploiting massive parallelism on GPUs. Especially where the resources of the GPU are already needed for e.g. computationally intensive physically based rendering for realistic visualization, our pure CPU based simulation can show its advantages.

6 CONCLUSION & FUTURE WORK

With this work we have shown that with the help of an upsampling algorithm it is possible to scale up relatively small numbers of particles to produce visually vivid results within interactive frame rates for the simulation of granular material. It was shown that the up-

sampling method described by Ihmsen et. al. [15] for SPH simulations, which are far away from real-time computation times, is very well suited for application in interactive PBD simulations. We believe that this is advantageous for future real-time applications in interaction with granular material.

7 LIMITATIONS & FUTURE WORK

On one hand, in position-based simulations friction is dependent of the number of iterations, which has a negative impact on the correctness of the friction effects in the LR simulation part. The modeled static and kinetic friction effects are only approximations. On the other hand, static friction cannot be modeled correctly with HR upsampling, which prevents stable piling in some situations.

Furthermore, currently only one constant radius for all particles in the LR simulation and another constant radius for all particles in the HR upsampling is possible.

In the future, we plan to take advantage of the high parallizability of the PBD variant described by Macklin et. al. [18] and the trivial parallelizability of the upsampling algorithm to create a GPU-based version of our simulation that can simulate even larger numbers of particles in real time.

ACKNOWLEDGMENTS

REFERENCES

[1] Iván Alduán and Miguel A. Otaduy. Sph granular flow with friction and cohesion. In *Proceedings of the 2011 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '11, pages 25–32, New York, NY, USA, 2011. Association for Computing Machinery.

[2] Iván Alduán, Ángel Tena, and Miguel Otaduy. Simulation of high-resolution granular media. *CEIG 2009: Congreso Español de Informática Gráfica*, 09 2009.

[3] Nathan Bell, Yizhou Yu, and Peter Mucha. Particle-based simulation of granular material. pages 77–86, 01 2005.

[4] Jan Bender, Matthias Müller, and Miles Macklin. A Survey on Position Based Dynamics. In Adrien Bousseau and Diego Gutierrez, editors, *EG 2017 - Tutorials*. The Eurographics Association, 2017.

[5] John Bowers, Rui Wang, Li-Yi Wei, and David Maletz. Parallel poisson disk sampling with spectrum analysis on surfaces. 29(6), December 2010.

[6] Blender Online Community. Blender - a 3d modelling and rendering package. http://www.blender.org, 2021.

[7] William Coombs and Charles Augarde. Ample: A material point learning environment. *Advances in Engineering Software*, 139:102748, 01 2020.

[8] P. Cundall and O. Strack. Discussion: A discrete numerical model for granular assemblies. *Geotechnique*, 30:331–336, 01 1980.

[9] L. Dagum and R. Menon. Openmp: An industry-standard api for shared-memory programming. *Computational Science & Engineering, IEEE*, 5:46 – 55, 02 1998.

[10] Mihai Frâncu and Florica Moldoveanu. Unified simulation of rigid and flexible bodies using position based dynamics. 04 2017.

[11] Gaël Guennebaud, Benoît Jacob, et al. Eigen v3. http://eigen.tuxfamily.org, 2010.

[12] Nicholas John Higham. Computing the polar decomposition with applications. *Siam Journal on Scientific and Statistical Computing*, 7:1160–1174, 1986.

[13] Yuanming Hu, Yu Fang, Ziheng Ge, Ziyin Qu, Yixin Zhu, Andre Pradhana, and Chenfanfu Jiang. A moving least squares material point method with displacement discontinuity and two-way rigid body coupling. *ACM Trans. Graph.*, 37(4), July 2018.

[14] Markus Ihmsen, Nadir Akinci, Markus Becker, and Matthias Teschner. A parallel sph implementation on multi-core cpus. *Comput. Graph. Forum*, 30:99–112, 03 2011.

[15] Markus Ihmsen, Arthur Wahl, and Matthias Teschner. High-Resolution Simulation of Granular Material with SPH. In Jan Bender, Arjan Kuijper, Dieter W. Fellner, and Eric Guerin, editors, *Workshop on Virtual Reality Interaction and Physical Simulation*. The Eurographics Association, 2012.

[16] Chenfanfu Jiang, Craig Schroeder, Joseph Teran, Alexey Stomakhin, and Andrew Selle. The material point method for simulating continuum materials. pages 1–52, 07 2016.

[17] Toon Lenaerts and Philip Dutré. Mixing fluids and granular materials. *Comput. Graph. Forum*, 28:213–218, 04 2009.

[18] Miles Macklin, Matthias Müller, Nuttapong Chentanez, and Tae-Yong Kim. Unified particle physics for real-time applications. *ACM Trans. Graph.*, 33(4), July 2014.

[19] J. Monaghan. Smoothed particle hydrodynamics. *Annual review of astronomy and astrophysics*, 30:543–574, 1992.

[20] Matthias Müller, Bruno Heidelberger, Marcus Hennix, and John Ratcliff. Position based dynamics. *Journal of Visual Communication and Image Representation*, 18(2):109 – 118, 2007.

[21] Matthias Müller, Bruno Heidelberger, Matthias Teschner, and Markus Gross. Meshless deformations based on shape matching. *ACM Trans. Graph.*, 24:471–478, 07 2005.

[22] Rahul Narain, Abhinav Golas, and Ming C. Lin. Free-flowing granular materials with two-way solid coupling. In *ACM SIGGRAPH Asia 2010 Papers*, SIGGRAPH ASIA '10, New York, NY, USA, 2010. Association for Computing Machinery.

[23] Alvaro Sanchez-Gonzalez, Jonathan Godwin, Tobias Pfaff, Rex Ying, Jure Leskovec, and Peter W. Battaglia. Learning to simulate complex physics with graph networks. *Proceedings of the 37th Internation Conference on Machine Learning*, abs/2002.09405, 2020.

[24] Barbara Solenthaler and Renato Pajarola. Predictive-corrective incompressible sph. In *ACM SIGGRAPH 2009 Papers*, SIGGRAPH '09, New York, NY, USA, 2009. ACM.

[25] Alexander Sommer and Ulrich Schwanecke. Lightweight surface and volume mesh sampling application for particle-based simulations. *29. International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision'2021*, pages 155–160, 01 2021.

[26] Deborah Sulsky, Shi-Jian Zhou, and Howard Schreyer. Application of particle-in-cell method to solid mechanics. *Computer Physics Communications*, 87:236–252, 05 1995.

[27] Erik Wallin and Martin Servin. Data-driven model order reduction for granular media. *Computational Particle Mechanics*, 02 2021.

[28] Yonghao Yue, Breannan Smith, Peter Chen, Maytee Chantharayukhonthorn, Ken Kamrin, and Eitan Grinspun. Hybrid grains: adaptive coupling of discrete and continuum simulations of granular media. volume 37, pages 1–19, 12 2018.

[29] Yongning Zhu and Robert Bridson. Animating sand as a fluid. *ACM Trans. Graph.*, 24(3):965–972, July 2005.