UNIVERSITÄT PADERBORN
Die Universität der Informationsgesellschaft

Modeling and simulation of metallic, particle-damped spheres for lightweight materials

D I S S E R T A T I O N

zur Erlangung des akademischen Grades

Doktor der Naturwissenschaften (Dr. rer. nat.)

vorgelegt an der
Fakultät für Elektrotechnik, Informatik und Mathematik
der Universität Paderborn von

Herrn Dipl.-Math. Tobias Steinle

Gutachter:
1. Prof. Dr. Andrea Walther
2. Prof. Dr. Jadran Vrabec

eingereicht am: 28. Oktober 2015
Tag der mündlichen Prüfung: 15. Dezember 2015

Acknowledgements

Foremost, I wish to thank Prof. Dr. Andrea Walther for supervising this thesis, for giving me the opportunity to work in her research group in Paderborn and her guidance, time, patience, support and many hours of fruitful discussions during my time there. I would also like to express my gratitude to her for introducing me to the scientific research world and for giving me the opportunity to get to know and participate in the international research community.

My special thanks also to Prof. Dr. Jadran Vrabec for providing insight into the workings of molecular dynamics simulations, many long discussions, and his open ear for any questions. I'm very grateful to Ulrike Jehring of the Fraunhofer IFAM in Dresden for providing the technical motivation for this thesis along with the experimental data for the simulations and her support with any questions I had.

Also, I would like to thank all members of Prof. Walther's research group at the University of Paderborn for their valuable input and discussions during my studies in Paderborn and for making the group a fun and pleasant place to work at. For technical support, I would like to explicitly thank Dr. Kshitij Kulshreshtha for his patient Linux and Latex support, Dr. Sebastian Grottel of the University Stuttgart for providing custom builds of Megamol for the visualization of my simulation results, Matthias Heinen for his custom data conversion script and the Paderborn Center for Parallel Computing, PC^2, for providing me with access to the OCuLUS computer, which greatly reduced the time required for my simulations.

The University of Paderborn and the Graduate School GSANS provided me with scholarships, for which I am very grateful. I would also like to thank my parents for their encouragement and great support during my whole studies and their faith in me.

Last but not least, I thank God for his great love, wisdom, strength and guidance always.

Zusammenfassung der Arbeit

In vielen technischen Anwendungen spielt heute der Leichtbau eine große Rolle, denn durch Gewichtseinsparungen lässt sich auch Energie einsparen. Allerdings birgt der Leichtbau die Gefahr einer erhöhten Störanfälligkeit gegenüber Vibrationen, die durch die Operation von Maschinen entstehen können. Das Fraunhofer Institut für Fertigungstechnik und Angewandte Materialforschung in Dresden beschäftigt sich mit den Möglichkeiten einer Schwingungsdämpfung durch Verbundwerkstoffe. Dabei wird in die Leichtbaustruktur eine Vielzahl von Hohlkugeln eingebracht, die mit Keramikpartikeln gefüllt sind. Gerät das Material in Schwingung, wird kinetische Energie durch Reibung der Partikel untereinander und mit der Gefäßwand dissipiert.

Diese Fragestellung bildet die technische Motivation für diese Arbeit. Ziel ist, ein Experiment zur Bestimmung des Restitutionskoeffizienten numerisch nachzubilden. Dadurch sollen sich vielfältige Simulationsmöglichkeiten eröffnen, um das Material optimieren zu können. In dem Experiment wird dazu eine einzelne, mit Partikeln gefüllte Hohlkugel aus einer vorgegebenen Höhe fallengelassen und das Verhalten des Verbundes untersucht.

Die Simulation basiert auf einer Diskreten Elemente Methode um die Trajektorien der einzelnen Partikel und der Kugel berechnen zu können. Basierend auf einem Potentialansatz für die Interaktionsberechnung in der Molekulardynamik kann das Reibungsverhalten vielfältig angepasst werden. Das Simulationsvolumen wird durch reflektierende Randbedingungen abgeschlossen und umfasst die Kugelhülle. In dieser Arbeit kam eine hochflexible Speicherstruktur zum Einsatz, um die heterogene Verteilung der Partikel im Raum mit einer effizienten Linked Cell Methode abbilden zu können. Dadurch wird der Rechenaufwand für die Interaktionen stark eingeschränkt, so dass eine in der Partikelzahl lineare Komplexität erreicht wird. Umfangreiche numerische Experimente zeigen den großen Effekt der Partikelfüllung auf das Dämpfungsverhalten.

Summary of the Thesis

Lightweight materials play an ever growing role in today's world. Saving on the mass of a machine will usually translate into a lower energy consumption. However, lightweight applications are prone to develop performance problems due to vibration induced by the operation of the machine. The Fraunhofer Institute for Manufacturing Technology and Advanced Materials in Dresden conducts research into the damping properties of composite materials. They are experimenting with hollow, particle filled spheres embedded in the lightweight material. If the material starts to vibrate, kinetic energy is dissipated via friction of the particles with each other and the spherical enclosure.

Such a system is the technical motivation of this thesis. Ultimately, a numerical experiment to derive the coefficient of restitution is required. With such a tool, multiple experiments can be conducted easily to optimize the damping performance. To that end, a sphere filled with particles is dropped from a certain height and the behavior of the system is observed.

The simulation developed in this thesis is based on a discrete element method to track the individual particle and sphere trajectories. Based on a potential based approach for the particle interactions deployed in molecular dynamics, the behavior of the particles can be controlled effectively. The simulated volume is using reflecting boundaries and encloses the hollow sphere. In this work, a highly flexible memory structure was used with a linked cell approach to cope with the highly flexible mass of particles. This allows for a linear complexity of the method in regard to the particle number by reducing the computational overhead of the interaction computation. Multiple numerical experiments show the great effect the particles have on the damping behavior of the system.

Contents

List of Figures

List of Tables

Nomenclature

α	angular acceleration acting on a particle
ΔT	rebound time
Δt	specified time step during the simulation
ω	angular velocity
σ	particle diameter
τ	torque on a particle
ε_r	coefficient of restitution obtained by experiment
a, a_i	acceleration acting on a body (particle i)
c_s	hollow sphere position
d, d_{ij}	distance of two particles (or, explicitly, particles i and j)
d_s	distance between sphere and fundament
F	force acting on a particle
$F(d)$	force induced by the interaction potential U
F_{ij}	force acting between particles i and j
h	experimental rebound height of the hollow sphere
h_d	experimental drop height
I, I_s	moment of inertia tensor (with rotational axis s)
L	angular momentum of a particle
M	hollow sphere mass
m, m_i	mass of a particle (i)
N	total number of particles
o	spatial orientation of a particle

p_i	particle position in space
q	quaternion for the representation of spatial orientation
R	sphere radius
r_c	cut-off radius for the sphere-fundament interaction
r_c^*	maximal cut-off radius of all particles
r_i^*	composed particle cut-off radius used for the interaction check
r_i^c	cut-off radius of single sphere particle i
r_{ij}	distance vector between particles i and j
s	center of mass of a body
t	time
U	interaction potential between particles
U_b	interaction potential between particles and sphere
U_s	interaction potential between sphere and fundament
V	body volume
v	velocity of a particle
v_s	sphere velocity

Chapter 1

Introduction

Imagine a world where every aspect of life has been optimized to utilize as many lightweight materials as possible. Energy consumption would be lowered dramatically, driving down operating cost while also benefitting the environment. In the search for greener technologies in recent years, research in lightweight materials has become very important. New technologies offer the potential of allowing a much more fuel-efficient operation of machines, such as for personal and public means of transportation and machines used in manufacturing processes. These applications require a high level of precision and reliability to ensure a high quality of the end product as well as low maintenance costs. However, a completely lightweight world would probably also be very noisy. Vibrations, induced by the operation of those machines themselves, pose a problem to achieve these goals. E.g., rattling marks can occur during production in computer numerically controlled (CNC) mills, making the produced part potentially useless. Oscillating parts eventually break at weak spots, e.g., in fasteners. The constant movement is also quite audible.

The classic remedy to vibrations, adding more mass, runs contrary to the goal of low energy consumption. Thus, another approach has to be taken, such as using composite materials. They combine two or more different materials. The Fraunhofer Institute for Manufacturing Technology and Advanced Materials (IFAM) in Dresden has conducted research on the applicability of light materials in the area of manufacturing. They have focused especially on conglomerates that in the best case combine the positive traits of its constituents. Sandwiched materials have become popular in this area because they allow for easy customization. Fraunhofer IFAM Dresden has focused its attention on sandwiched sphere structures, cf. Ref. [20]. These materials use hollow metallic spheres that are encased between, e.g., two

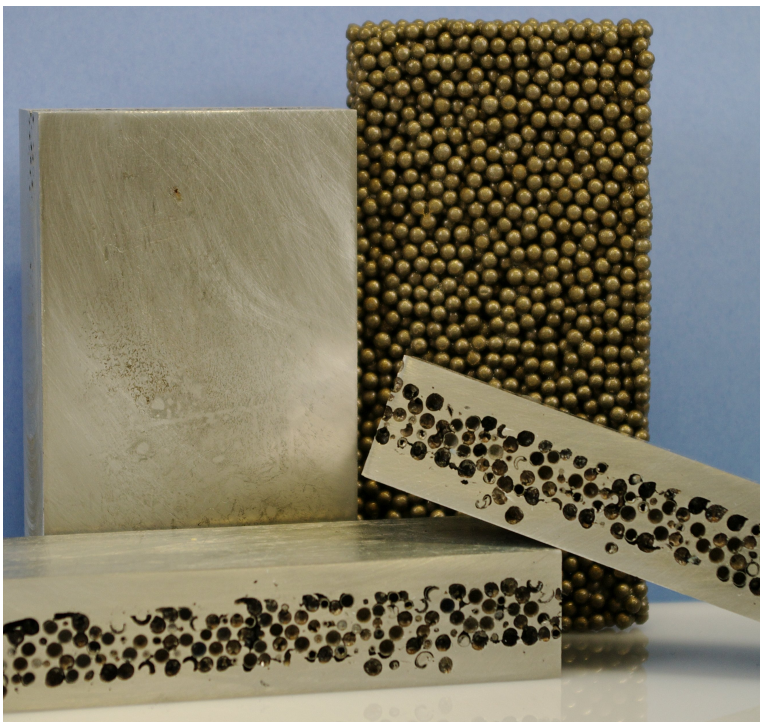


Figure 1.1: Sandwiched sphere structure, ©Fraunhofer IFAM Dresden

sheets of metal that can then be used in a machine casing, cf. Fig. 1.1. There have been experiments to fill the spheres with a material that, while still keeping the overall weight low, can help to dampen vibrations. In the past, there have already been efforts to use so-called particle dampers to eliminate vibration, cf. Ref. [28]. These devices are often encased and attached to the machine externally. Fraunhofer IFAM Dresden is working on manufacturing techniques to include the dampers in the casing itself. The particles inside help vibrations to subside more quickly by substantially converting kinetic energy to heat through friction that occurs between the particles and the hull. The metallic hull at the same time ensures that the particles inside are protected from environmental hazards, such as moisture or solvents, which is important to guarantee the reliability of the damping behavior in the long run.

The positive influence of these particles can be observed in a simple experiment, where a filled sphere is dropped from a certain height to the fundament and the rebound time is measured, i.e., the time it takes between the first and second impact on the fundament. This measurement allows for the derivation of the coefficient of restitution. In experiments, a filled and an empty sphere show a completely different behavior. The empty sphere takes a lot longer to come to rest on the fundament. Hence, the particles

obviously open up possibilities for damping. The connection of time and damping is explained in the forthcoming chapters.

Using particle filled spheres in structures could help to cope with the negative effects of vibrations, such as noise, wear, and tear. However, due to the challenging manufacturing procedure, it is not practical to conduct real-world experiments with many different material combinations. Additionally, the manufacturing process imposes limits on the materials that can be used. New procedures may need to be invented to exploit the potentially excellent damping behavior of new composite materials containing these particles. Therefore, a software simulation tool is desirable that will help to explore and select suitable materials for the particles inside the spheres, leading the way to an efficient design process in new light-weight applications, without the need for great investment in new technologies.

During manufacturing, a basic sphere made of a polymer is covered with a metallic powder and sintered. While the metal sphere is forming, the polymer is chemically transformed and broken down into a powder that remains in the hollow sphere. This complex manufacturing process is being studied at Fraunhofer IFAM Dresden. The sintering process results in many differently shaped and sized particles (cf. Fig. 1.2). Thus, the particle heap inside a hollow sphere may be highly heterogeneous. The effect of the particle mass and shape distribution therefore needs to be analyzed in a systematic way to be able to make recommendations how the particles should be produced with respect to their shape and size.

In the scope of this work, the standard experimental setup is considered to be as follows (cf. Figure 1.3). The drop height of the sphere is 10 cm. The diameter of the hollow sphere is around 3 mm and the particles inside have a diameter in the range of 10^{-1} mm and a mass that is of the order of magnitude of 10^{-7} g. Practically, there are up to around 10^5 particles in the sphere. To cope with this number, a suitable and efficient numerical simulation is required.

For this PhD thesis, a sophisticated model and a corresponding simulation tool was developed that allows for the analysis of a particle filled sphere. In the following chapters, detailed explanations will be given on how this experiment was modeled and simulated. Furthermore, results as well as a comparison to Fraunhofer IFAM Dresden experiments will be presented. It will be shown that this approach can be used to successfully apply numerical tools to analyze the damping behavior of particle filled hollow spheres.

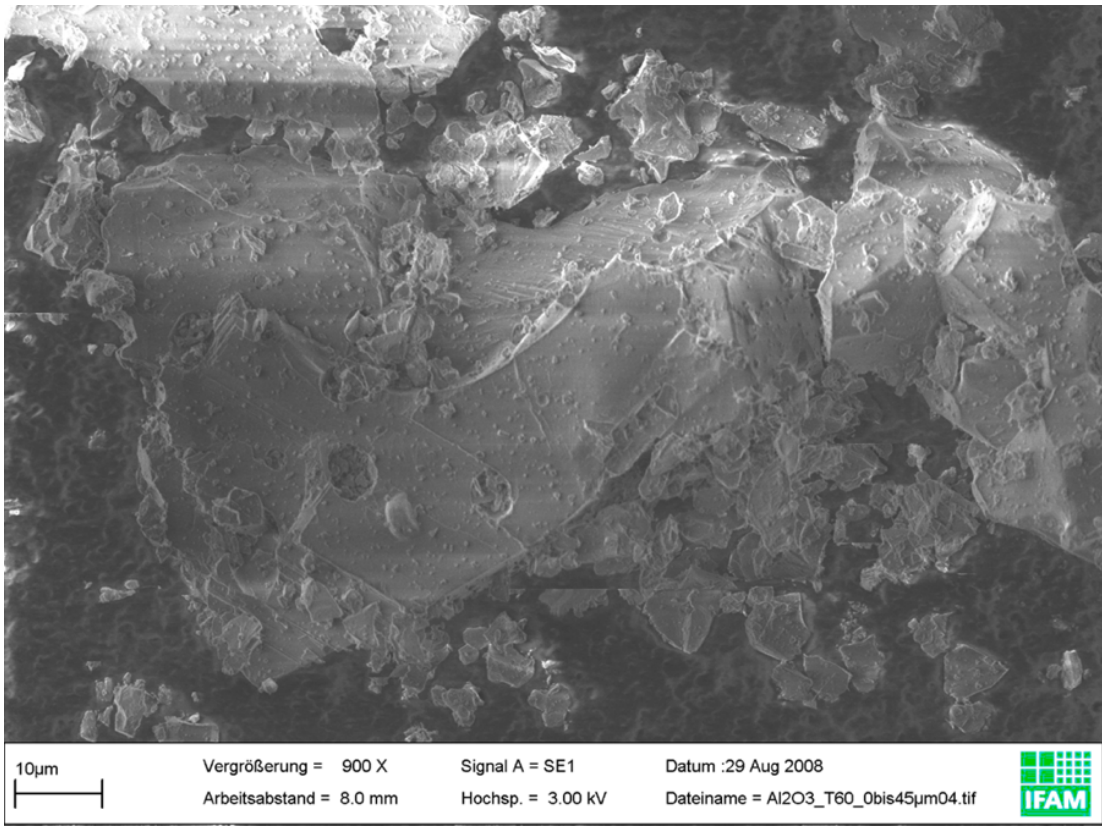
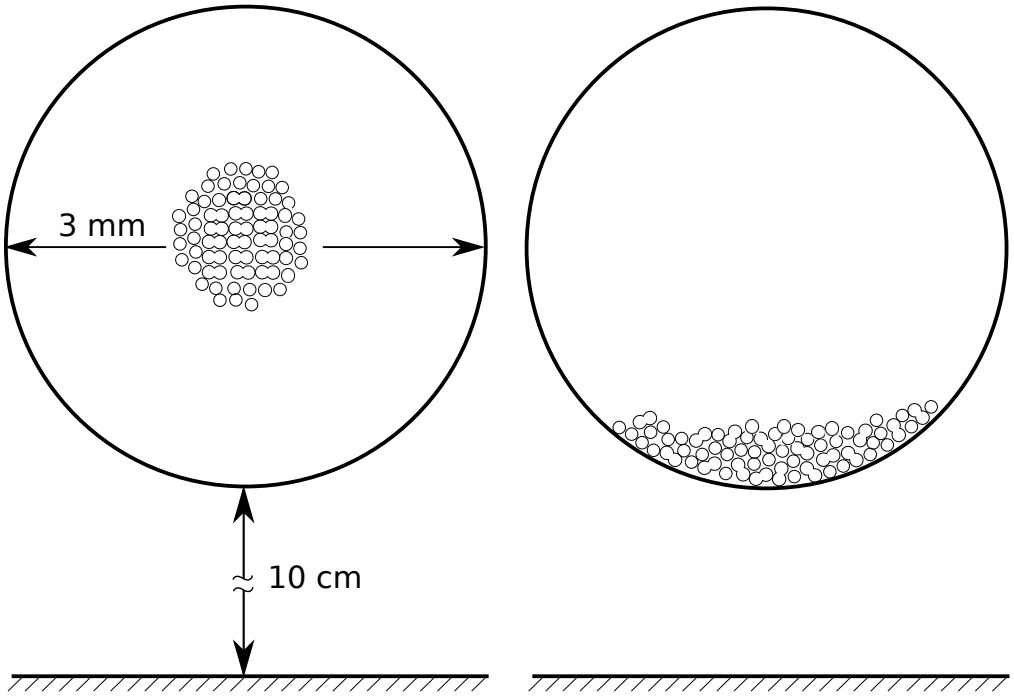


Figure 1.2: Particles in an electron microscope view, ©Fraunhofer IFAM Dresden

Eventually, this opens up the route to optimize such lightweight composite materials to achieve high damping in practical use cases. Chapter 2 explains the basics of molecular dynamics (MD) simulation that is used as the foundation of this work. This includes the discussion of the potential based interaction between particles. Chapter 3 focuses on the model developed for the application of MD towards the simulation of the problem discussed in this thesis. Chapter 4 presents the simulation approaches including the linked cell approach that improves the complexity and therefore the overall runtime of the simulation, and discusses numerical stability, while chapter 5 gives extensive numerical results of the energy damping performance of a particle filled hollow sphere. Finally, chapter 6 presents thoughts on future expansions of the software.



Step I: Initial setup, where the particles are in the center of the sphere

Step II: After the particles have settled at the bottom of the sphere, the sphere is released and the experiment starts

Step III: The sphere drops to the fundament

Step IV: The sphere rebounds and the particles are dispersed inside

Figure 1.3: Setup and process of the experiment to determine the damping behavior by measuring the rebound time

Chapter 2

State of the art

For the numerical simulation of this problem, various approaches can be taken. Any selected method should be able to capture the translational and rotational movement of each individual particle and allow for interactions with the sphere. An obvious and straightforward idea is to use an event driven method, where the time interval that the system is propagated by in each iteration is directly derived from the next interaction that occurs in the whole system. This entails a comprehensive computation to compare all particle trajectories. A collision table needs to be computed upon initialization and needs to be kept updated throughout the simulation, stating the time interval it would take for each particle to collide with all others if they continue on their momentary trajectory. The minimum time in the table would then govern the propagation of the system [16].

An attempt to simulate the particular experiment at hand has first been made in a diploma thesis by Blase [5] using such an event driven approach. In that work, the sphere and particles were modeled as two-dimensional objects only. The trajectory of each particle was computed and to account for the interaction of particles, all trajectories were checked against each other to determine the next collision. The system was then propagated to that point in time, the new trajectories were computed and so on. This particular approach is also known as hard sphere molecular dynamics and was described by [2]. The simulation was able to match the experiment to some extent, but it proved to be difficult to correctly capture the behavior of the particles sliding or rolling around the boundary. Additionally, it is computationally too expensive for the expansion to the three spatial dimensions and large numbers of particles required for the problem posed by Fraunhofer IFAM Dresden.

Moving away from event driven methods, another possibility for a par-

ticle simulation is a time driven approach, where the system is iteratively propagated over a specified time step. A check for possible interactions is performed after each iteration. This chapter will provide an introduction to a corresponding time integration method used for the computation of the dynamic behavior of a particle damped sphere.

2.1 Time driven approaches

Instead of focusing on the exact time of the next interaction and propagating the system to that point in time, a time integration method propagates the simulated system by a specified time step during each iteration regardless of when the next interaction occurs. Time stepping methods are used in a wide variety of applications, such as fluid simulations, astronomy, chemical processes and deformation modeling.

There is a number of methods that can be used for particle simulations. The techniques vary, depending on the size of the system as well as on the fraction of simulation volume occupied by the particles. The main differentiating feature of these methods is the type of interaction laws. A popular and groundbreaking method is the discrete element method (DEM). Such a simulation keeps track of the individual (hence discrete) particle's positions in space and their velocities. After each iteration, the system is analyzed to account for any interactions. This method was introduced by Cundall [7] for the simulation of rock fracture, i.e., the collapses of rock faces. The method allowed to keep track of the individual rocks as they descended a slope. The interaction laws are governed by a soft or hard contact model and are meant to simulate the loss of kinetic energy due to friction, cf. Ref. [27]. The soft model allows the particles to overlap while the hard contact model does not. Cundall described interactions between the elements by a simple contact law [6]. He used the overlap δ_n of two bodies and a stiffness parameter K_n to derive a normal force

$$F_n = \delta_n K_n,$$

and similarly, using a shear stiffness K_s to calculate an incremental shear force proportional to the shearing displacements of the two interacting bodies

$$\Delta F_s = K_s(\Delta\theta_1 + \Delta\theta_2),$$

with the increments of rotation of the two bodies $\Delta\theta_1$ and $\Delta\theta_2$. The Newtonian law of motion served as the basis to simulate the particle system, using the forces acting between the particles to propagate the system over time. A wide variety of methods evolved from that initial idea for different problems. Generally, DEM software is characterized by a high computational effort due to the large number of particles considered and the resulting need for interaction checks. These software packages are particularly popular for industrial applications, such as chute and belt problems in mining operations.

Some numerical schemes have been developed specifically for applications where the particles are densely packed and make up most of the simulated volume, requiring a different mathematical approach. For instance, the contact between particles can be formulated using complementarity constraints which leads to a differential variational inequality problem, cf. Ref. [32].

For certain applications, it can be helpful to substitute several particles by a single representative particle. This is helpful for scenarios, where the movement of the particles resembles a fluid. Smoothed particle hydrodynamics (SPH) is one method that is used in this case. Its governing equations are dominated by the pressure as it originates from the simulation of compressible fluids. Instead of simulating each individual molecule, it is based on the idea to substitute the fluid with discrete particles that represent the properties of their immediate vicinity, cf. Ref. [24]. SPH is an empirical method that revolves around the idea to use approximate equations for the variables that have an easily computable derivative. It facilitates a kernel function $W(x, h_s)$ for the approximation. To interpolate any quantity V that depends on spatial coordinates x over a domain Ω, SPH evaluates

$$V_{SPH}(x) = \int_{\Omega} V(x')W(x - x', h_s)dx',$$

where h_s is called the smoothing length of the kernel function W. Thus, the physical properties of a particle are influenced only by those of the particles around it. The neighboring particles are determined by the kernel which is usually symmetric and non-negative with

$$\int W(x - x', h_s)dx' = 1.$$

The integral can be approximated by

$$V_{SPH}(x) = \sum_b V_b \frac{m_b}{\rho_b} W(x - x_b, h),$$

where the quantity V_b, mass m_b, density ρ_b and spatial position x_b are the values of the b neighboring particles that all influence the considered particle. For many applications, the Gaussian kernel is selected. Another popular kernel is a cubic spline kernel that reduces the influence of particles on the properties of the considered particle to zero if they are further away than $2h_s$, cf. Ref. [25]. SPH is popular for use in special effects and animation, as it gives good results for the visual representation of liquids without the requirement for extensive computation time.

According to Fraunhofer IFAM Dresden, there are approximately 10^5 particles (occupying up to about 20 % of the volume) inside the cavity. The particles do not represent a fluid and can move freely in the hollow sphere. Thus, the use of a DEM type method is plausible.

2.2 Molecular dynamics

Molecular dynamics (MD) is a discrete element method that is able to consider large numbers of molecules. It keeps track of every particle's spatial position and rotational-translational propagation. Initially it was only possible to simulate very few particles, starting at about 100 in 1959 (cf. Ref. [1]), but modern computers are now able to deal with very large particle numbers. There are implementations for large problems that utilize parallel supercomputers for trillions of atoms [18]. Still larger systems require more memory and computational power and will continue to greatly benefit from the technological evolution in the future. For the case considered here, approximately 100000-200000 particles have to be simulated. This is well within the magnitude that MD can be used for. The simulated bodies will not represent tiny molecules but rather the particles inside the sphere. The flexibility in modeling different molecules will be beneficial for the composition of the heterogenous powder inside the sphere. MD does not necessarily require information such as density or pressure inside the simulation volume to correctly predict the movement of the particles. This allows to simulate sparsely filled volumes with variable particle distributions. Thus, an adapted MD variant is a suitable choice for the observation of the move-

ment of particles in a hollow sphere considered for this work.

MD are governed by the Newtonian equations of motion for both the translational and rotational part of the particle motion [18]. This law states (cf. Ref. [23]):

Theorem 2.1. *(Newton's law)*
The force F acting on a body of mass m and the acceleration a caused by it act in the same direction and are proportional.

$$F = ma.$$

This equation is used to describe the motion of all particles in the MD ensemble and can be written as a second order ordinary differential equation

$$\ddot{p} = a = \frac{F}{m}.$$

Transforming it to a system of first order differential equations results in

$$\dot{v} = \frac{F}{m}, \quad \dot{p} = v, \tag{2.1}$$

with appropriate initial conditions for position $p(0)$, velocity $v(0)$ and force $F(0)$. For instance, in MD the initial velocity of the molecules might be assigned at random within a certain interval, with initial $F(0) = 0$ and the positions taken from an initial lattice configuration. The forces F are derived from a potential acting between the different particles during the propagation of the system. This will be expanded upon in section 2.2.

Time integration

The system (2.1) has to be solved numerically. There are many options to do this with a varying degree of accuracy. For instance, starting with a Taylor expansion

$$p(t + \Delta t) = p(t) + \Delta t \dot{p}(t) + O(\Delta t^2),$$

and after omitting all but the first two terms, an approximate solution p is achieved. Thus, with the Taylor expansion used on eqs. (2.1) to solve for p

and v, the desired values can be approximated by

$$v(t+\Delta t) \approx \quad v(t) + \Delta t \frac{F}{m}, \tag{2.2}$$
$$p(t+\Delta t) \approx \quad p(t) + \Delta t v(t). \tag{2.3}$$

This approximation for the solution of a differential equation is known as the explicit Euler method. Due to its construction, it is of first order accuracy.

To achieve a higher accuracy, the solutions are evaluated alternatingly, with eq. (2.2) computed on the half step and then evaluating eq. (2.3) on the full step. Also on the full step, and using the new particle positions, the forces are re-evaluated. Then, the velocity is computed again in the next half step using the updated forces. This is called Leapfrog algorithm. Consequently, the translational motion of a body is computed by

$$v(t+\frac{\Delta t}{2}) \quad = v(t-\tfrac{\Delta t}{2}) + \Delta t \tfrac{F(t)}{m}, \tag{2.4}$$
$$p(t+\Delta t) \quad = p(t) + \Delta t v(t+\tfrac{\Delta t}{2}). \tag{2.5}$$

During a simulation, v can be synchronized with p by just adding $\frac{\Delta t}{2}\frac{F(t)}{m}$ at the beginning of the iteration and then, after the p and F updates, another half step is performed at the end of the simulation to approximate v. This allows for easier extraction of the desired results, such as temperature, pressure and energy at the full time step. An alternative to this method is called the Velocity-Verlet (or Störmer-Verlet) method [14], which approximates the full step velocity using the forces calculated in the previous and current step. This work makes use of the Leapfrog method.

Theorem 2.2. *(Leapfrog accuracy)*
The leapfrog scheme as described by eqs. (2.4) *and* (2.5) *is a second-order method.*

Proof. Although the scheme is derived from a Taylor expansion truncated after the first term, and therefore it would be natural to conclude that it is only a first order method, the method is in fact a second order method. Substituting $v(t-\frac{\Delta t}{2})$ in eq. (2.4), using the previous iteration of eq. (2.5) yields

$$v(t+\frac{\Delta t}{2}) = \frac{p(t)-p(t-\Delta t)}{\Delta t} + \Delta t \frac{F(t)}{m}.$$

This is then substituted into eq. (2.5) again

$$p(t+\Delta t) = p(t) + \Delta t\left(\frac{p(t)-p(t-\Delta t)}{\Delta t} + \Delta t\frac{F(t)}{m}\right),$$

finally leading to

$$\frac{p(t+\Delta t) - 2p(t) + p(t-\Delta t)}{\Delta t^2} = \frac{F(t)}{m}. \tag{2.6}$$

This equation is a finite difference scheme for the second derivative, i.e. Newton's second law, proving that the Leapfrog method does approximate it. To calculate the error of this scheme, the Taylor expansions for $p(t \pm \Delta t)$ are introduced

$$p(t+\Delta t) = p(t) + \Delta t\dot{p}(t) + \frac{\Delta t^2}{2}\ddot{p}(t) + \frac{\Delta t^3}{6}\frac{\partial^3 p(t)}{\partial t} + \frac{\Delta t^4}{24}\frac{\partial^4 p(t)}{\partial t} + \dots,$$
$$p(t-\Delta t) = p(t) - \Delta t\dot{p}(t) + \frac{\Delta t^2}{2}\ddot{p}(t) - \frac{\Delta t^3}{6}\frac{\partial^3 p(t)}{\partial t} + \frac{\Delta t^4}{24}\frac{\partial^4 p(t)}{\partial t} - \dots.$$

These are substituted into eq. (2.6), yielding

$$\ddot{p}(t) + \frac{\Delta t^2}{12}\frac{\partial^4 p(t)}{\partial t}(+\dots) = \frac{F(t)}{m} + \epsilon_T. \tag{2.7}$$

Thus, the error ϵ_T is of order $O(\Delta t^2)$, establishing the leapfrog scheme as a second order method. □

The error introduced by the finite difference scheme discussed above is not the only one present in the system. Another aspect that needs to be addressed is numerical stability. This concept describes how the errors are propagated, i.e., whether they are amplified during simulation as time progresses. A method is numerically stable, when these errors are not amplified. Stability is the major concern in MD simulations and the alternating evaluation of eqs. (2.4) and (2.5) in the leapfrog scheme results in higher numerical stability compared to just evaluating both equations together [8]. The leapfrog algorithm is conditionally stable, the condition of which is the step size. Stability will be discussed further in section 2.3.

For the rotational movement of a particle, a similar scheme can be followed for its angular velocity and orientation in space. This section introduces the mechanical concepts used in rotational movement. The governing

model is the equation of rigid body rotational movement and is given as

$$\tau = I\alpha. \tag{2.8}$$

Here, τ denotes the torque, while α and I are the angular acceleration and the moment of inertia tensor, which is explained below. This can be rewritten once more as a first order system

$$\dot{L} = \tau, \quad \dot{o} = \omega \times o, \tag{2.9}$$

where o is a vector fixed in the rotating body, representing the orientation of the particle in space, and L the angular momentum. The angular velocity ω links the two equations and can be evaluated as

$$\omega = I^{-1}L.$$

The moment of inertia tensor I acts as a descriptor for the resistance of a specific body to a change in the rotation about a certain axis s. For each individual particle shape, a different tensor needs to be derived. It is defined by an integral over the volume V of the body

$$I_s = \int_V \rho(r) r^2 dV, \tag{2.10}$$

where r is the perpendicular distance to the axis s and $\rho(r)$ the mass density at each point of the body. Thus, it depends on the distribution of mass in a body and the axis of rotation. For instance, for a solid sphere with radius R and mass m, the tensor is

$$I_s = \frac{2}{5} m R^2,$$

for any axis s through the center because of the symmetry of the sphere.

For spatial rotational movement, the scalar moments are included in a 3×3 matrix, describing the inertia of the body for rotation about a reference system of axes.

Definition 2.1 (Cartesian system of axes)**.** *A system of axes or reference system in n-dimensional space consists of an origin $\mathcal{O}$ and n orthogonal unit vectors $o_1, o_2, \ldots, o_n$ in that point. Any point in that system can be described by a linear combination $\mathcal{O} + a_1 o_1 + a_2 o_2 + \cdots + a_n o_n$ of those vectors, with $a_i \in \mathbb{R}$ and is written as a n-tuple $(a_1, a_2, \ldots, a_n)$.*

Typically, several reference systems are used for a simulation, i.e. one system for the definition of molecules and another one for the global position of the particles. The molecules in MD are made up of atoms. Atoms concentrate their mass in their nucleus. Therefore, MD uses the idealization of point masses. Consequently, a molecule consisting of just one atom (e.g. a noble gas) does not have any inertia to rotation at all, reducing I to zero. Still, the tensor has to be specifically evaluated for each individual particle type. For simplicity, the particles are assumed to be rigid, so the tensor only has to be evaluated once at the beginning of the simulation.

The tensor is not the same for any reference system in the body and it is best to evaluate it in a system attached to the center of mass as is shown below. To achieve this, the center of mass of the molecule (or any other particle) has to be calculated. Here, the case of point masses is considered. In each particle's body fixed system of axes it can be evaluated as

$$s = \frac{1}{m} \sum_{i=1}^{n} m_i p_i,$$

with individual cartesian positions p_i and masses m_i of the n spheres that define that particle type with a total mass m. The spheres with coordinates p_i are then transformed to a new system of axes (x, y, z) with the origin at s. The result is

$$\tilde{p}_i = p_i - s.$$

In any arbitrary system of axes, with the center of mass as the origin, the inertia tensor can then be computed using the new coordinates $\tilde{p}_i := (x_i, y_i, z_i)$ of the n individual point masses as

$$I_{xx} = \sum_{i=1}^{n} m_i (y_i^2 + z_i^2), \quad I_{yy} = \sum_{i=1}^{n} m_i (x_i^2 + z_i^2), \quad I_{zz} = \sum_{i=1}^{n} m_i (x_i^2 + y_i^2),$$
$$I_{xy} = I_{yx} = \sum_{i=1}^{n} m_i x_i y_i, \quad I_{xz} = I_{zx} = \sum_{i=1}^{n} m_i x_i z_i, \quad I_{yz} = I_{zy} = \sum_{i=1}^{n} m_i z_i y_i,$$

with rotational axes x, y and z. The tensor is written as a symmetric matrix

$$I = \begin{pmatrix} I_{xx} & I_{xy} & I_{xz} \\ I_{yx} & I_{yy} & I_{yz} \\ I_{zx} & I_{zy} & I_{zz} \end{pmatrix}.$$

The particles in this work are built up from multiple spheres. However, the origin of the system used to define the particle types does not necessarily coincide with the center of mass. Thus, the particle system either needs to be transformed to a mass centric system as described above, or the inertia tensor needs to be transformed after it has been evaluated in a system with any origin.

The Huygens-Steiner theorem can be used to evaluate the moment of inertia tensor in a new reference system, once it has been calculated for the body in any other system of axes, cf. Ref. [29].

Theorem 2.3. *(Huygens-Steiner)*
If the moment of inertia tensor I_{ij} is known in the center of mass in a body with mass m described in a body fixed system of axes (x, y, z) and the tensor needs to be reevaluated at another point that is moved from the center of mass by a vector a, the tensor can be reevaluated as

$$\tilde{I}_{ij} = I_{ij} + m(a^2\delta_{ij} - a_i a_j),$$

with $i, j \in \{x, y, z\}$ and Kronecker symbol δ_{ij} and scalar product $a^2 =< a, a >$.

For instance, the particle in Figure 2.1 consists of 5 identical spherical bodies with mass $m_1 = m_2 = ... = m_5$ arranged in a T shape with point masses at the coordinates $(0,0,0), (1,0,0), (2,0,0), (1,1,0)$ and $(1,2,0)$ in a global reference system. The center of mass is calculated using the coordinates (x_i, y_i, z_i) in that system

$$s = \frac{1}{5m_1} m_1 \left((0,0,0) + (1,0,0) + (2,0,0) + (1,1,0) + (1,2,0)\right) = (1, 0.6, 0).$$

If the original coordinates are used, the inertia tensor evaluated at the origin is

$$I = \begin{pmatrix} 5 & 3 & 0 \\ 3 & 7 & 0 \\ 0 & 0 & 12 \end{pmatrix}.$$

Using Theorem 2.3, the tensor can now easily be re-evaluated in the center

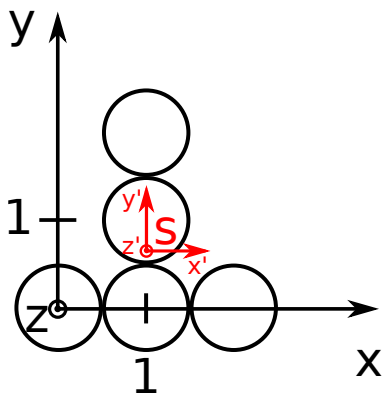


Figure 2.1: Particle consisting of five spheres described in two coordinate systems; the system $(S, \{x', y', z'\})$ is fixed in the center of mass and the axes coincide with the principal axes of the body

of mass with $a = (1, 0.6, 0)^T$, which yields

$$I = \begin{pmatrix} 3.2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 5.2 \end{pmatrix}.$$

Using an arbitrary reference system for the inertia tensor computation generally does not produce such a simple form. In those cases, to simplify the moment of inertia tensor to only include the diagonal elements, it needs to be transformed into a new reference frame, called the system of principal axes, which is the coordinate system where the axes correspond to the principal axes of the body. This system always has the origin at s.

To accomplish the alignment of the axes, the eigenvalues of the tensor have to be calculated along with their corresponding eigenvectors. The eigenvectors are the columns of the tensor aligned to the principal axes. If all eigenvalues are equal, the rotating body is symmetrical.

The tensor together with eqs. (2.9) allows for the full description of the rotational movement.

However, despite of the simpler inertia tensor in the center of mass and the system of principal axes , the computation of the rotational orientation in space remains less than optimal. The angular velocity ω, which is required in order to update the orientation o of the particle, is dependent on the inertia tensor, which, in the *global* reference frame, is dependent on the spatial orientation that usually changes over time. This means that the angular velocity would need to be re-evaluated continuously by a constant use

of rotational matrices for transforming it between the body-fixed system, where the inertia tensor does not change, and the global system, to obtain the orientation of the particle in space. This additional computational effort can be avoided.

Quaternions

To account for the rotational movement of particles, it is reasonable to use a quaternion representation of the orientation in space. Quaternions, first introduced by Hamilton in 1843, expand on complex numbers and are often used in inertial navigation, mechanical design and robot dynamics [26] to represent rotational movements and spatial attitude. They are defined using three units of imaginary numbers, denoted by coefficients i, j and k.

Definition 2.2. *(Quaternion)*
The commonly used representation of a quaternion is

$$q = q_0 + iq_1 + jq_2 + kq_3 := q_0 + \hat{q},$$

with factors $q_i \in \mathbb{R}$ and imaginary units i, j and k. The quaternion norm is defined as

$$\|q\| = q_0^2 + q_1^2 + q_2^2 + q_3^2.$$

To be able to multiply quaternions, the imaginary units must satisfy the Hamiltonian multiplication rules.

Definition 2.3. *(Hamiltonian products)*
For the products of the quaternion imaginary units i, j and k, the following rules apply

$$\begin{aligned} i^2 = j^2 = k^2 = ijk &= -1, \\ ij = k &= -ji, \\ jk = i &= -kj, \\ ki = j &= -ik. \end{aligned} \tag{2.11}$$

Unit quaternions with $\|q\| = 1$ can be used to describe a rotation of a vector $x \in \mathbb{R}^3$ simply by quaternion multiplication. For the rotation, the vector x must be transformed into a quaternion $\bar{x}$, cf. Ref. [22].

Definition 2.4. *(Representation of a $\mathbb{R}^3$ vector as a quaternion)*
A vector $x = (x_1, x_2, x_3)^T \in \mathbb{R}^3$ can be represented as a quaternion $\bar{x}$ using

$$x = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = 0 + ix_1 + jx_2 + kx_3 := \bar{x}.$$

Such quaternions with a real part zero are also called pure quaternions. With this representation, the rotated vector x^r can be calculated by the multiplication

$$x^r = qxq^*, \tag{2.12}$$

where q^* is the conjugate quaternion of q

$$q^* = q_0 - iq_1 - jq_2 - kq_3.$$

Naturally, x^r will also be a quaternion. It should be noted that the multiplication of quaternions is not commutative due to the multiplication rules of the imaginary units in eq. (2.11), thus, the order of multiplication is important to obtain the correct result of eq. (2.12).

Definition 2.5. *(Quaternion multiplication)*
The product pq of two quaternions p and q evaluates as

$$pq = p_0q_0 - \hat{p}\hat{q} + p_0\hat{q} + q_0\hat{p} + \hat{p} \times \hat{q},$$

with the cross product

$$\hat{p} \times \hat{q} := \begin{vmatrix} i & j & k \\ p_1 & p_2 & p_3 \\ q_1 & q_2 & q_3 \end{vmatrix},$$

and the scalar product

$$\hat{p}\hat{q} := p_1q_1 + p_2q_2 + p_3q_3.$$

Note the omission of the imaginary units.

For the purpose of the simulation, the multiplication can also be written using the matrix notation.

Proposition 2.1 (Matrix vector multiplication scheme)**.** *Given two quaternions p and q, their quaternion product is equivalent to*

$$pq = \begin{pmatrix} p_0 & -p_1 & -p_2 & -p_3 \\ p_1 & p_0 & -p_3 & p_2 \\ p_2 & p_3 & p_0 & -p_1 \\ p_3 & -p_2 & p_1 & p_0 \end{pmatrix} \begin{pmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{pmatrix}. \tag{2.13}$$

With these rules in place, the computational description of the rotation in (2.12) is complete.

Of course, to specify the specific rotational axis represented by the unit quaternion in that equation, the following definition is required.

Definition 2.6. *(Quaternion rotation)*
To perform a rotation about a rotational axis that is described by a unit vector $d = (d_1, d_2, d_3)^T$ and an angle ϕ, the quaternion q in eq. (2.12) *is*

$$q = \cos\frac{\phi}{2} + \sin\frac{\phi}{2}(id_1 + jd_2 + kd_3). \tag{2.14}$$

An advantage of the quaternion representation for a rotation is that it is not as computationally demanding as using a rotation matrix. However, if required, for a given unit quaternion, a rotation matrix can be extracted that performs the same rotation [2].

Proposition 2.2. *(Rotational matrix)*
A quaternion with norm $\|q\| = 1$ is equivalent to a rotational matrix

$$A_{rot} = \begin{pmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1q_2 - q_0q_3) & 2(q_1q_3 + q_0q_2) \\ 2(q_1q_2 + q_0q_3) & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2(q_2q_3 - q_0q_1) \\ 2(q_1q_3 - q_0q_2) & 2(q_2q_3 + q_0q_1) & q_0^2 - q_1^2 - q_2^2 + q_3^2. \end{pmatrix}.$$

Quaternions are a good choice for numerical calculations when working with rotational movement, because they do not possess the singularity that the Euler angle representation suffers from. This is known as gimbal lock and may cause the software to malfunction, cf. Ref. [30]. In the course of this work, Fincham's leapfrog rotational algorithm [10] was used to evaluate the rotational orientation in space and the angular velocity of all particles in the simulation. It makes use of the quaternion representation described above and suggests a leapfrog scheme similar to the translational one. With this framework, the movement of each particle can be simulated.

Force computation

Equation (2.4) contains a force term. This force represents the interaction of the particles with each other. In MD, this is derived from a potential U acting between two particles with distance d. The potential is related to the force through the first derivative

$$F(d) = -\frac{\partial U(d)}{\partial d}. \tag{2.15}$$

There are many different potentials that are used, pair-potentials as well as multi-body-potentials. A popular potential [15] is the Lennard-Jones (LJ) or (12,6) potential, cf. Figure 2.2. It was introduced by the mathematician Lennard-Jones in 1924 and describes the highly complex interactions of atoms in a simplified manner. It is defined as

$$U(d) = 4\varepsilon \left(\left(\frac{\sigma}{d}\right)^{12} - \left(\frac{\sigma}{d}\right)^{6} \right), \tag{2.16}$$

where ε denotes the depth of the potential well and σ the diameter of the considered particle. The parameter ε indicates the magnitude of binding forces between atoms. Since the force acting between particles is defined as

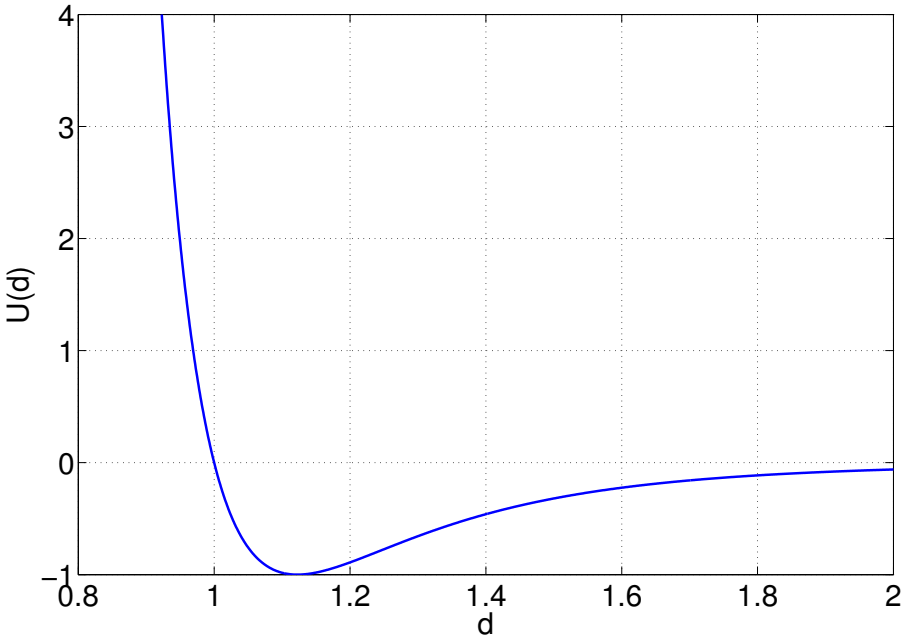


Figure 2.2: Lennard-Jones potential with parameters $\varepsilon = 1$, $\sigma = 1$

the negative first derivative of U, the 12th power term results in a repulsive force between particles while the 6th power term results in an attractive force. The attraction is an approximation of the dispersive force and the other one is an approximation of the Pauli repulsion, induced by overlapping electron orbits of the atoms concerned. At the potential minimum at $d = 2^{\frac{1}{6}}\sigma$ of U, the dominating term changes. The LJ potential is especially well suited for the simulation of noble gas atoms, cf. [33].

Using this interaction potential and taking into account eq. (2.15), the force induced between interacting particles i and j with a distance d_{ij} is given as

$$F_{ij}(d_{ij}) = 24\varepsilon \left(2 \left(\frac{\sigma}{d_{ij}} \right)^{12} - \left(\frac{\sigma}{d_{ij}} \right)^{6} \right) \frac{r_{ij}}{d_{ij}^2}. \tag{2.17}$$

The vector $r_{ij} = p_j - p_i$ is the direction of the induced force. The potential acts on all particles and in theory, an interaction occurs between all particles. Naturally, this leads to high computational costs. Therefore, a cut-off-parameter $r_i^c > 0, r_i^c \in \mathbb{R}$ is introduced that limits the reach of the potential around each particle, which can be different depending on the particle type. Only particle pairs (i, j) with a distance $d_{ij} < \min(r_i^c, r_j^c)$ are explicitly considered to interact, thus the cut-off radius resembles a sphere of influence around each particle, similar to the function of the kernels in SPH. Therefore, the force $F_{\text{MD}}(i, j)$ often used in MD is

$$F_{\text{MD}}(i,j) = \begin{cases} F_{ij}(d_{ij}), & d_{ij} < \min\{r_i^c, r_j^c\} \\ 0, & \text{otherwise.} \end{cases} \tag{2.18}$$

Boundary conditions

In MD, the simulation volume is often considered to be just a very small part of an infinite space among which the molecules are equally distributed. The material, be it a gas, liquid or solid, stretches out in all directions and the behavior inside is expected to be largely homogeneous. This means that this large space can be filled by fitting a much smaller simulation volume or cell inside and multiplying it in every direction. Periodic boundary conditions perfectly support this concept by treating a particle that exits the cell on one side as a particle that is entering from the neighboring cell on the opposite side, cf. Figure 2.3. Periodic boundaries can be easily implemented. For this

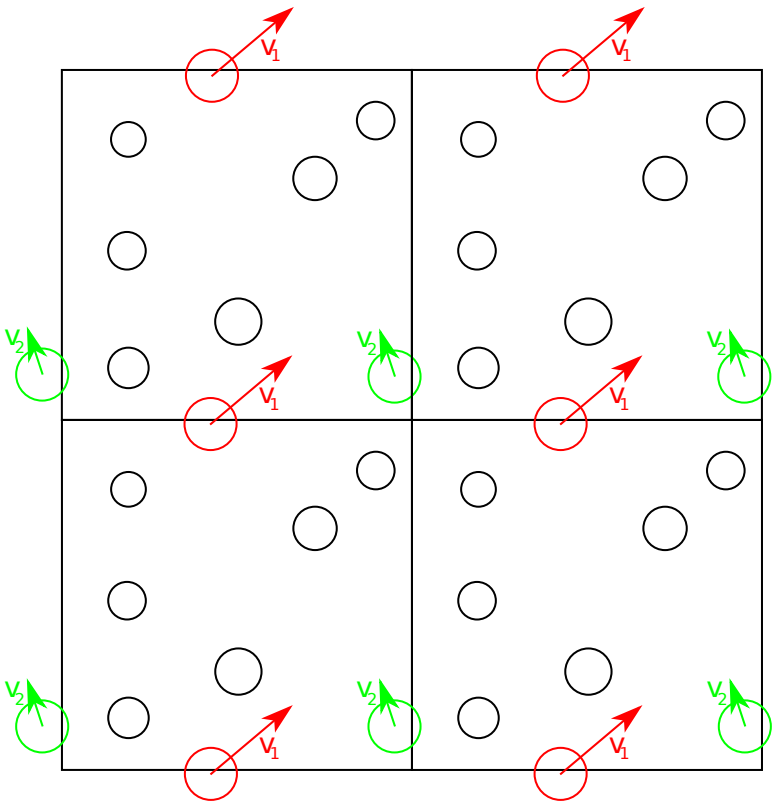


Figure 2.3: Periodic boundary conditions

purpose, the simulation box could be set such that one corner of the cuboid coincides with the origin, for instance normalized to $(0,1) \times (0,1) \times (0,1)$. The boundary condition can now be easily enforced on the particle positions by calculating

$$p(i) = p(i) - \lfloor p(i)/l \rfloor l,$$

where l is the length of the cell box and $\lfloor p \rfloor$ calculates the largest integer p_l so that $p_l < p$. This operation is performed in each iteration.

Reduced units

The particles in MD have a very small size and mass. Very small numbers may introduce instability in numerical computations on computers as they are prone to subtractive cancellation, i.e., terms cancel themselves out during subtraction. This is termed underflow, which results in inexact computations because of vastly differing scales of numbers used in an operation. For instance, subtracting an extremely small number from a large number will not give the correct result when the precision of the employed arith-

metics is insufficient to represent all digits. Therefore, reduced units are used to scale the relevant quantities to a safe number range and set the stage for all calculations. In MD, the dimensions of the molecules define the reduced units, cf. Table 2.1. E.g., a commonly used time step in MD is in the range of one to five femtoseconds. All relevant parameters, such as the velocities, are therefore calculated in the magnitude regime set by these reduced units and have to be carefully chosen. If the velocities, for instance, would still have a unit such as m/s, they would be extremely small and there would be a good chance that they may underflow to 0 when used in actual numerical computations. All constants, such as the standard gravity g, and other simulation parameters, such as positions and potentials, are also scaled accordingly and converted to the same reference system. The concept of reduced unit remains relevant for this work and more details on the reduced units used are discussed in chapter 4.

Variable	Description	Unit
σ	particle diameter	10^{-10} m (or 1 Å)
m	particle mass	10^{-24} g
Δt	time step	10^{-15} s

Table 2.1: Typical units in molecular dynamics

2.3 Numerical properties of molecular dynamics

MD is based on a symplectic integration method such as the leapfrog integrator. This means, that energy is conserved in the system. However, when using computers, numerical errors are introduced because of the inexact operations and can accumulate. Thus, the energy will not remain constant as the simulation progresses. This can be prevented by using a small time step and a correction once every few steps, e.g., by adjusting the temperature using a thermostat that allows the overall kinetic energy to only fluctuate slightly around a specified constant value, cf. Ref. [19]. In the case considered in this work, external forces act upon the system in the form of gravity and friction, where the effect of the numerical error is much less pronounced and thus, a thermostat is not essential. The time step must still be selected very carefully, as will be discussed later in chapter 4.

Chapter 3

Modeling of a particle-filled sphere

Molecular dynamics is a method that is tailored for the use with very small particles on the nano scale, representing atoms. On that scale, the physical model needs to be a different one than in the case of much larger or macroscopic particles. For instance, attractive forces between molecules need to be taken into account as well as forces induced by charges that the molecules carry. These effects do not play a role on the comparatively large scale considered here. Instead, a form of friction must be introduced. Also, the boundary conditions need to be reflective instead of periodic to allow for the spherical enclosure.

This chapter discusses the modeling of the particles as well as the handling of boundary conditions used in this work. Furthermore, continuous (potential based) and discrete possibilities will be discussed for the interactions that drive the dampening effect. A model is suggested to provide a consistent method for the simulation.

The goal of the numerical simulation in this work is to model the following experiment:

Definition 3.1. *(Drop experiment I)*
A hollow sphere with radius R and mass M is filled with N particles of varying size and shape. The sphere is dropped onto a fundament from height h_d. The time ΔT between the first two bounces of the sphere onto the fundament is called the rebound time. This is recorded to derive the coefficient of restitution of the combined sphere-particle system.

The first step in modeling the experiment is to allow for a detailed representation of the particles.

3.1 Particles

To account for the heterogeneous particles in the model, they are built up similarly to molecules that are made up of different atoms, each idealized as a sphere. Therefore, the concept of defining molecules can conveniently be carried over to the present model, where the particles are much larger. The most basic particle is thus also a sphere, while more complex particles are modeled as combinations of spheres, just like a molecule, cf. Fig. 3.1. Every particle has its own fixed system of axes in which the positions of all spheres that make up the particle are configured. It is possible to define several different particle shapes and sizes for a simulation run via an input data file, providing a heterogeneous mixture of particles inside the sphere. To keep track of all particles, they are denoted by a particle number i and are characterized by their position of the center of mass p_i, velocity v_i and the rotational equivalent orientation o_i and angular velocity ω_i. The hollow sphere is also characterized by its position c_s and velocity v_s.

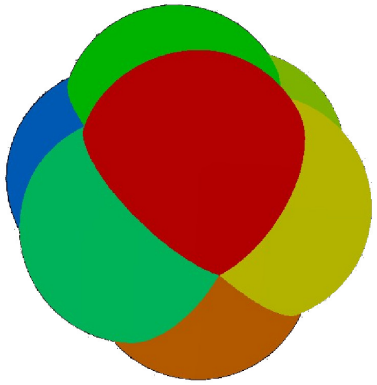


Figure 3.1: Example of a cube shaped particle consisting of eight spheres

Interaction potential

In theory, any interaction potential has to be evaluated between each particle pair because it has an infinite range. However, considering an individual particle with diameter σ_i, most particles in the system are very far away and will therefore hardly interact, unless a long reaching potential is desired. To reduce the computational effort, the concept of a cut-off radius $r_i^c \geq \sigma_i$ explained in the previous chapter carries over from MD into this new approach. Particles with a distance $d_{ij} > r_i^c$ were treated as interaction-free.

Consequently, a larger cut-off radius r_i^* attached to the center of mass is required for more complex particles that encloses all individual spherical parts. This is required in order to capture potential interactions that take place with outlying parts of interaction partners that may touch before their centers of gravity get close enough, e.g. in rod shaped particles. This larger cut-off radius must be evaluated only once at the start of the simulation for each particle type. When checking for interactions, r_i^* is used in a first step. If that check is positive, the interaction forces can be evaluated using the exact distances of each individual spherical part contained in the particles. For that computation, the specific cut-off radii of the individual parts concerned are used.

In simulations with particles of different size, the cut-off radius of the smaller particle is selected when checking for interactions. This eliminates the evaluation of potential interactions where small particles that may momentarily enter a larger particle's cut-off radius could interact, even when their trajectories would not cross. The position of the sphere with the greatest distance D_{max} from the center of mass s plays the dominant role in the calculation of r_i^*

$$D_{max} = max\{|s - p_1|, |s - p_2|, \ldots, |s - p_n|\}.$$

Consequently, the body's cut-off radius r_i^* must satisfy

$$r_i^* > D_{max} + r_i^c, \tag{3.1}$$

in order to detect any interaction that may occur. For simple spherical particles, $r_i^* = r_i^c$. For complex particles, r_i^c is selected as the cut-off radius of the largest spherical part in that particle, if the parts have different diameters.

In the considered scenario, the particles are much larger than atoms and it is therefore sensible to assume that they do not exhibit significant attractive forces. Therefore, a different potential can be used, i.e. the LJ potential without the attractive 6th power term

$$U(d) = 4\varepsilon \left(\frac{\sigma}{d}\right)^{12}. \tag{3.2}$$

Characteristically of the 12th power potential is that the slope close to σ is very steep and will therefore result in an abrupt interaction force when two particles interact with a reasonably small cut-off radius, cf. Figure 3.2. Therefore, this potential resembles a very hard material.

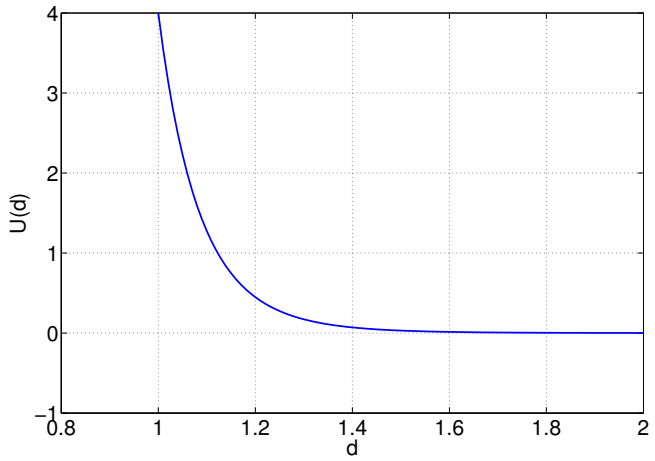


Figure 3.2: Repulsive interaction potential

Alternatively, the original LJ potential remains a valid choice when a softer material is required. The cut-off radius can be selected specifically to still completely eliminate the attracting part of the potential. For that purpose, the portion of the graph with positive slope is cut off and therefore, only repulsive forces can occur. That cut-off radius r_p is calculated by using the derivative of the potential and finding the minimum where the slope turns positive.

$$\begin{aligned} \frac{\partial U(d)}{\partial d} &= 0, \\ \frac{24\varepsilon}{d}\left(2\left(\frac{\sigma}{d}\right)^{12} - \left(\frac{\sigma}{d}\right)^{6}\right) &= 0, \\ 2\left(\frac{\sigma}{d}\right)^{12} - \left(\frac{\sigma}{d}\right)^{6} &= 0, \end{aligned}$$

yielding

$$r_p = 2^{1/6}\sigma.$$

Note that the particle index i is omitted for better readability. Thus, the maximal cut-off radius that should be used depends on σ and should be selected independently for each particle shape to satisfy $\sigma_i \leq r_i^c \leq r_p$. When the cut-off radius is set close to r_p, the interactions do not immediately induce a very large force, but instead introduce a gradually rising force between the particles proportional to their distance d_{ij}. This can be interpreted as accounting for the small deformations that occur in real-life interactions. Thus, the interaction model used here can be understood as a soft repulsive

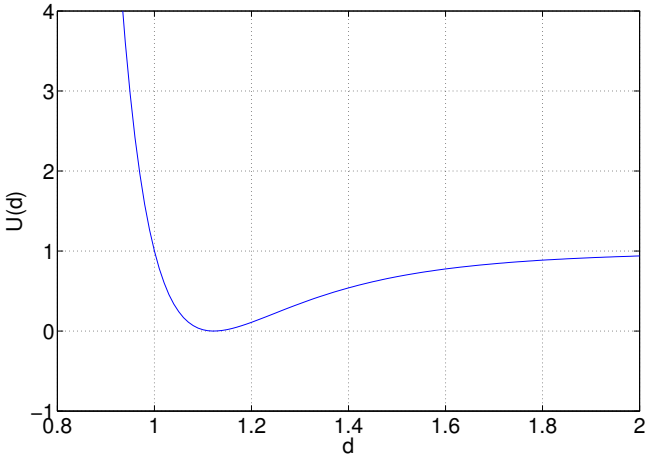


Figure 3.3: Shifted Lennard-Jones potential, with $r_i^c = 1.122\,\sigma$

sphere model. To also ensure a smoother slope in the potential energy, the potential may be shifted to have its root at $d = r_i^c$, cf. Figure 3.3, in the following way

$$U_s(d) = U(d) - U(r_i^c) = 4\varepsilon \left(\left(\frac{\sigma}{d}\right)^{12} - \left(\frac{\sigma}{r_i^c}\right)^{12} - \left(\frac{\sigma}{d}\right)^{6} + \left(\frac{\sigma}{r_i^c}\right)^{6} \right).$$

This shifted potential does not have an impact on the force computation, as the only change to eq. (2.16) is the addition of a constant.

3.2 Initialization

In the scope of this work, the experiment of the dropping sphere was simulated by subjecting the system to gravity. For this purpose, the sphere was positioned inside the simulation volume at the height h_d from which it was dropped later. The particles were placed inside of it in a spherical configuration. This ensured that all particles remained inside the sphere during their initial placement. To describe the position of a point in space, spherical coordinates can be used.

Definition 3.2. *(Spherical coordinates)*
Spherical coordinates use an origin, for instance $\mathcal{O} = (0,0,0)^T$ *and three parameters* $r, \theta \in [0,\pi)$ *and* $\phi \in [0,2\pi)$*, with* $r, \theta, \phi \in \mathbb{R}$*. The first parameter* r *is the radial distance from the origin. The other two are values that characterize the polar angle* θ *and azimuthal angle* ϕ*.*

Given spherical coordinates (r, θ, ϕ), the cartesian coordinates (x, y, z) can be extracted as

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \mathcal{O} + \begin{pmatrix} r\sin\theta\cos\phi \\ r\sin\theta\sin\phi \\ r\cos\theta \end{pmatrix}.$$

To set up the experiment, the first particle was placed at the center of the hollow sphere, at the position $p_1 = (0,0,0)^T$. The other particles were placed in layers around it using spherical coordinates. To achieve this, the radial distance was increased in steps for each layer, starting from $r = 0$,

$$r = r + \alpha \max_i\{r_i^*\},$$

where $\alpha \geq 1 \in \mathbb{R}$ to prevent particle overlaps at the beginning of the experiment. For the increments of the angles, that position layers of particles in circles around the center, it must hold that the distance of the particles should always remain at least $r_c^* = \max_i\{r_i^*\}$. Otherwise, an interaction would occur immediately at the start of the simulation. The polar angle θ starts at 0 and is incremented from there. Consider the triangle (O, A, B) in Fig. 3.4. After the placement of the very first particle at O, r has been incremented and $\theta = 0$ resulted in the placement of the second particle at A. Now, the increment for θ needs to be found to place the third particle at B in a manner that will result in a spherical layer around the particle at O when θ is successively incremented until the layer is complete. Note that only in the case of this first layer, the triangle (O, A, B) is equilateral; the computation is shown for the general case.

To find the increment θ', consider that the side opposite to θ' must also have the length αr_c^*, so that particles at O and B do not overlap. Thus, it is clear that the angular increment also depends on the current layer's radial distance r and can be found using

$$\begin{aligned} \sin\frac{\theta'}{2} &= \frac{\alpha r_c^*}{2r}, \\ \implies \theta' &= 2\arcsin\frac{\alpha r_c^*}{2r}. \end{aligned} \tag{3.3}$$

After each increment of θ, a full circle of particles was laid out in the $(x-y)$-plane about the z axis (by doing a full sweep over increments of ϕ). The radius of this circle can be found as the height h_r in the triangle considered

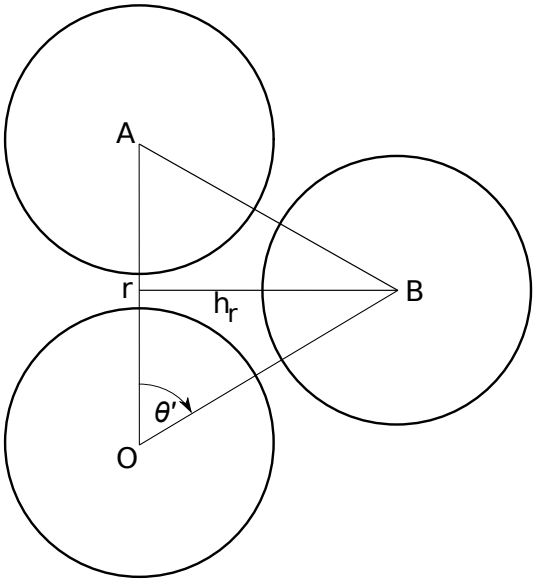


Figure 3.4: Angle increment θ' and radius h_r in the initialization phase for the placement of the first three particles

for the increment of θ. To calculate this height, consider that for the area A of the triangle, the equations

$$A = \frac{1}{2} r h_r, \quad \text{and also}$$
$$A = \frac{1}{2} r^2 \sin\theta,$$

hold. Combining those equations, h_r may be evaluated as

$$h_r = r \sin\theta.$$

This serves as the basis for the computation of the increment for ϕ. Similarly to the derivation of eq. (3.3), with h_r as the hypotenuse,

$$\phi' = 2 \arcsin \frac{\alpha r_c^*}{2 h_r}. \tag{3.4}$$

The angle ϕ is then incremented, starting from 0 to 2π. After the layer of particles on the ϕ circle is completed, θ is incremented again, h_r is re-evaluated and another circle is layered by incrementing ϕ. Once θ reaches π, the radial distance r is increased again and the next layer is applied. This is continued until all particles have been placed. The algorithm requires three nested loops, as described in Listing 3.1.

To compute the total number of particles with cut-off radius r_c^* that fit in a

Listing 3.1: Initial configuration

```
radius=0.0
while (particles need to be placed)
   theta=0.0
   radius=radius+alpha*rc
   while (particles need to be placed)
    phi=0.0
    while (particles need to be placed)
      xi=radius*sin(theta)*cos(phi)
      yi=radius*sin(theta)*sin(phi)
      zi=radius*cos(theta)
      p =(xi,yi,zi) !particle position
      phi=phi+2.0*asin(alpha*rc/(2.0*radius*sin(theta)))
      if (circle is full) exit
    end while
    theta=theta+2.0*asin(alpha*rc/(2.0*radius))
    if (layer is full) exit
   end while
if (no unplaced particle remains) exit
end while
```

hollow sphere with radius R, first the number of particles that can be placed on a circle are calculated, using eq. (3.4)

$$n_{circle} = \left\lfloor \frac{2\pi}{\phi'} \right\rfloor = \left\lfloor \frac{\pi}{\arcsin \frac{\alpha r_c^*}{2h_r}} \right\rfloor .$$

This is repeated for each layer (θ, r), yielding

$$N_{max} = 1 + \sum_{h_r(\theta,r)} \sum_{j=1}^{\lfloor R/\alpha r_c^* \rfloor} \left\lfloor \frac{\pi}{\arcsin \frac{j\alpha r_c^*}{2h_r}} \right\rfloor . \tag{3.5}$$

With the particles in place, the numerical experiment can be described.

Definition 3.3. *(Drop experiment II)*
The experiment commences by placing the hollow sphere at a certain height h_d. The particles are placed inside in a spherical configuration as described above. When the simulation is started, the particles are released first while the hollow sphere remains fixed in space, cf. Figure 3.5. When the particles have come to a steady state at the bottom of the cavity, due to gravitation, the sphere is released and the experimental phase commences. The system

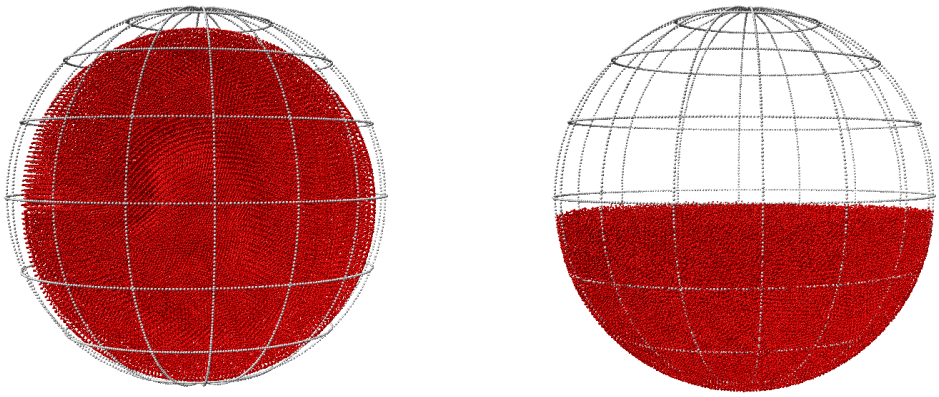


Figure 3.5: Initial placement of 200000 particles inside a hollow sphere and their position after the initial settlement phase before the drop experiment starts

is simulated through multiple impacts with the fundament. These impacts are recorded and the rebound time ΔT is calculated.

The initial settlement phase guarantees the comparability of the results. In a simulation with different particles, the particles in the initial placement are assigned their shape at random to have a heterogeneous mix. This introduces a stochastic component into the experiment. To prevent the dilution of the numerical results, it is possible to extract the particle positions at any point in the simulation, i.e. after the initial phase, and to enter such a configuration into a new simulation with different parameters. The potential between the hollow sphere and the particle mass may cause a minute initial delay in the free fall of the particles. To prevent this behavior, it is advisable to set the particle velocities to the sphere velocity during the beginning of the experiment.

3.3 Boundary conditions

To account for the reflections of the particles from the sphere during the experiment, the boundary has to be modeled differently than in straightforward MD. Motivated by the underlying application, reflective boundary conditions were used. There are several ways to model reflections and a decision has to be made for either a discrete or continuous approach.

The discrete route may use the actual geometric trajectories. Each par-

ticle's trajectory is identified by its position p_i and velocity v_i. When the particle is propagated in the next time step to p_i^{new}, a breach of the sphere boundary with the center at the position c_s^{new} can be verified by computing

$$d_{is} = \|p_i^{new} - c_s^{new}\|. \tag{3.6}$$

Comparing the distance d_{is} to the radius R of the sphere yields the relative position of the particle in the sphere. If the particle is outside of the sphere, it is clear that the previous time step has to be examined more carefully and that a reflection has to take place. This can be computed exactly by using geometric relations.

Theorem 3.1. *(Reflection of a particle in the sphere)*
A particle with new position p_i^{new} that does not satisfy $d_{is} \leq R$ must be reflected by the sphere hull. Given the particle's previous position p_i and velocity v_i and the sphere's position c_s, the reflected position can by computed using the tangential plane in the point of impact.

Proof. First, let the particle be placed at the position p_i of the preceding time step. Then, the distance to the boundary can be derived by solving

$$\|(p_i + tv_i) - c_s\| = r_s,$$

for t. The intersection point with the sphere is

$$p_{temp} = p_i + tv_i. \tag{3.7}$$

Next, the reflection angle β has to be considered. The reflection occurs on the tangential plane at p_{temp}. The unit normal vector of that plane is

$$n_i = (p_{temp} - c_s) / \sqrt{\langle p_{temp} - c_s, p_{temp} - c_s \rangle}. \tag{3.8}$$

Now, the velocity vector v_i needs to be manipulated according to the reflection angle. This can be done without evaluating a rotation matrix by using the rotation formula of Rodrigues [3]

$$v_i' = v_i - 2\langle v_i, n_i \rangle n_i. \tag{3.9}$$

This is equivalent to a rotation of $-v_i$ around n_i by π, cf. Figure 3.6. Now, the particle can be advanced by considering the reflection, advancing the remainder of the time step from p_{temp} using the new velocity v_i'. □

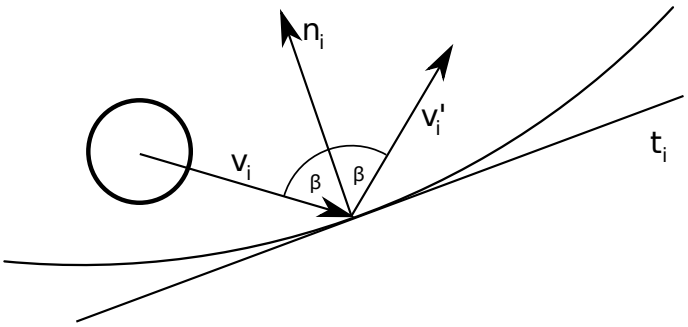


Figure 3.6: Example of a particle-sphere interaction with pre- and post-collision velocity vectors v_i and v_i' and tangential impact plane t_i with normal vector n_i

During the collision, energy will be dissipated. If the coefficient of restitution between particle and sphere material is known, velocities of the collision partners after the impact can be obtained using the principle of conservation of momentum

$$m_s v_s + m_p v_p = m_s \bar{v}_s + m_p \bar{v}_p, \tag{3.10}$$

and the definition of the coefficient of restitution

$$\varepsilon_r = \frac{\bar{v}_p - \bar{v}_s}{v_s - v_p}. \tag{3.11}$$

The velocity $\bar{v}_p$ of the particle after the interaction can be obtained by solving eq. (3.11) for $\bar{v}_s$

$$\bar{v}_s = \bar{v}_p - \varepsilon_r (v_s - v_p),$$

and substituting the result in eq. (3.10). Solving for the particle's post-impact velocity results in

$$\bar{v}_p = \frac{m_s v_s + m_p v_p - m_s \varepsilon_r (v_s - v_p)}{m_s + m_p}.$$

Similarly, $\bar{v}_s$ is obtained.

However, a number of problems arises when implementing this approach on a computer. The inexact nature of the computation will result in numerical errors and eventually in the failure to correctly predict the particle's position. For very small reflection angles, small errors in the computation can have a great effect. For particles sliding along the hull of the sphere, several reflections could take place during a single time step and it would be

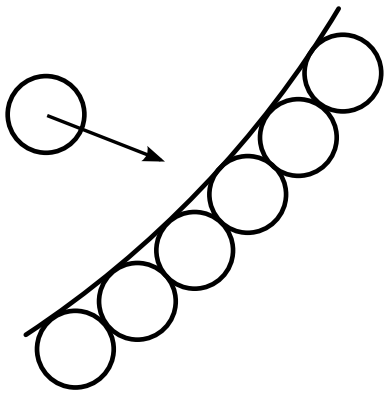


Figure 3.7: Particle-sphere interaction model using phantom particles at the boundary layer

computationally too expensive to implement multiple checks for that case. This will eventually result in particles exiting the sphere. Annoyingly, this sometimes occurred after many hours of computation. To eliminate this problem, smaller time steps could be chosen, however, this quickly leads to unacceptable runtimes. In fact, the simulation time may grow from a few hours to several weeks. Therefore, this approach is not feasible for the simulation of the systems considered in this thesis, as confirmed by corresponding computations.

Instead, a continuous, potential based approach was used in the present work. One possibility to achieve this is to use phantom particles positioned around the sphere, cf. Figure 3.7. However, this method proved to be infeasible. The phantom particles need to be small enough to approximate the hull reasonably well but with those particles that are around the same size as the powder inside, problems may arise during the simulation. A very small time step, smaller than 1/100th of the time step used with the approach discussed below, had to be selected or the particles managed to breach the barrier. The most serious problem, however, was the aggregation of particles that posed a threat to the integrity of the barrier. With a larger heap, as particles start to accumulate at the bottom of the sphere, those at the bottom of the pile are pushed against the barrier by gravity acting on the other particles above, and are eventually forced outside of the sphere as the potential that is originating from the heap overcomes the potential around the phantom layer of the sphere.

To circumvent these problems, the reflection was enforced by using a barrier potential function. This could also be interpreted as barrier particles with an infinitely large diameter.

Using the same concept as in the particle interaction, the repulsing force increases as the distance between the sphere and particle grows closer. Contrary to the particle potential, however, it continues to increase and act in the same direction outside the sphere. Particles that manage to exit the sphere due to the heap's push would still be subjected to that potential and pushed back inside. This allows the simulation to have a much greater flexibility in the time step size. A potential that can be used for this purpose is of the form

$$U_b(d_i) = 0.5\beta \exp(d_i^2) - 1. \tag{3.12}$$

The resulting force is

$$F(d_i) = -\beta d_i \exp(d_i^2), \tag{3.13}$$

where $\beta \in \mathbb{R}$ controls the magnitude of the potential and was set to have the same magnitude as the gravitational potential to allow for deflection. This empowers the barrier to overcome the gravitational potential and decelerate the particles. The vector d_i determines the direction of the rejecting force and is based on the position of particle i relative to the center of the sphere

$$d_i = \frac{c_s - p_i}{\|c_s - p_i\|}.$$

Similarly to the interaction of particles, the particle's specific cut-off radius r_i^* was also used for these particle-sphere interactions, only inducing a force close to the boundary. Particles further inward were not considered as interaction partners. As an additional benefit, the resulting method is fully based on a unified approach for all particle interactions. This keeps the method easily modifiable, allowing to substitute other potentials for the interactions, if required.

During the experiment, the sphere is subjected to exterior forces, i.e., the impact at the fundament. This will cause the hull to deform. Additionally, it should later be able to transmit vibrations that occur in the composite material to the particles. Two kinds of deformation were implemented in this work. The first approach is aimed at modeling the movement of a sphere trapped inside a larger body, such as a casing that is excited to vibrate, e.g., by the operation of a machine. In that case, the whole sphere will be displaced relative to the particles. It would be plausible to incorporate

the movement of the center of the sphere into the model, however, using a pulsing hollow sphere to transmit vibration to the particles by varying the radius of the sphere by a small amount s_v will yield comparable results and shows better performance.

Definition 3.4. *(Pulsing radius)*
Let the sphere have a radius R and the simulation is performed with the time step Δt. The pulsing vibration mode is given as

$$r(t) = (1 - s_v)R + s_v \sin(\frac{t}{\Delta t}).$$

The second approach takes into account the way the boundary deforms upon impact and introduces an ellipsoidal deformation, thus, this model more closely resembles the behavior in the drop experiment. When the sphere interacts with the fundament, it will suffer a compression along the vertical axis. For this work, it was assumed that the deformation of the sphere is uniform, i.e., the sphere expands along the sides and flexes into an ellipsoid.

Definition 3.5. *(Ellipsoid)*
The ellipsoid is defined by its half-axes e_1, e_2, and $e_3 \in \mathbb{R}$ and is parametrized by

$$\frac{x^2}{e_1^2} + \frac{y^2}{e_2^2} + \frac{z^2}{e_3^2} = 1,$$

with $x, y, z \in \mathbb{R}$.

Note that the sphere is also covered by this definition when all half-axes are equal. Because of the assumption above, during deformation, $e_1 = e_2$ holds, which is called an oblate ellipsoid. After the deflection of this ellipsoid, it can be suggested that the shape of the hollow sphere starts to oscillate between the ellipsoidal and spherical form. During this oscillation, the volume must remain constant. The volume of an ellipsoid is

$$V_{ell} = \frac{4\pi e_1 e_2 e_3}{3}, \tag{3.14}$$

thus, the product of the half-axes always has to remain constant during oscillation. Similar to the pulsing radius, a sine-based vibration can be introduced on the half-axis e_3 and the other half-axes may then be obtained with (3.14).

For the ellipsoidal deformation, the previously presented discrete reflective computation for the spherical case no longer holds because the radius of a sphere cannot be used for the ellipsoid. Therefore, a more complex derivation is necessary.

Proposition 3.1. *(Ellipsoid-particle deflection)*
Given an ellipsoid Σ *and a particle trajectory* Γ

$$\Sigma : \frac{x^2 + y^2}{e_1^2} + \frac{z^2}{e_3^2} = 1, \tag{3.15}$$

$$\Gamma : p_i + gv_i = (x, y, z)^T, \tag{3.16}$$

the intersection of Σ *and* Γ *is*

$$\begin{aligned} p^* &= p_i + \tau v_i, \text{ with} \\ \tau_{1/2} &= \frac{-b + \sqrt{b^2 - 4ac}}{2a}, \text{ where} \\ a &= \left\langle v_i, \left(v_{i_x}, v_{i_y}, v_{i_z} \frac{e_1^2}{e_3^2} \right) \right\rangle, \quad b = \left\langle 2p_i, \left(v_{i_x}, v_{i_y}, v_{i_z} \frac{e_1^2}{e_3^2} \right) \right\rangle, \text{ and} \\ c &= p_{i_x}^2 + p_{i_y}^2 + p_{i_z}^2 \frac{e_1^2}{e_3^2} - e_1^2. \end{aligned}$$

Proof. The problem of finding the intersection of the ellipsoid and the trajectory of the particle i is to find τ where

$$\begin{aligned} \frac{x^2 + y^2}{e_1^2} + \frac{z^2}{e_3^2} = 1, \text{ so that} \\ (x, y, z)^T = p_i + \tau v_i, \end{aligned}$$

holds. Note that for readability, only the case where the ellipsoid is centered in the origin is described here, for the general case, the ellipsoid equation would be expanded to

$$\frac{(x - m_x)^2 + (y - m_y)^2}{e_1^2} + \frac{(z - m_z)^2}{e_3^2} = 1,$$

where (m_x, x_y, m_z) is the midpoint of the ellipsoid. Substituting eq. (3.16) in

eq. (3.15) results in a quadratic equation for τ

$$\tau^2 \left\langle v_i, \left(v_{i_x}, v_{i_y}, v_{i_z} \frac{e_1^2}{e_3^2} \right) \right\rangle + \tau \left\langle 2p_i, \left(v_{i_x}, v_{i_y}, v_{i_z} \frac{e_1^2}{e_3^2} \right) \right\rangle + \\ + p_{i_x}^2 + p_{i_y}^2 + p_{i_z}^2 \frac{e_1^2}{e_3^2} - e_1^2 = 0. \tag{3.17}$$

The solution is

$$\tau_{1/2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}, \quad \text{where}$$
$$a = \left\langle v_i, \left(v_{i_x}, v_{i_y}, v_{i_z} \frac{e_1^2}{e_3^2} \right) \right\rangle, \quad b = \left\langle 2p_i, \left(v_{i_x}, v_{i_y}, v_{i_z} \frac{e_1^2}{e_3^2} \right) \right\rangle, \text{ and}$$
$$c = p_{i_x}^2 + p_{i_y}^2 + p_{i_z}^2 \frac{e_1^2}{e_3^2} - e_1^2.$$

The negative solution for τ can be discarded, because of the orientation of the velocity vector that is always pointing towards the boundary of the sphere. Therefore, τ will always be a positive fraction of the time step Δt and thus

$$\tau = \frac{-b + \sqrt{b^2 - 4ac}}{2a}.$$

□

For the duration τ, the particle i propagates with velocity v_i before reaching the impact point on the hull. After that, it is reflected and propagates the remainder of the time step $\Delta t - \tau$ with the new, post-impact velocity. To derive it, the normal vector of the tangential plane at the intersection point has to be computed and then, similarly to the spherical case, the velocity vector v_i is reflected to v_i'. The normal vector in any point on the surface of the ellipsoid is obtained using the gradient of the ellipsoid surface which results in

$$n = \begin{pmatrix} 2x/e_1^2 \\ 2y/e_1^2 \\ 2z/e_3^2 \end{pmatrix}, \tag{3.18}$$

for any point (x, y, z) that lies on the surface. The normal unit vector after

normalization is

$$\hat{n} = \frac{n}{\|n\|}.$$

With this normal vector, the reflected velocity vector is computed similarly as in eq. (3.9). Consequently, the new position of the particle after the reflection is

$$p_i^{new} = p_i^{old} + \tau v_i + (\Delta t - \tau) v_i'.$$

This approach can also be used in the case of a non-deforming sphere by additionally setting $e_1 = e_2 = e_3$. Just like in the case of the sphere discussed above, this discrete computation of the trajectory of a reflected particle carries the risk of miscalculations due to numerical errors or when the impact angle on the surface is very small with the possibility of multiple impacts within one time step Δt. Additionally, solving the quadratic equation (3.17) for every particle that comes in contact requires an increased computational effort during phases where the bulk of particles inside move against the sphere during the experiment, greatly increasing simulation time. For the continuous potential-based approach, the particle-sphere interaction involves the problem of finding the closest distance of a point to an ellipsoid surface for the potential evaluation. The result is a minimization problem, further introducing numerical effort. Therefore, the pulsing radius deformation was primarily used in this work to observe system behavior under external excitation.

Modeling of contact with the fundament

For the interaction of the hollow sphere with the fundament that it is being dropped upon, a strictly mechanical approach of performing an inelastic collision could be taken. The coefficient of restitution is required for this computation. It can be evaluated by performing the drop experiment with an empty sphere and measuring the height that the sphere reaches after rebounding. When dropping from height h_d and measuring the rebound height h, the coefficient of restitution ε_r is given by [11]

$$\varepsilon_r = \sqrt{\frac{h}{h_d}}. \tag{3.19}$$

Alternatively, in the one-dimensional case, it can be derived from the velocities v_A and v_B of two bodies A and B before the impact and the velocities u_A and u_B after the impact by

$$\varepsilon_r = \frac{u_A - u_B}{v_A - v_B}. \tag{3.20}$$

In the three-dimensional simulation, this holds for the velocity component perpendicular to the impact surface. In the considered scenario, that surface is the $(x-z)$ plane and therefore, the y component of the sphere velocity is affected. Since the surface is supposed to be the fundament and the motion of the sphere is relative to it, the velocity of the surface is always zero. This results in

$$v_y^{new} = -\varepsilon_r v_y^{old}.$$

Because the mass and velocity of the sphere are known at every point of time in the simulation, the new velocity after the impact with the fundament is immediately computable if the coefficient of restitution ε_r is known. If it is possible to derive the coefficient from a physical experiment, this is a clear advantage of this reflection approach, as the collision can then be performed exactly.

However, since the particle interaction is based on a potential and a discrete mechanical approach for the fundament interaction would disrupt the otherwise continuous potential-based formulation of the model, it is favorable to simulate the whole system using potentials. Therefore, the fundament contact is modeled similarly to the contact of particles with the sphere with a potential U_s, using the barrier force (3.13) and a cut-off radius r_c for the interaction. Instead of d_i, the distance of the sphere to the fundament is used after taking into account r_c

$$d_s := c_s - R - r_c.$$

If $\langle d_s, d_s \rangle < r_c^2$, an interaction takes place, a rejecting force is induced and the sphere is reflected back up from the surface. A great advantage of the continuous potential approach is that relatively large time steps can be chosen as the exponential potential will ensure that all particles are reflected and can not exit the simulation. Note that the model now uses potentials for every possible interaction; particle-particle, sphere-particle and sphere-fundament. Thus, the method is homogenized by using potentials exclu-

sively. The potentials are easily interchangeable and can be adjusted to different requirements, e.g., for soft or hard interactions. A continuous approach for all interactions simplifies event handling overall, since a discrete approach would lead to a high numerical effort to always identify the next interaction to propagate the system to that point in time. The potentials allow for overlaps to occur and still capture the physical interaction, under the condition that the specified time step is small enough.

3.4 Friction

Friction is problematic in many industrial applications. However, it is desired in the particular scenario considered, because it is the driving force behind the damping effect studied here and needs to be taken into account. Many models for solid body interaction depend on rather elaborate computations. One example is using a soft particle model [28]. The interaction forces for the normal and tangential direction are evaluated separately. The normal forces are based on the Hertz-Kuwabara-Kono (HKK) model and the tangential force on Coulomb's law of friction. The HKK model takes into account specific material parameters, such as Young's modulus E and Poisson's ratio ν

$$F_n = -k_n \alpha^{3/2} - \gamma_n v_n \sqrt{\alpha}, \tag{3.21}$$

with normal stiffness $k_n = \frac{2}{3} E \sqrt{R/2}(1-\nu^2)^{-1}$ and R^{-1} the sum of the inverse radii of the interaction partners. The other parameters are the overlap α, the normal damping coefficient γ_n, and the relative normal velocity v_n of the two spheres. For the tangential force, the authors [28] use

$$F_s = \min(|\gamma_s v_s \sqrt{\alpha}|, |\mu_d F_n|)\text{sign}(v_s).$$

Here, the material parameter γ_s represents the shear damping coefficient, μ_d the dynamic friction coefficient, and v_s the relative tangential velocity.

Models such as this one are able to match real world behavior very well. However, due to their complexity, in a simulation that has to consider ~10^5 interacting bodies, these models are not applicable due to the computational effort required. Furthermore, the material inside the sphere changes during the sintering process and the material parameters are altered, cf. Figure 3.8.

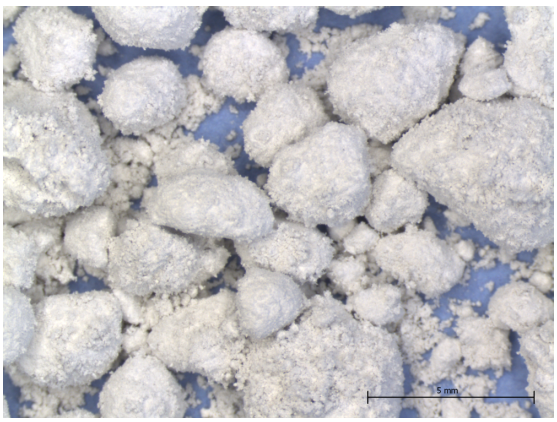


Figure 3.8: Clumping of particles after sintering, ©Fraunhofer IFAM Dresden

Therefore, a much more efficient and simple approach was required in the course of this work. Whenever an interaction occurs, each particle that takes part is marked. During the update of the velocities in the simulation, this information is recovered and results in a minimal reduction of the kinetic energy. The reduction is computed as

$$v_i^{new} = (1 - n_{int})(v_i^{old} + \frac{\Delta t}{2m_i} F_i). \tag{3.22}$$

If more than one interaction occurs in a time step, i.e. with a particle and the sphere, the factor n_{int} may be incremented for each interaction. This reduction can be calibrated to real world experimental data. At the beginning of the next time step, the factor n_{int} is reset to zero. The same concept is used to reflect friction on the rotational movement and to model friction of the hollow sphere with the fundament.

Chapter 4

Simulation of the physical experiment

The efficiency of the damping effect of the particles can be observed using physical experiments. A simulation software to match those findings on a computer was developed based on the model discussed in the previous chapter. It allows to perform many variations of the experiment without the need to actually manufacture the sphere.

The ultimate goal of this work was to derive the coefficient of restitution of the simulated system. This value directly quantizes the amount of damping that can be achieved by the addition of particles when compared to the experiment without particles.

4.1 Coefficient of restitution

In the physical experiments at Fraunhofer IFAM Dresden, the rebound time ΔT is measured using a microphone that records the impacts on a plate [21]. Thus, computing this time via simulation is the most desirable result, because it will directly yield the coefficient of restitution and enables a direct comparison to the physical experiment. To use ΔT as a measure of damping, its relation to the coefficient of restitution must be explored.

According to Ref. [21], the coefficient of restitution (COR) ε_r in an experiment with drop height h_d can be derived from the vertical throw upwards

$$y(t) = v_b t - \frac{g}{2} t^2, \tag{4.1}$$

with the standard gravity $g = 9.81\ \mathrm{ms}^{-2}$ and starting velocity v_b, in this case, the velocity immediately after the bounce with the fundament. Coupled with

the definition of the coefficient of restitution [11],

$$\varepsilon_r = -\frac{v_b}{v_0}, \tag{4.2}$$

where v_0 is the velocity the sphere reaches before the bounce after the drop from height h_d, the COR can be derived from the rebound time. First, to obtain v_0, the drop time t_* from h_d to the fundament is calculated

$$y(t_*) = h_d - \frac{g}{2}t_*^2 = 0$$
$$t_* = \sqrt{\frac{2h_d}{g}}.$$

The velocity at the impact is then

$$v_0 = \dot{y}(t_*) = -gt_* = -\sqrt{2h_d g}$$

To obtain the velocity v_b after the bounce, the measured time ΔT is used. This is the time after which the height y according to eq. (4.1) reaches 0 again after the first deflection, indicating the second bounce on the fundament. Thus, the velocity of the sphere after the first impact can be evaluated by

$$0 = t(v_b - \frac{g}{2}t),$$
$$\implies v_b = \Delta T\frac{g}{2}.$$

Substituting the results for v_0 and v_b in eq. (4.2) yields

$$\varepsilon_r = \sqrt{\frac{g}{8h_d}\Delta T^2}.$$

4.2 Straightforward algorithm

The *naive* straightforward algorithm must check each particle for potential interactions with every other particle. Naturally, this leads to a double loop over the particles in the implementation and results in quadratic complexity in terms of the particle number N. Algorithm 4.1 shows that basic algorithm in pseudo code. Consequently, the particle number has a big direct impact on the computational effort. However, this double loop is not necessary for the interaction computation as the particles are not distributed uniformly in

Listing 4.1: Quadratic algorithm

```
!import simulation parameters
!initialize simulation
for integration=1: intsteps
  for i=1:N
    update v_s, v(i) = v(i) + 0.5Δt(F(i)/m(i) − g)
    update p_s, p(i) = p(i) + Δtv(i)
    update ω(i), o(i)
  end for
  for j=1:N
    if ‖p(i) − p(j)‖ < r_cut then
      compute F_ij using potential U
      update F(i), F(j)
    end if
  end for
  for i=1:N
    update v(i) = v(i) + 0.5Δt(F(i)/m(i) − g)
  end for
  compute energy
end for
! return results: simulation data, impact times
```

the simulation volume and the actual number of interactions is quite small, so that steps can be undertaken to reduce the effort spent on the interaction calculation. Selecting a suitable cut-off radius, where only the immediate vicinity of the particle is considered, is a first step in reducing the number of possible interactions dramatically. However, while the simulation of moderately sized systems with spatial equally distributed particles benefits from this, for the present application, it is inadequate. In addition to the high number of particles, the number of interactions increases greatly, even with a small cut-off radius, when the simulation reaches the point where particles start to accumulate at the bottom of the sphere. In that case, a dramatic increase of the computational effort is observed. Thus, measures to handle the interactions more effectively must be taken.

4.3 Linked cell method

To lower the complexity of the method, a way to significantly reduce the number of potential interaction partners has to be found. This can be

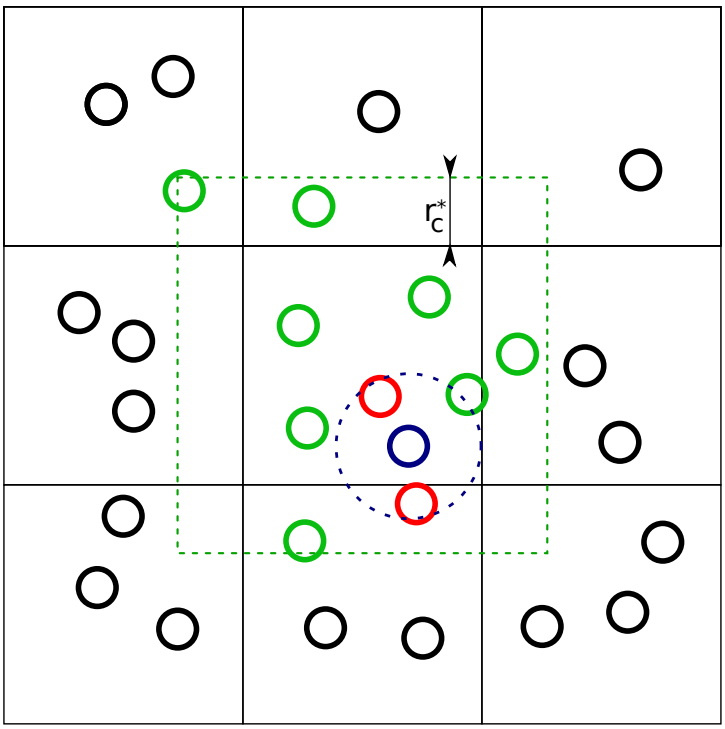


Figure 4.1: Linked structure with overlap r_c^*; the potential interaction partners of the blue particle within the cut-off radius r_c^* are shown in red

achieved by dividing the simulation volume into small cubic cells. For every particle in such a cell, the other particles in that cube have to be considered as potential interactions. However, for particles close to the boundary of the cell, the neighboring cells have to be considered as well, so there is an interaction overlap around each individual cube, linking the cells. Information about particles in that domain has to be exchanged with neighboring cells. The optimal cell size may depend on the cut-off radius of the largest particle in the simulation, r_c^* [14]. This ensures that all physically possible interactions will only occur among particles in immediately neighboring cells, cf. Figure 4.1. Thus, only a very limited number of neighboring cells are considered, instead of the whole simulation volume. For a simulation volume with length $L = (l_x, l_y, l_z)^T$, the resulting number of cells in each direction is

$$N^* = (N_x^*, N_y^*, N_z^*)^T = \lceil L/r_c^* \rceil,$$

where $\lceil x \rceil$ of a floating point number x is the smallest integer number larger or equal to x. In each iteration, the particles can be distributed with little numerical effort among the array of cells using the coordinates for the i-th particle $p_i = (p_x, p_y, p_z)_i$. Each cell is identified by three integer values

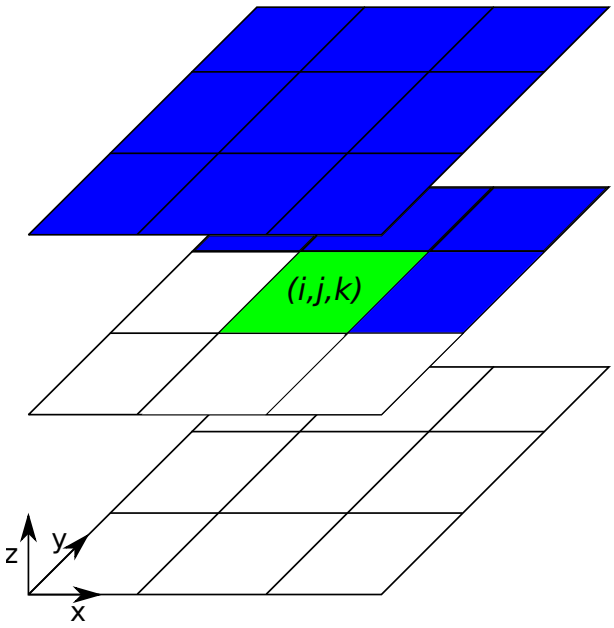


Figure 4.2: Cell neighbors of the cell (i, j, k) that are considered by the algorithm (shaded in blue)

$(i_c, j_c, k_c) := (i, j, k)_c$ and the cell index for a particle is calculated by

$$(i, j, k)_c = \lfloor (N_x^* p_x / l_x, N_y^* p_y / l_y, N_z^* p_z / l_z) \rfloor .$$

The amount of work that has to be done in each iteration during the simulation run can be greatly reduced by choosing a traversing order of the cells that only computes interactions with the cells that have not yet been traversed. This prevents the need to check for the double counting of interactions of particles in neighboring cells (particle i in cell a with particle j in cell b and vice versa). In this work, the traversing is started at one of the corners of the simulation region, i.e., the cell that contains the origin. Only cells in positive x and y direction contain valid interaction partners. Additionally, the next layer in positive z direction is also permitted to contain interaction partners. This method results in a maximum of 13 neighboring cells that have to be considered for every linked cell, cf. Figure 4.2. Fewer neighboring cells occur at the edges and corners of the simulation volume. The algorithm is sketched in Listing 4.2. The distribution of particles among the cells leads to an improvement in algorithmic efficiency. In fact, the linked cell approach is characterized by a linear complexity in terms of the particle number as opposed to the quadratic complexity of the first approach [14]. In standard MD applications, the equally distributed particles allow for an efficient operation of the linked cell (LC) algorithm

Listing 4.2: Cell traversing algorithm

```
for k=2,zcells-1 !cells in z direction
  for j=2,ycells-1 !cells in y direction
    for i=2,xcells-1 !cells in x direction
      !set current cell index as (i,j,k)
      !now cascade through neighbor cells
      !with indices (in,jn,kn) and calculate interactions

      !first on the same z plane
      kn=k
      for jn={j,j+1}
        for in={i,i+1}
          !calculate interactions
          !between particles in (i,j,k), (in,jn,kn)
        end for
      end for
      jn=j+1
      in=i-1
      !calculate interactions
      !between particles in (i,j,k), (in,jn,kn)

      !now move onto the cells in the next z layer
      kn=k+1
      for jn={j-1,j,j+1}
        for in={i-1,i,i+1}
           !calculate interactions
           !between particles in (i,j,k), (in,jn,kn)
        end for
      end for

    end for
  end for
end for
```

N	LC	quadratic
625	27m 36s	5 min 53s
1250	38min 21s	15min 13s
2500	74min 32s	44min 30s
5000	n/a	145 min 17s

Table 4.1: Run times for a simulation over 70000 time steps for a linked cell algorithm with equal memory allocation to all cells compared to a standard algorithm

with a basic memory allocation system that uses the same amount of memory for each cell, avoiding the need for any runtime memory management.

However, in the experiment that was considered here, a large amount of particles congregate in a small volume at the bottom due to the gravitational force. Thus, the cells that are at the top of the sphere are temporarily basically empty. Over the course of the simulation, the load balance of the cells regarding their memory requirement is varying strongly. An example of this is shown in Table 4.1, where the drop experiment was simulated with a commonly used linked cell algorithm using equally distributed memory and compared to the required computation time of the standard algorithm as in Listing 4.1. Note that the overhead associated with the memory allocation leads to higher run times for the LC method. Furthermore, the simulation even breaks down for as few as 5000 particles because the accumulation of particles in a few cells and the resulting memory requirement exhausts the available memory. In order to use the linked cell algorithm for the application of the particle filled sphere, it has to be able to cope with very uneven particle distributions, e.g., by adaptively managing memory allocation for each individual cell.

4.4 Adapted linked cell method

Memory system

In this work, a method with a highly variable memory structure was implemented that is able to deal with uneven particle distributions. It uses a pointer administrated linked list structure. For easy access, every particle is represented by a identification number (ID). These IDs are organized in lists associated with each individual linked cell. The first list element (cf. Listing 4.3) in a cell is attached to a cell header and the remaining list elements

Listing 4.3: List element representing a particle

```
type listelement
  integer :: id=0 !particle ID number
  type(listelement), pointer :: next => null()
  !pointer to the next particle in the cell
end type
```

are connected by pointers, cf. Figure 4.3. During each iteration, the cells are traversed and the list for each individual cell is accessed successively to obtain the potential interaction partners of the particles in that cell. The neighboring cells with their particle lists can also easily be retrieved using the pointers attached to the cell header.

As the simulation evolves, the particles move across cell borders and need to be reattached to another cell's element list. The lists are reordered in each iteration and are organized in a last-in-first-out order, i.e., new elements are always added as the new head element. To avoid repeated deallocation of the list elements, an additional index vector provides a permanent pointer to each element. This allows for an access to each particle's list element without destroying it, i.e., reallocating the memory structure in every time step. Additionally, the problem of dangling pointers is avoided as no element is deallocated until the termination of the program and continuous access is always provided by the index vector. The pointers between

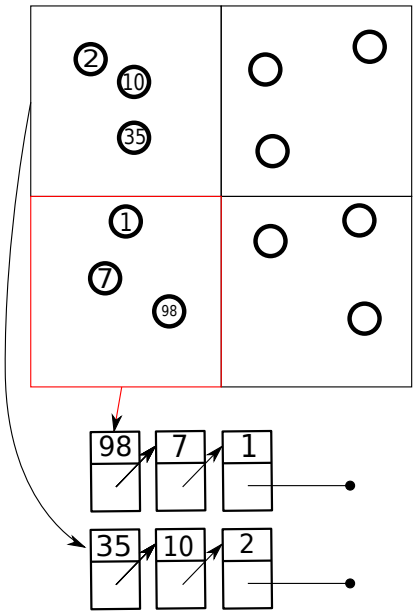


Figure 4.3: Linked cell list memory structure

Listing 4.4: Linked cell algorithm

```
!import simulation parameters
!initialize simulation and cell lists
!store particles in cells using pointer linked lists
for integration= 1:IntSteps
  for i=1:N
    update v(i), v_s(i), p(i), p_s(i)
    update ω(i), o(i)
    update linked cell lists
  end for
  for every cell !cascade through all neighbor cells
    if ||p(i) − p(j)|| < r_cut then
      compute F_ij using potential U
      update F(i), F(j)
    end if
   end for
  for i=1:N
    update v(i) = v(i) + 0.5Δt(F(i)/m(i) − g)
  end for
  compute energy
end for
! return results: simulation data, impact times
```

the IDs that make up a cell list can easily be reconfigured by using the index vector, therefore avoiding the loss of any elements in memory. The cells themselves are set to be fixed relative to the sphere position and are defined at the beginning of the simulation run, so cell borders are only evaluated once.

Instead of using an index vector that points to elements representing particles, an approach was also considered that omits the extra elements used in Listing 4.3 and reconfigures the pointers inside the index vector itself. However, this led to higher runtimes. An explanation for this behavior could be that the individual elements perform better because they fit in a faster cache due to being few in number per cell, whereas using pointers in the larger index vector requires using slower high-level memory that is large enough to store the complete vector. E.g., in a test on a machine with an Intel Core2 Duo 8500 Processor, the vector-based simulation took around 6.6% longer than the implementation that uses additional single element memory blocks (450 h vs. 422 h for 150000 time steps with 35000 particles). Consequently, the linked cell memory structure is associated with

an additional effort compared to the straightforward quadratic approach. The distribution and pointer reconfiguration needs to be done in each iteration as particle positions change over time and they enter and leave cells. However, as the number of particles increases to tens of thousands, the advantage of the linear complexity compared to the standard approach's quadratic complexity far outweighs that overhead. Chapter 5 elaborates on that. The basic LC algorithm is shown in Listing 4.4.

Cell size

Due to geometric considerations, a computationally useful cell size for the linked cell algorithm is equal to the cut-off radius specified for the simulation. Note that reducing the cell size past the cut-off radius r_c^* is not advisable, because interactions beyond the immediate neighboring cells may occur that would not be captured. Thus, the algorithm would have to consider additional cells beyond the immediate neighbors, while choosing larger cells adds additional potential interaction partners to be evaluated with respect to their distance. In the latter case, more interaction checks are performed even though the partners are further apart than r_c^*. When several different particle sizes are used in a simulation, it is obvious that the largest particle's cut-off radius dictates the cell size to be used.

The following experiment compares the simulation of the same system using r_c^* and $2r_c^*$. It can be expected that increasing the cell size will also increase the runtime. Unsurprisingly, the results in Table 4.2 show that the

	runtime in minutes			runtime in minutes	
N	$l = 2r_c^*$	$l = r_c^*$	N	$l = 2r_c^*$	$l = r_c^*$
10000	21.59	19.69	120000	458.35	269.61
20000	48.37	36.81	130000	488.88	286.39
30000	81.12	62.80	140000	539.87	315.76
40000	116.68	77.55	150000	590.13	328.12
50000	150.08	104.57	160000	658.37	363.60
60000	188.87	127.80	170000	667.87	393.15
70000	228.14	135.51	180000	726.60	423.22
80000	269.47	171.09	190000	794.99	448.03
90000	310.88	181.57	200000	843.60	479.25
100000	364.47	216.38	210000	889.04	483.78
110000	396.21	244.77	220000	951.34	523.80

Table 4.2: Computational runtime for the linked cell method with optimal and double cell size

simulation is much faster with the cell size set to r_c^*. While doubling the cell size reduces the pointer administrative overhead, the additional effort for computing the interactions in each cell significantly outweighs that effect.

For very small particles, the optimal length can quickly result in thousands of cells in each direction, leading to an excessive memory requirement, which is in fact limited by the available system memory. For such simulations, a larger cell size must be selected. In addition to the cells covering the sphere, two more layers of cells are added around it. This padding is necessary because of the potential based particle-sphere interaction, due to which particles may momentarily exit the sphere during one iteration only to be propelled back inside in the next time step by the barrier potential (cf. Chapter 3). The padding layer ensures that particles that are interacting with the barrier potential and have momentarily breached the sphere are captured in a cell and can be handled. Thus, at least one layer of cells must be outside of the sphere itself. Still, the time step has to be selected carefully or particles may move beyond that outer layer. This problem may also occur when a particle is subjected to many interactions at once, resulting in an increased accumulated repelling force that results in a significant change in velocity, emphasizing the importance of the selection of a stable time step. Additionally, a second padding cell layer allows to omit special treatment of the border cells in the cell traversing phase of the algorithm. The sphere is embedded in a cuboid volume divided by cells. Thus, many cells are outside of the sphere and add to the memory footprint even though they do not contain any particles. However, as an advantage, this prevents having to deal with a special treatment of cells that are on the very edge of the simulation volume. Having a cuboid volume around the sphere allows for easy traversing of the linked cells and reduces the computational effort.

Proposition 4.1. *(Memory requirement for the cells)*
The maximum memory required for the linked cell structure with N cells in x, y, and z direction, respectively, is $N^3 \cdot 12$ Bytes.

Proof. Each linked cell requires a pointer to an element, which is a custom variable requiring 8 Bytes for the pointer and 4 Bytes for the integer on a 64 Bit system. In the chosen programming language Fortran, the pointer has to be of the same type as the variable that it is associated with so every cell pointer, when allocated, requires 12 Bytes. This leads to a memory

requirement for the cell pointer structure of

$$M_r = N^3 \cdot 12 \text{ Bytes.}$$

□

E.g., when considering an experimental setup typical for this work, where the diameter of the sphere is 3 mm and the typical particles have a diameter of 0.031 mm, the number of cells are calculated by dividing the sphere diameter by the cut-off radius ($r_c^* = 0.035$ mm), which yields 86 cells in each direction plus the 4 padding cells. Thus, simulating the drop experiment with the standardized particle in the standardized hollow sphere would take approximately 8.7 MB of system memory for the linked cell array. However, when the particle diameter is decreased, system memory is exhausted quickly. Due to the cubic memory complexity, the required memory rises excessively, which could play an important role once particle breakup is introduced. Using particles of one tenth the regular size, over 8 GB are required and at one hundredth of the size, approximately 8 TB are required. This problem can be overcome, e.g., by increasing the size of the linked cells to conserve memory at the cost of checking for more interactions.

The cells inside of the sphere stay fixed in their position relative to the sphere during the simulation. This requires some careful adjustments for the involved relative positions of the particles. The simulation works with two different reference frames, one for the observation of the experiment that is fixed relative to the fundament, cf. Figure 4.4. The other one is the system for the sphere and the cells that changes its position relative to the first system over time. To account for this during simulation, the frames need to be synchronized [13].

Definition 4.1. *(Galilean transformation)*
A Galilean transformation transforms the dynamics in one frame so as to be observed in the other frame when the two frames change their position relative to each other.

This concept holds only for constant relative motions. In the considered scenario, the sphere and the particles are accelerated by gravity. However, the velocities of all elements in the system remain constant during every time step and thus, a Galilean transformation can be performed in each iteration.

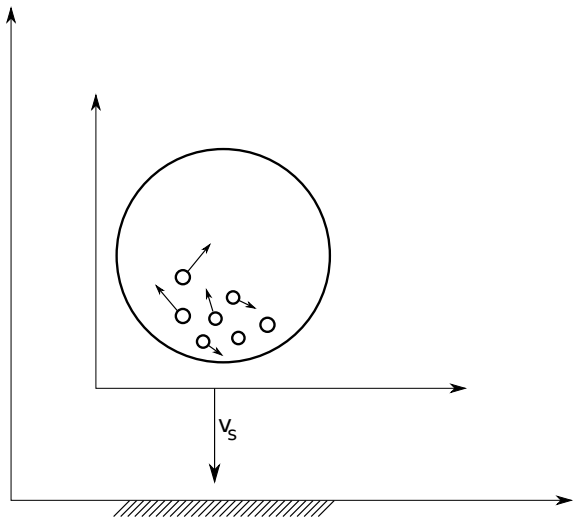


Figure 4.4: Two reference frames in 2D; the first frame is fixed to the fundament and the second one is fixed to the sphere and moves with the velocity v_s

Proposition 4.2. *(Relative frame movement)*
The movement of the particles in the sphere-fixed frame can be tracked in the global frame via a Galilean transformation using the velocity v_s of the frame containing sphere and particles. The sphere is contained in reference frame $\mathcal{A}$ while the other frame $\mathcal{B}$ is the one for the observer. It coincides with the fundament, which is in the $x-z$ plane at $y=0$. In an iteration, the frame $\mathcal{A}$ with the sphere is moved with v_s inside the other system B. Thus, the resulting transformation required for each particle is

$$p_i^B = p_i^A - v_s \Delta t. \tag{4.3}$$

Comparing the results of the quadratic approach and linked cell method, due to the nature of inexact numerics on a computer, this correctional step will lead to slightly different results when using the same simulation parameters. This is visible in the results in the next section below, where an experiment was conducted to compare the two methods.

The interaction of the sphere is the same in both approaches. The barrier potential on the fundament induces a force that acts upon the hollow sphere together with the accumulated forces from the particle-sphere interactions. With the simulation set up, the drop experiment can now be performed numerically.

4.5 Comparison of the linked cell method and the quadratic approach

Compared to the straightforward approach with quadratic complexity, the linked cell algorithm allows to use more particles in a simulation, because a linear behavior of the computation time is maintained. Based on the quadratic complexity of the former, increasing the particle number by a factor of ten, the run time increases hundredfold. Therefore, a linear complexity approach, such as the linked cell algorithm is the only viable option to simulate systems with a large particle number within a tolerable computational time frame. Generally, because of the linear complexity of the linked cell algorithm, simulations complete quicker than with the straightforward quadratic approach. However, the sorting of the particles into the cells also takes some effort, so simulations with small particle numbers will not benefit. The results of both approaches were verified using the same experimental parameters, such as frictional coefficients. To emphasize the need and the advantages of the linked cell method, Table 4.3 lists the computational time required for $1.5 \cdot 10^6$ time steps of 0.1 μs using the straightforward and LC algorithms. Additionally, the rebound time between impacts is presented.

The linked cell algorithm completes faster than the quadratic algorithm for systems that are larger than a few thousand particles. The impact time varies only very slightly. This can be explained by minute numerical differences that occur because the linked cell approach uses two coordinate systems instead of one in the quadratic approach. Still, the quadratic approach retains its use for small systems, e.g., to train friction parameters or for simulations with few (large) particles.

	rebound time in ms		run time in min	
N	SF	LC	SF	LC
10	34.72	34.72	0.05	59
100	33.74	33.74	0.92	65
1000	26.42	26.34	48	80
2500	18.77	18.75	280	105
5000	12.22	12.24	881	113

Table 4.3: Linked cell (LC) and straightforward approach (SF) computational runtime and rebound time

4.6 Energy conservation

The energy of the system has to be conserved in the simulation. In the case considered here, potential and kinetic energy occur. The kinetic energy is due to translational and rotational motion. The kinetic energy of the particles with the individual masses m_i, velocities v_i, inertia tensors I_i and angular velocities ω_i is given by

$$E_p^{kin} = \frac{1}{2}\sum_{i=1}^{n} m_i v_i^2 + \sum_{i=1}^{n} I_i \omega_i^2.$$

The potential energy of the particles consists of two parts; the energy due to gravitation and the energy due to the particle-particle potential U,

$$E_p^{pot} = \sum_{i=1}^{n} m_i g h_i + \sum_{i=1}^{n-1} \sum_{\substack{j=i+1 \\ i \smile j}}^{n} U(d_{ij}).$$

Here $i \smile j$ denotes an interaction of particle i with particle j, h_i the height of the particles above the fundament and g the standard gravity. Similarly, the kinetic energy of the hollow sphere is calculated from the sphere's mass m_s and its velocity v_s,

$$E_s^{kin} = \frac{1}{2} m_s v_s^2,$$

and its potential energy

$$E_s^{pot} = m_s g h_s + U_s(d_s),$$

with h_s the height of the sphere above the fundament and, if present, U_s the potential for the sphere-fundament interaction. Additionally, there is a contribution to the potential energy from the particle-sphere boundary potential,

$$E_{ps}^{pot} = \sum_{\substack{i=1 \\ i \smile s}}^{n} U_b(d_i),$$

where $i \smile s$ denotes that particle i interacts with the hollow sphere. Consequently, the total energy is

$$E = E_p^{pot} + E_p^{kin} + E_s^{pot} + E_s^{kin} + E_{ps}^{pot}. \tag{4.4}$$

Comparing the energy between two time steps serves as a measure of the energy dissipated due to the pseudo-friction introduced in eq. (3.22). Provided that the algorithm is numerically exact and the friction is omitted, the energy should be conserved. However, the leapfrog integrator is only a second order method and therefore it is not to be expected to have perfect energy conservation. Whenever there is an interaction, the length of the time step in combination with the potential-based approach has a direct impact on the resulting kinetic energy. This leads to a so-called energy drift [12].

When considering two separate simulations that proceed with different time steps $\Delta t_1, \Delta t_2$ and assuming, that at a certain point in time they both have the numerically exact same energy E_{total}, this drift can be explained. If in the propagation to the next time step an interaction in both simulations occurs where the positions, and therefore the distance d, between the interaction partners is also the same, the same force will be present due to the potential

$$F = \frac{24\varepsilon}{d^2}\left(2\left(\frac{\sigma}{d}\right)^{12} - \left(\frac{\sigma}{d}\right)^{6}\right)d,$$

where ε is the depth of the potential well and σ the particle diameter. However, it is assumed that the same force acts throughout the duration of a time step. In order to propagate both systems to the next time step, this force will act longer in case of a larger step $\Delta t_2 > \Delta t_1$. Assuming that there is no other interaction during this time step, and omitting gravity for the moment, the resulting change in kinetic energy is

$$\Delta E_1^{kin} = \frac{\Delta t_1}{2m}F \neq \frac{\Delta t_2}{2m}F = \Delta E_2^{kin}.$$

Thus, the kinetic energy cannot be perfectly conserved, but depends on the chosen time step. The total energy of the system after the interaction will not be exactly the same as before the interaction occurred.

It is advisable to use smaller time steps, however, a balance between accuracy and simulation time has to be found. A very small time step would prohibitively increase the computational effort. In MD, a thermostat is often used to recalibrate the dynamics of the system every few time steps. Because the velocity of the molecules is tied to the temperature of the material, the simulation can be forced to retain a constant temperature, meaning that a correctional factor has to be applied to the velocities [2, 12]. In the

problem modeled here, the decrease in energy due to friction is much more pronounced than the impact of a large time step. Thus, highly accurate energy conservation was not as important and a larger time step was chosen in this work.

4.7 Energy experiment

To exemplify the energy conserving nature of the potential based approach in connection with the leapfrog algorithm, an experiment was conducted first with the sphere itself only. For that purpose, the drop on the fundament was performed first for an empty sphere with and without friction and then, particles were added.

The experiment was run for 150000 time steps of 1 μs. During that time, a total of four rebounds were observed. Figure 4.5 shows the various forms of energy present in the system along with the total energy. Friction was turned off and the barrier potential was used for the interaction with the fundament. The gravitational potential was shifted to reflect a drop onto a raised platform - resulting in a higher potential energy, to allow for better readability of the different parts of energy in the graph. As can be seen in the graph, the total energy remains constant over time, making a strong case for the use of the barrier potential. Now, with added pseudo-friction, the same forms of energy are plotted in Figure 4.6. In the total energy, the velocity dependent friction can be observed. In each time step where the barrier was active, 0.001% of the velocity was subtracted, resulting in a reduction of total energy. The sphere behaves like a bouncing basketball, with the bounce height decreasing over time.

When particles were added (2000 in this next experiment), i.e., a filled sphere was considered, with the friction set to zero, the energy should still be preserved. As can be seen in Figure 4.7, this is in fact the case. However, there is a slight shift visible that favors kinetic energy over time. The potential energy decreases slightly, signaling a damping effect in the sphere, while the particles are accelerated, visible in a rise of kinetic energy. Visualizing very long simulation runs, a speedup of the particles inside the sphere can be clearly observed because of this shift. Adding pseudo-friction, the effect of the particles on the damping of the sphere is very noticeable in the total energy, cf. Figure 4.8. The most prominent dissipation of energy is clearly visible in the graph and occurs shortly after the first rebound of the

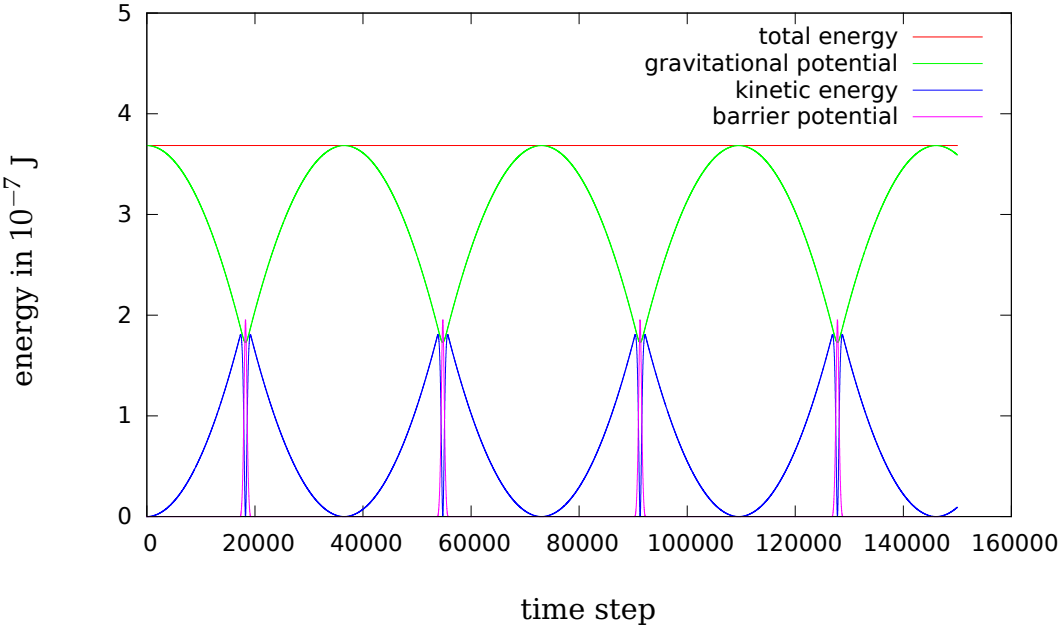


Figure 4.5: Energy contributions of an empty bouncing sphere without friction

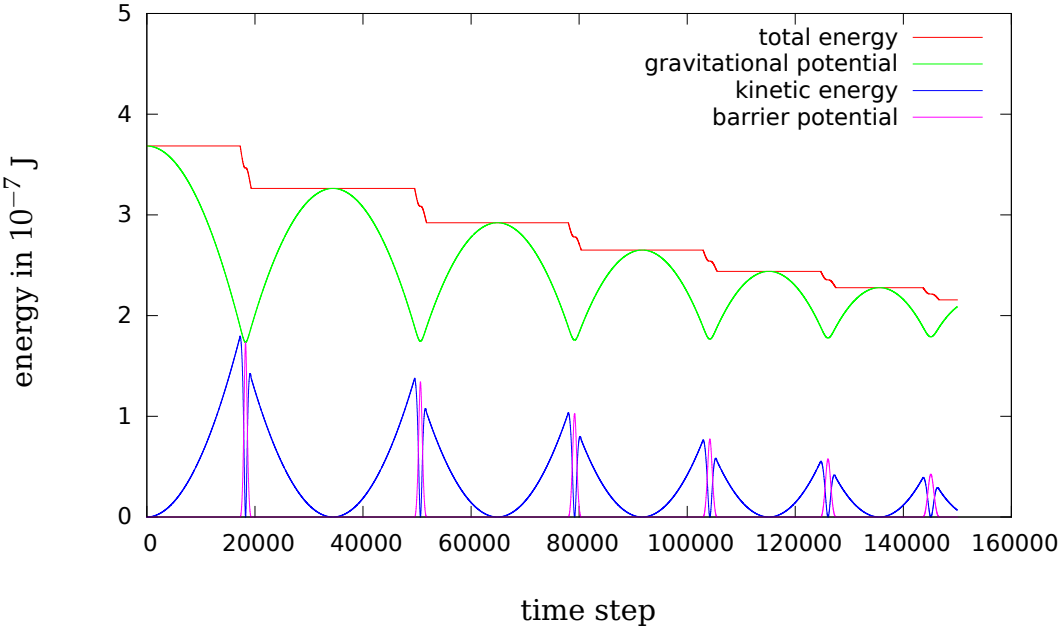


Figure 4.6: Energy contributions of an empty sphere with friction

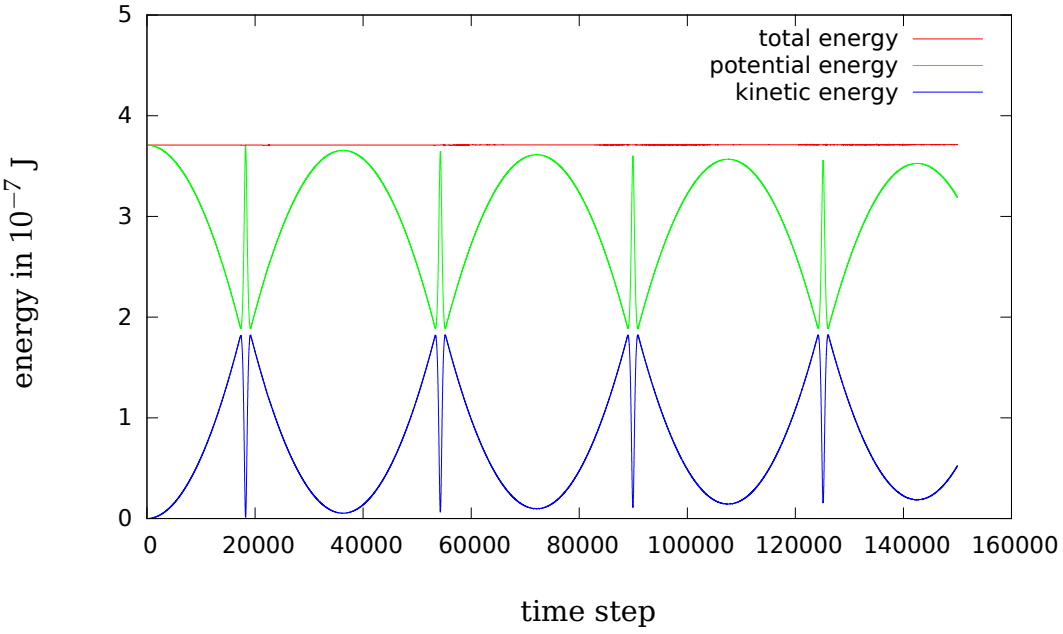


Figure 4.7: Energy contributions of a sphere containing 2000 particles without friction

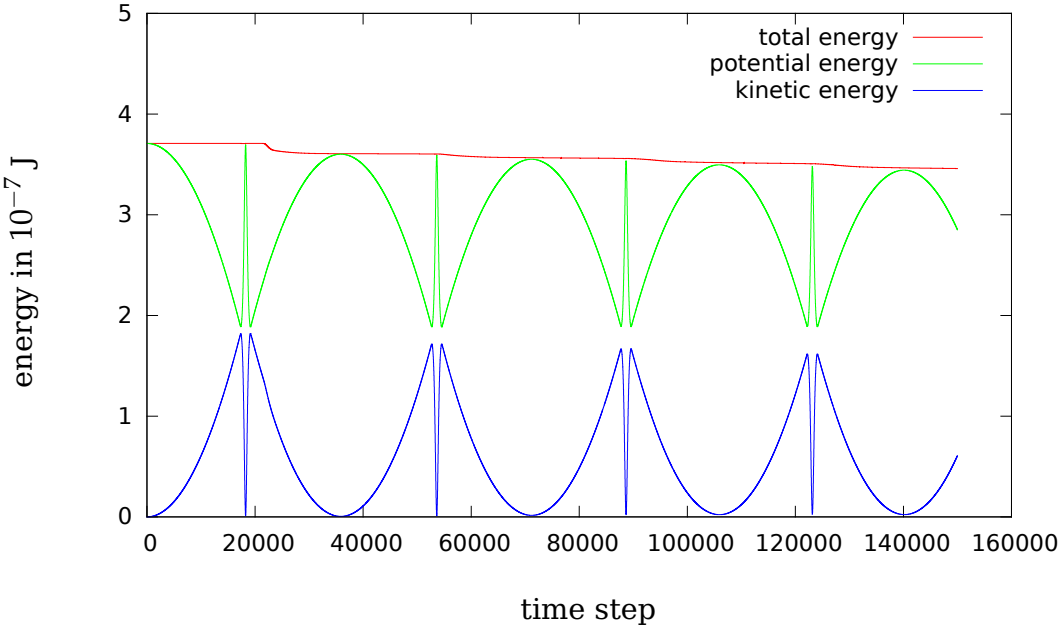


Figure 4.8: Energy contributions of a sphere containing 2000 particles with friction

sphere, when it collides with the bulk of particles. After this initial forceful interaction, the particles move freely inside the cavity in all directions. Thus, a similar single imposing event with the same impact on the energy cannot be expected, but it continues to be dissipated gradually over time. In simulations with well over 100000 particles this initial event can be so powerful, that the sphere rebounds immediately. In that case, the third bounce will take much longer than the second bounce, requiring another measure than the rebound time ΔT for the calculation of the coefficient of restitution.

4.8 Numerical properties

The simulation is based on the MD leapfrog algorithm derived from a Taylor expansion on the first order differential equations. As shown in chapter 2, in the leaping scheme, second order accuracy is achieved. The changes in the model do not influence that result since the differences are in the computation of the forces and boundaries. Therefore, second order accuracy is maintained.

The changes do, however, have an impact on numerical stability. The term stability has been coined for numerous different things. For the purpose of this thesis, the numerical aspect is considered. This means that an examination of stability refers to the behavior of an algorithm with respect to error amplification, or conservation of energy. The leapfrog method is based on a truncated Taylor expansion that will lead inevitably to discretization or truncation errors in the computation. The question that needs to be answered is, whether these errors cause the solution to become unstable. For this purpose, it is necessary to evaluate the errors in each time step. The stability of the method ultimately depends on the time step Δt. This becomes evident in the interaction computation, where an inadequate time step choice may lead to interacting particles with a large overlap, resulting in a great repulsive force that can destabilize the simulation. In the considered problem, the maximum velocities of the particles will influence the maximum force, and the velocities depend on the time interval when the sphere drops and is subjected to gravity. Recall that in the leapfrog scheme, the velocity is

$$v(t + \Delta t) = v(t) + \Delta t F(t)/m. \tag{4.5}$$

To achieve stability, the energy in the simulated system should remain constant. Thus, the interaction force F in that time step must not accelerate the

particles and at its greatest permitted extent, cause the particles to assume the positions they had just prior to the interaction. Thus, it has to hold that

$$v_{max} = -v_{max} + \Delta t F/m \tag{4.6}$$

$$\Longrightarrow F = 2mv_{max}/\Delta t \tag{4.7}$$

To calculate the maximum force F that is permitted to occur, the maximum velocity the particles may achieve is required. For any body that is dropped from height h_d, their terminal velocity at the fundament is $v_{drop} = \sqrt{2gh_d}$. The particles achieve their maximum velocity when they are reflected from the sphere boundary and when that collision is elastic. In a perfect elastic head-on collision of two masses m_1 and m_2 with pre-collision velocities v_1 and v_2, the post-collision velocities v_1' and v_2' are [4]

$$v_1' = \frac{m_1 - m_2}{m_1 + m_2} v_1 + \frac{2m_2}{m_1 + m_2} v_2 \tag{4.8}$$

$$v_2' = \frac{2m_1}{m_1 + m_2} v_1 - \frac{m_1 - m_2}{m_1 + m_2} v_2. \tag{4.9}$$

In the problem considered here, the masses of the sphere and particle differ greatly. Then, with $m_1 \gg m_2$, the sphere's velocity in (4.8) remains virtually unchanged, while the particle's velocity after the interaction (4.9) is dramatically increased by the impact with the heavy sphere in addition to a complete reversal of the particle's own pre-collision velocity. Thus, the maximum velocity of a particle is $v_{max} = 3\sqrt{2gh_d}$.

The greatest theoretical force that can occur is when two particles collide with that velocity, when they had barely touched in the previous step (the distance equal to the cut-off radius r_c^*). Their distance during the interaction in the next time step is

$$d = r_c^* - \Delta t 2 v_{max}$$

which means that the force is

$$F(d) = \frac{48\varepsilon}{d} \left(\left(\frac{\sigma}{d} \right)^{12} - \frac{1}{2} \left(\frac{\sigma}{d} \right)^{6} \right).$$

This term can be used in eq. (4.7) to obtain a solution for Δt (here, it is

solved for d first for better readability)

$$\begin{aligned}
\frac{48\varepsilon}{d}\left(\left(\frac{\sigma}{d}\right)^{12}-\frac{1}{2}\left(\frac{\sigma}{d}\right)^{6}\right) &= 4mv_{max}^2/(r_c^*-d)\\
12\varepsilon(r_c^*-d)\left(\left(\frac{\sigma}{d}\right)^{12}-\frac{1}{2}\left(\frac{\sigma}{d}\right)^{6}\right) &= dmv_{max}^2\\
12\varepsilon\sigma^{12}(r_c^*-d)-6\varepsilon\sigma^6 d^6(r_c^*-d)-d^{13}mv_{max}^2 &= 0\\
-mv_{max}^2 d^{13}+6\varepsilon\sigma^6 d^7-6\varepsilon r_c^*\sigma^6 d^6-12\varepsilon\sigma^{12}d+12\varepsilon\sigma^{12}r_c^* &= 0.
\end{aligned}$$

For time steps that satisfy this condition, the numerical algorithm performs well. The dynamics of the system may behave badly for larger time steps and will yield invalid results. Solving this equation numerically for an exemplary drop experiment with $h_d = 0.1$ m and a representative particle with $\sigma = 0.31 * 10^{-1}$ mm, results in

$$d = (-1.44581, -0.34797, 0.346592, 0.351522, 1.32497)\ 10^{-4}\text{m}.$$

Only the third solution is a valid choice for d (positive distance and smaller than r_c^*) and returns a critical step size

$$\Delta t = 1.2823 \cdot 10^{-7}\ \text{s}. \tag{4.10}$$

With the step sizes selected in this manner, the experiments can commence in the next chapter.

Chapter 5

Results

With the suggested method, the damping behavior of a particle filled sphere can be simulated. There are several interesting experiments that can be performed with the software to compare them with real world experiments. The following pages give a detailed view on the results of these numerical experiments.

The simulation results were obtained by using the experimental data supplied by Fraunhofer IFAM Dresden during the project. The real particles vary in size due to the manufacturing process, but a representative particle has been identified by Fraunhofer IFAM Dresden that was used for most of the presented simulations. Table 5.1 gives an overview of the typical parameters used in the simulations for this standard particle as well as the representative sphere, where the parameters were also obtained from Fraunhofer IFAM Dresden. For reference, commonly used parameters for MD simulations are provided as well. The simulations for the damping behavior were performed on the OCuLUS system at the Paderborn Center for Parallel Computing on nodes equipped with Intel Xeon Sandy Bridge E5-2670 CPUs with 4 GB RAM per core.

Explanation	Parameter	Typical value used	MD value
time step	Δt	0.1 μs	1 fs
particle diameter	d_p	$3.1 \cdot 10^{-2}$ mm	$30 - 300$ pm
particle mass	m_p	$4.1 \cdot 10^{-5}$ mg	$10^{-27} - 10^{-25}$ kg
friction part.-part.	f_p	0.1	-
friction part.-sphere	f_s	0.01	-
sphere diameter	r	3 mm	-
sphere mass	m_s	12.518 mg	-

Table 5.1: Input data for the numerical experiment

Friction parameters

Due to the nature of the potential-based friction used in this work, an undesired effect of particle clumping can occur. This happens when the friction between particles is too large, and particles start to move together. This can be avoided by adjusting the parameters accordingly. For this purpose, the quadratic simulation tool was very helpful. It provided a quick visualization of just a few particles, where clumping is immediately visible. However, the effect is also desirable to some extent, because real particles do clump together, especially during sintering, cf. Figure 3.8. For the following experiments, the parameters f_p for the particle-particle interactions and f_s for the particle-sphere interactions will be stated for each simulation. Concerning the parameter governing the sphere-fundament interaction, an experiment with an empty sphere was performed to train the simulation to the real world behavior of such a sphere and resulted in a parameter of 0.9. Additionally, all friction parameters were multiplied by the time step to make the results comparable when using different time steps. Not all numerical values are provided in the following sections, where plots offer a clearer presentation. The complementing tables can be found in the Appendix and are referenced in the text.

5.1 Simulation 1: Damping behavior

The first simulation follows up on the most interesting question on how the damping evolves with a growing number of particles and examines the time between the first two impacts of the sphere on the fundament. Recall that from this rebound time, the coefficient of restitution can be calculated. This experiment used a drop height of $h_0 = 10$ cm and friction parameters $f_p = 0.05$ and $f_s = 0.02$ as well as the dimensions of the representative particle in Table 5.1. These settings were also used for the other simulations, unless stated explicitly. The results of the experiment are shown in Figure 5.1 and, for reference, the numerical values can be found in Table A.1. As expected, the damping effect of the particles grows with the number of the particles. This is also the case in the physical experiment at Fraunhofer IFAM Dresden [20]. The impact of the particles is already very pronounced for relatively few particles, and continues to increase - though to a lesser extent - as very high numbers of particles are reached. A similar behavior has also been observed in particle damper experiments [28]. These devices

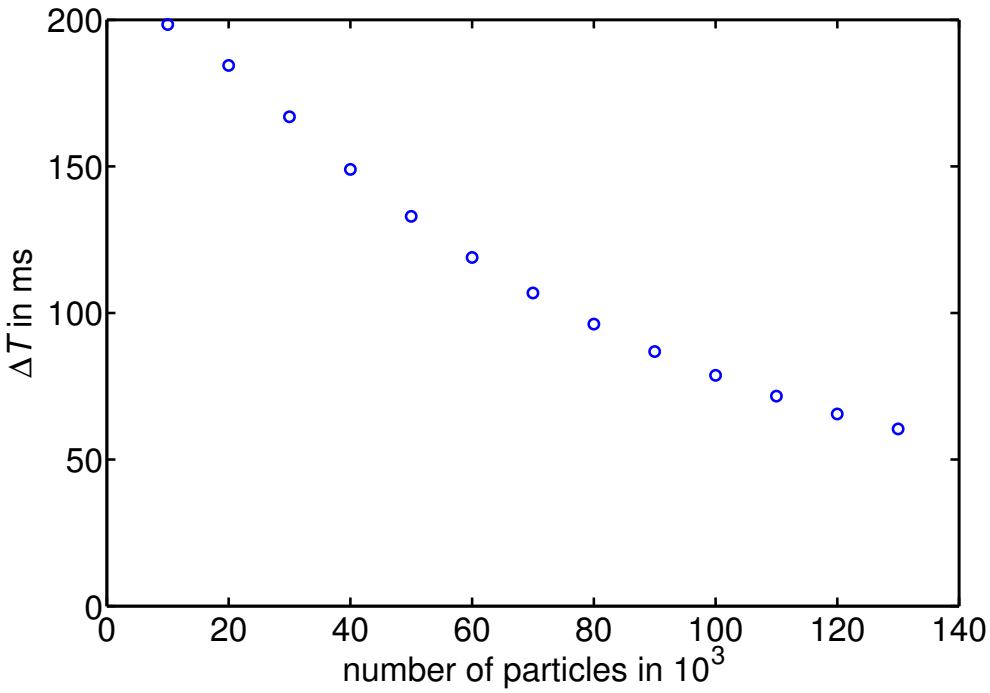


Figure 5.1: Rebound time ΔT in the drop experiment for an increasing number of particles

consisted of a particle filled enclosure attached to another body. The damping performance of the assembled system depended on the gap between the particle heap at rest and the damper ceiling. The experiments showed that as the number of particles increases and continue to fill up the cavity, the impact of the particles is diminished. The resulting damper approached the behavior of an unfilled, heavier damper with a mass that accounted for the added particle mass, but where that system performed worse than a lighter, more sparsely filled damper. It was suggested that this occurs because in that case, the particles are not free to move inside the enclosure and are not able to contribute a damping effect. Having established an effect of the particle filling with this first experiment, it is now of interest to look closer at the developed method.

5.2 Simulation 2: Time step size

This simulation was performed to test the stability measure developed in the previous chapter. To improve the time required for the computations, a larger time step could be used. To verify the integrity of the results, the time

N	Δt=0.01	Δt=0.05	Δt=0.1
1	207.51	207.51	207.52
25	207.48	207.48	207.49
50	207.45	207.45	207.46
100	207.38	207.39	207.38
250	207.18	207.18	207.18
500	206.84	206.83	206.84
750	206.47	206.50	206.49
1000	206.14	206.13	206.15
1500	205.47	205.46	205.47
2000	204.77	204.74	204.76
2500	204.10	204.10	204.12
5000	200.72	200.78	200.78
7500	197.48	197.43	197.44
10000	194.08	194.12	194.13
12500	190.85	190.89	190.86
15000	187.63	187.61	187.54
17500	184.38	184.41	184.46
20000	181.29	181.34	181.29

Table 5.2: Rebound time (in ms) for various time steps

step was set as 0.01, 0.05 and 0.1 μs. For the friction parameters, $f_p = f_s = 0$ was used. As can be observed in Table 5.2, the results are comparable and only offer minute differences that stem from the numerical noise. Thus, the method scales very well with the step size and, more importantly, does remain stable up to the calculated stability limit.

Another experiment on the impact of the variation of the time step without friction was also conducted in Ref. [31]. The drop height was much lower at $h_0 = 1.5$ mm and a variety of time steps ranging from 0.01 to 5 μs was used, well above the stability limit (which with this experimental setup would be around 0.637 μs). Figure 5.2 shows that a too large time step will lead to rebound time results that differ greatly from those obtained with smaller time steps. However, more significantly, as soon as the time step is small enough, the selection of an ever finer time discretization will not yield significantly differing results. Also, note how a step size of 1 and 2 μs still offers good results, indicating that the stability estimate is quite conservative and might be relaxed in the future. The numerical results can be found in Table A.2.

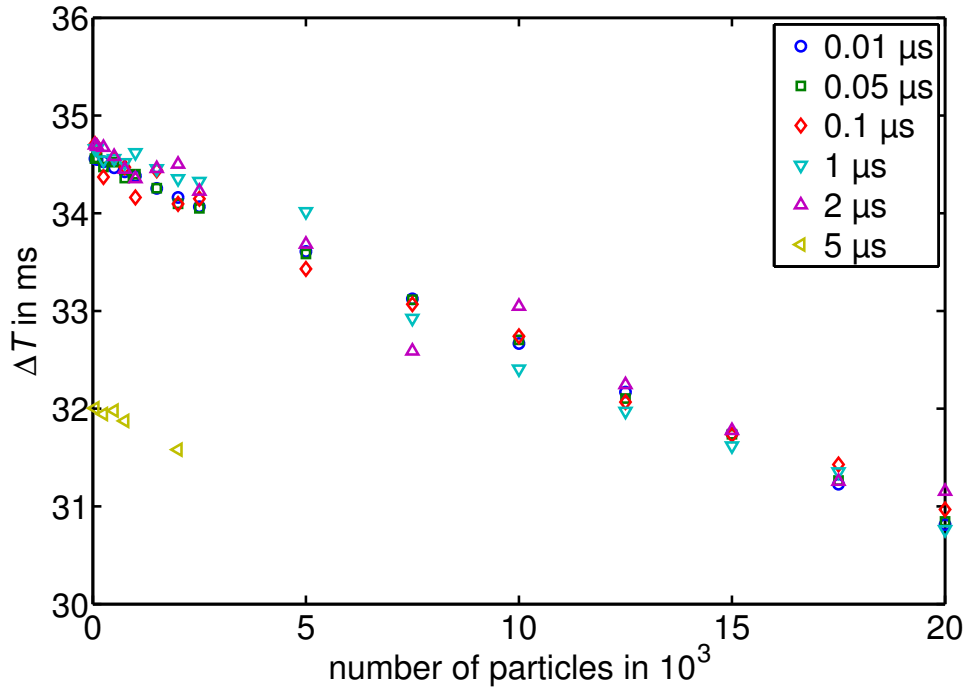


Figure 5.2: Rebound time for a variation of time steps in a drop experiment with an increasing number of particles without friction

5.3 Simulation 3: Particle size

The next experiment observes the damping behavior when varying the particle dimensions. As mentioned before, in the manufacturing process, the average diameter of the particles can be selected within a certain range. Thus, a natural question is, which type of particle is best suited and should be selected for the most pronounced damping effect. It could be suspected that there will be a conflict of mass vs. number of particles. While smaller particles may result in a higher number of interactions with the wall, their mass is low. Thus, larger particles with a greater mass may yield a better damping effect. In this next simulation, smaller and larger particles were compared, with constant density and total filling mass and friction parameters $f_p = 0.1$ and $f_s = 0.05$. According to the real-world experiments, smaller particles seem to offer a greater damping potential than larger particles [21]. However, as shown by this experiment, this is not the case for all simulations. Surprisingly, the results in Figure 5.3 (and Table A.3) show a fundamental change in the damping performance of the particles as the total mass of the filling increases. For a very small filling mass, with fewer particles, large particles yield a slightly better damping effect than small

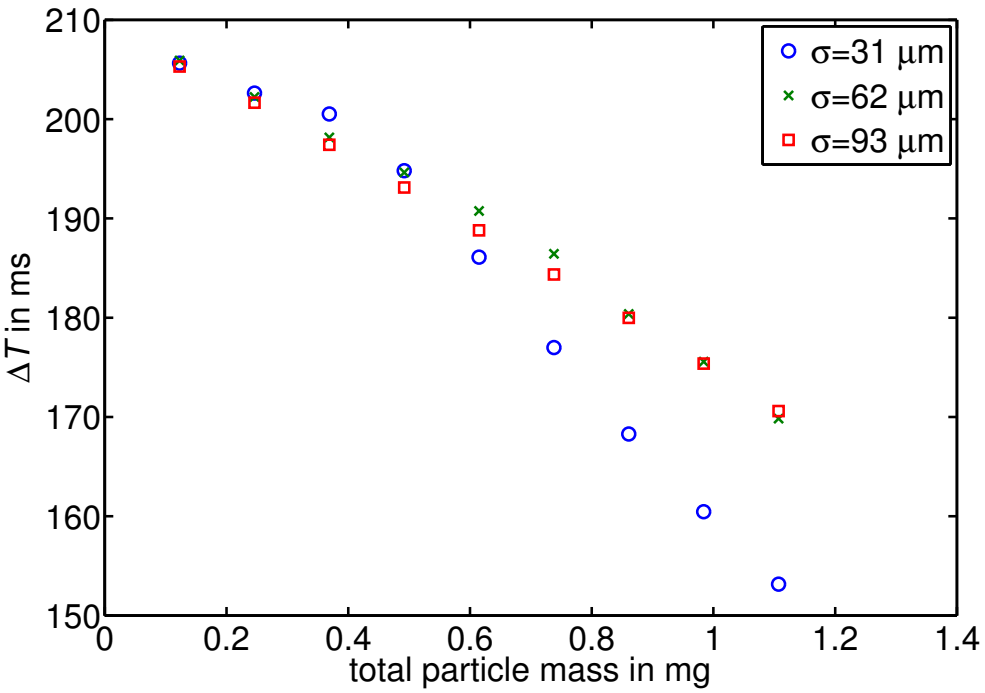


Figure 5.3: Rebound time for a variation of the total particle mass for different particle diameters

particles. However, this behavior reverses itself as the mass is increased. Thus, the ratio of total particle mass and hollow sphere mass plays an important role. For a greater number of particles, the smaller ones show a clear advantage over the large particles and the difference in performance is much more pronounced.

Another experiment with particles with a rod-like shape of varying length (1, 2, 3 and 4 rigidly connected spheres of the same mass and diameter) was conducted. Again, the total number of spheres - and thus, the total particle mass - was kept constant, with a total of 12000, 24000 and 36000 spherical particles for all shapes. The individual parts resembled the representative particle and friction parameters were $f_p = 0.1$ and $f_s = 0.02$ throughout the experiment. The results can be observed in Table 5.3. Similarly to the previous simulation, there are some particles that perform worse than others; in this experiment, the medium sized particles excel, while the smallest particle performed worst.

rod size	ΔT in ms		
1 particle	194.98	160.74	134.45
2 particles	164.65	135.00	116.73
3 particles	166.34	138.88	120.74
4 particles	169.05	142.11	121.89
total particles	12000	24000	36000

Table 5.3: Rebound time for varying particle lengths

5.4 Simulation 4: Impact of the friction parameters

To later train the simulation model on the basis of real world experimental data, the impact of the friction parameters was studied. Keeping the total mass of the filling constant, the experiment was performed with larger or smaller particles that have the same mass density. The experiment doubled the particle volume in each simulation run. The specific diameters and masses are stated in Table A.4. The ratio of particle mass and hollow sphere mass was motivated by Fraunhofer IFAM Dresden data. For the first part of the experiment, the parameter f_s governing the sphere-particle interactions remained constant and the particle-particle friction parameter f_p was varied. The results can be seen in Figure 5.4. In a second part, f_s was varied and f_p was set as constant for the simulations in Figure 5.5. For reference, all numerical values are listed in Table A.5.

The previous experiment suggested a better performance of the smaller particles when the filling mass is as high as in this simulation. In line with those observations, this friction experiment confirms that the smallest particles offer the best performance, with a dramatic decrease in performance when the diameter is increased. The friction experiment suggests a greater influence of parameter f_p, while variations of f_s only bring about small changes of the rebound time ΔT.

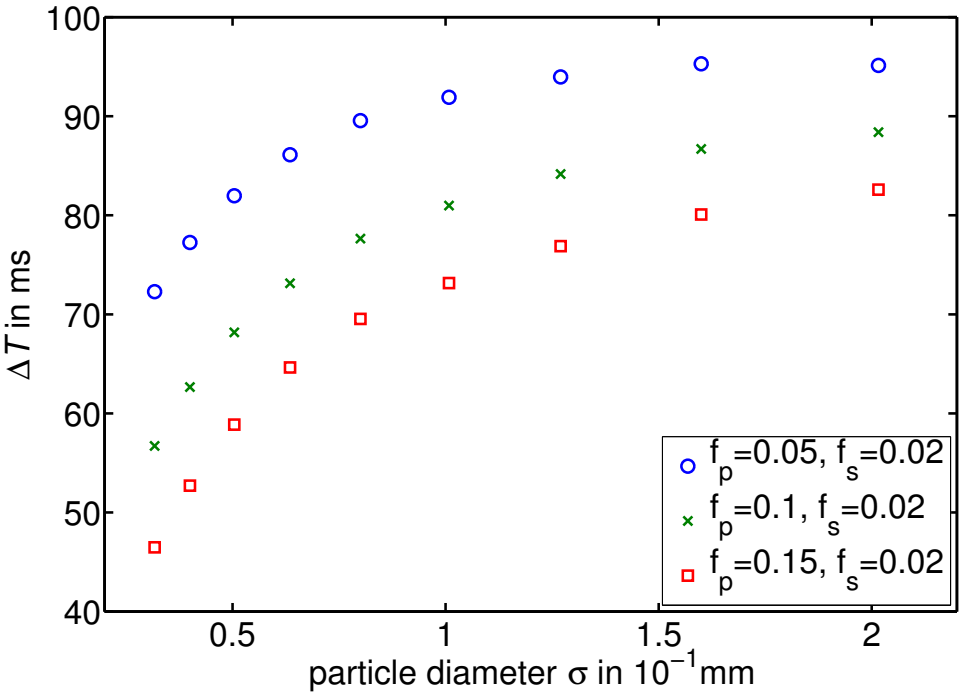


Figure 5.4: Rebound time ΔT for increasing particle diameter and varying particle-particle friction with constant particle-sphere friction parameter f_s

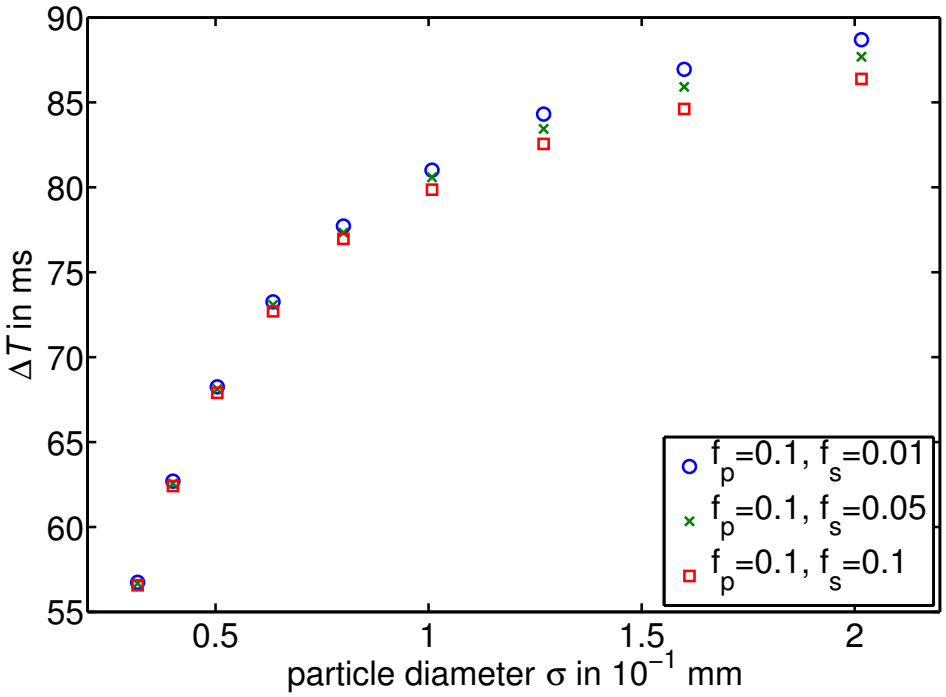


Figure 5.5: Rebound time ΔT for increasing particle diameter and varying particle-sphere friction with constant particle-particle friction f_p

5.5 Simulation 5: Particle shape

It is also of interest to conduct an experiment with variously shaped particles that have the same mass. For this purpose, the simulation was conducted with particles that consisted of five spherical particles and assumed the shape of “L”, “V”, “I”, “O” and “T”, cf. Figure 5.6 with $f_p = 0.1$ and $f_s = 0.02$. Particle definitions are provided in the Appendix in Table A.6. The experiment used up to a total of 12000 spherical parts, all similar to the representative particle, and the observed rebound times are presented in Table 5.4.

Figure 5.6: Different particle shapes: L, V, I, O and T (from left to right)

The results from these shape experiments seem to indicate that the shape is not as important in determining the damping performance of the particles. However, one exception is the rod shaped "I" particle, that performs worse across the board than the other, bulkier particles that concentrate their mass in a more compact shape.

	number of particles N					
type	1000	2000	3000	4000	5000	6000
L	206.21	203.90	200.72	196.76	192.68	189.03
V	206.56	204.18	201.44	197.92	193.69	189.03
I	206.77	204.55	201.33	197.70	195.37	191.42
O	206.80	204.17	201.51	197.23	193.82	188.64
T	206.35	204.14	200.62	196.76	192.34	187.61
	number of particles N					
type	7000	8000	9000	10000	11000	12000
L	184.25	179.73	176.53	171.84	169.36	166.07
V	184.66	180.48	176.60	173.05	169.55	165.89
I	188.11	184.20	181.93	179.19	175.15	171.73
O	184.29	180.70	175.53	172.10	168.76	165.71
T	183.43	179.90	176.17	173.06	169.22	165.55

Table 5.4: Rebound time (in ms) for particles with different shapes and constant mass

5.6 Simulation 6: Mixed particle fillings

Another question is whether a homogeneous heap of particles performs better with respect to the achieved damping than a mixed heap. For this purpose, several simulations were conducted for a mixture of particles that have different sizes. Because of the influx of heat during the sintering process of the encasing sphere, parts of the powder inside will begin to melt and combine to form larger particles. Expectedly, this results in a heterogeneous powder with larger and smaller particles. Thus, it is of interest to provide the ability to simulate particle mixtures. For the simulation, four different mixtures were used to compare the damping effect. Note that the total mass of particles was kept constant during all experiments as well as the mass density across the particle diameters. Table 5.5 lists the mixtures that were used. The mixtures were prepared such that the same fraction of mass was allocated to each particle type and the friction settings were $f_p = 0.1$ and $f_s = 0.02$. The results obtained are given in Table 5.6.

It was established in Simulation 3, that the ratio of particle mass and sphere mass determines what particle size performs best. In this mixture experiment, that ratio assumes a value where the particle size does not make a great difference on the damping performance. This is mirrored in the results, where, as the mass increases, only Mix 3 with both larger particle types shows a disadvantage compared to the other mixtures. One could suspect that this may get more pronounced as the mass increases even further. However, a numerical simulation of much higher masses would require improved packing strategies for the initial placement of the particles to fit more particles in the sphere.

	diameter in 10^{-1} mm		
	0.31	0.62	0.93
Mix 1	6000	750	
Mix 2	6060		220
Mix 3		771	210
Mix 4	4004	500	148

Table 5.5: Composition of particle mixtures in the hollow sphere for the experiment with a total mass of 0.369 mg

	rebound time in ms			
Mix 1	205.73	203.49	199.57	191.71
Mix 2	205.55	203.48	197.60	189.91
Mix 3	205.03	202.33	198.33	195.57
Mix 4	205.69	203.76	198.28	189.65
mass in mg	0.123	0.246	0.369	0.492

Table 5.6: Rebound time for particle mixtures

5.7 Simulation 7: Vibration of the sphere

When the lightweight material is deployed in the real world, the hollow spheres are subjected to vibration. To mirror this in the numerical experiment, the sphere was modeled to accommodate this by a deformable hull. For the following simulation, the pulsing radius vibration was used with an amplitude of 0.1 μm, with frequencies that may occur in industrial CNC machines, i.e. 6000, 12000, 18000 and 24000 revolutions per minute, or 100, 200, 300 and 400 Hz, while the friction parameters were with $f_p = 0.1$ and $f_s = 0.02$. Observing the results in Figure 5.7, adding vibration does not change the overall trend that the damping effect increases as the particle number rises. Because the movement of the sphere results in more interactions, a higher frequency actually improved the impact times slightly (also see Table A.7).

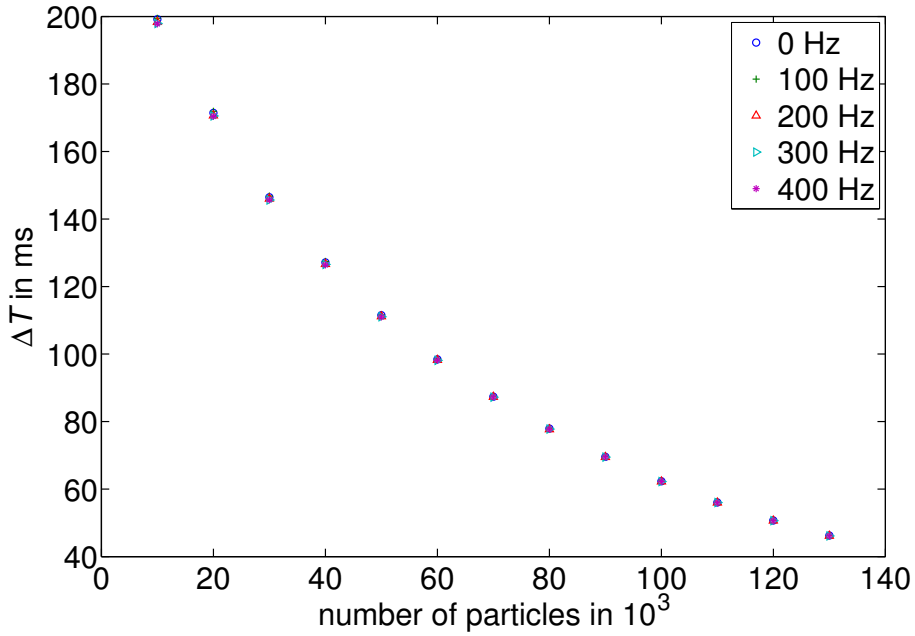


Figure 5.7: Rebound time for the numerical vibration experiment

Overall, the performed numerical experiments agree with the assessment of Fraunhofer IFAM Dresden, that smaller particles offer a greater potential for damping than larger particles. Additionally, the numerical simulations suggest that the optimal size depends on the ratio of sphere mass and particle mass. Concerning the shape of the particles, more compact particles that concentrate their mass in a smaller volume performed better. The friction parameters introduced offer flexibility in training the model to real-world data. With the capability to depict complex particles in all shapes and sizes, the simulation is well prepared to cover a wide variety of experiments.

Chapter 6

Outlook: Future developments of the model

The simulation of a single hollow sphere leads to the natural question of how to simulate the more complex and complete model of the lightweight material that makes use of these sphere dampers.

The first problem that arises in that broader model is how to couple the spheres. Recall that the spheres in the material can be bonded using different techniques, i.e., a soldered or glued bond. The underlying coupling model needs to be flexible enough to accommodate both stiff and more flexible bonds. A natural choice would be to use a coupled spring-dashpot system for representing the bonds between two spheres, cf. Fig. 6.1. The spring and dashpot can be adjusted to represent a multitude of materials by modifying the spring parameter k and the damper coefficient c. Note that these parameters are unrelated to the damping parameters in the previous chapter where potentials were used. Repeating the drop experiment with such a bond requires a new model for the forces acting on the spheres.

Definition 6.1 (Spring-dashpot bond model)**.** *A spring-dashpot model of a bond with bonding gap length w of two spheres and with parameters k and c is modeled by the system*

$$\begin{aligned} F_1 &= -m_1 g + k(x_1 - x_2 - w) - c(\dot{x}_2 - \dot{x}_1) \\ F_2 &= -m_2 g - k(x_1 - x_2 - w) + c(\dot{x}_2 - \dot{x}_1), \end{aligned} \tag{6.1}$$

where g is the standard gravity, and with initial conditions

$$\begin{aligned} x_1 = h_0,\ x_2 = h_0 + w \\ \dot{x}_1 = 0,\ \dot{x}_2 = 0. \end{aligned}$$

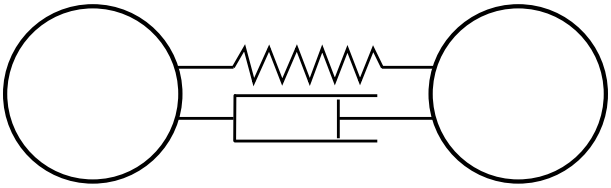


Figure 6.1: Two spheres with a spring-dashpot bond

The damper resists motion via friction while the spring restricts the displacement of the bond during the simulation. Apart from this adjustment of the forces acting on the bonded spheres, the basic task of solving the Newtonian equations of motion for the drop experiment remains unchanged. For the reflection on the fundament, either the discrete or continuous potential based approach could be used. Here, the discrete collisions could be more desirable, as one could assume that the correct coefficient of restitution for a filled sphere has been discovered in a prior, more detailed simulation run. The impact however, marks a drastic change in the velocity direction and thus a singularity in the model equation that needs to be dealt with, e.g., by setting the second derivative to zero. During an ECMI modeling week in Paderborn, a student group was tasked with the implementation of this model in MATLAB, cf. [9]. The model (6.1) along with a discrete fundament reflection was used to simulate the drop experiment of two spheres arranged in such a way that one sphere sits on top of the other and the spheres are dropped together with a spring-dashpot bond. The results of the simulation are shown in Fig. 6.2.

The graphs of the spheres' position above the fundament show the influence of the bond, the distance of the spheres oscillating around the equilibrium w. This basic model is a promising choice to examine the bonding of several particle-filled spheres while omitting the need to simulate the particle behavior in complete detail.

The spring-dashpot model only allows for one degree of freedom; moving to a three-dimensional simulation, tangential spring-dashpot connections on each displacement axis have to be added at each bond to allow movement in all directions, creating a set of three spring-dashpot pairs for every sphere-to-sphere contact. Naturally, the total number of bonds for each sphere depends on the number of neighboring spheres, which are added as additional terms in eq. (6.1). The parameters k and c can be used to model different bonds, controlling the stiffness of the matrix material.

Due to the complexity of the simulation, a complete, fully detailed numerical computation of the material seems out of the question for now.

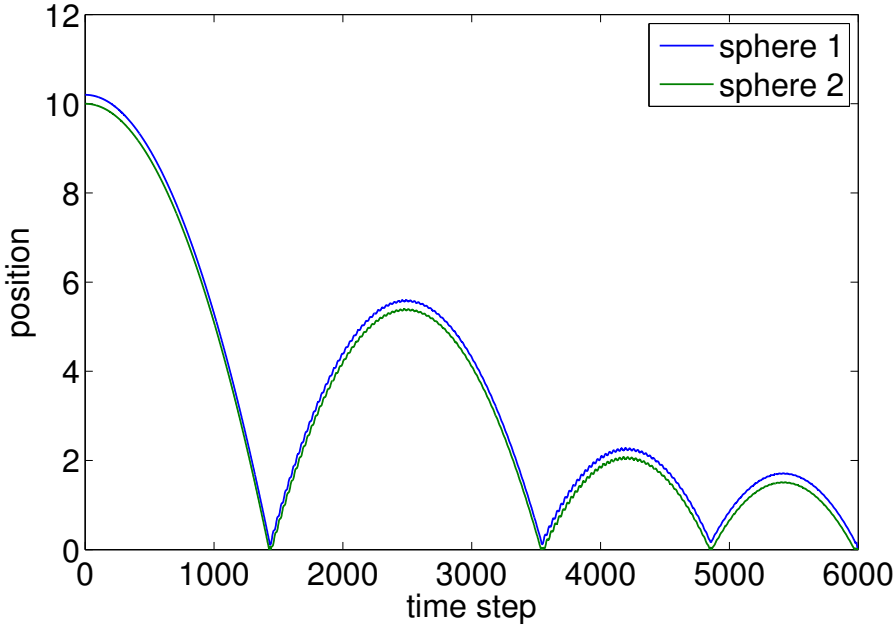


Figure 6.2: Simulation results of the drop experiment with two bonded spheres

Therefore, it would be plausible to move away from the detailed model to a broader perspective. Instead of simulating each sphere with the particles individually, the composite material with many embedded spheres is modeled, but still taking into account the effect of the particles.

To that end, the characteristic damping behavior of a filled hollow sphere could be approximated by a different mathematical model, e.g., by introducing a self-damping term in the sphere's force term

$$\begin{aligned} F_1 &= -m_1 g + k(x_1 - x_2 - w) - c(\dot{x}_2 - \dot{x}_1) + P(\dot{x}_1(t)) \\ F_2 &= -m_2 g - k(x_1 - x_2 - w) + c(\dot{x}_2 - \dot{x}_1) + P(\dot{x}_2(t)). \end{aligned} \tag{6.2}$$

In this model, the damping term P is inversely proportional to the particle's velocity $v_i(t) = \dot{x}_i(t)$

$$P(v_i(t)) = \begin{cases} \alpha v_i(t), & \text{if } v_i(t) < \nu \\ \frac{d}{v_i(t)}, & \text{if } v_i(t) \geq \nu, \end{cases}$$

where α and d are constants and a small tolerance $\nu > 0$. These parameters need to be trained to mimic the effect of the particles in the hollow sphere.

The results of such a simulation with $\alpha = 0.2$, $d = 0.55$ and $\nu = 0.1$ are

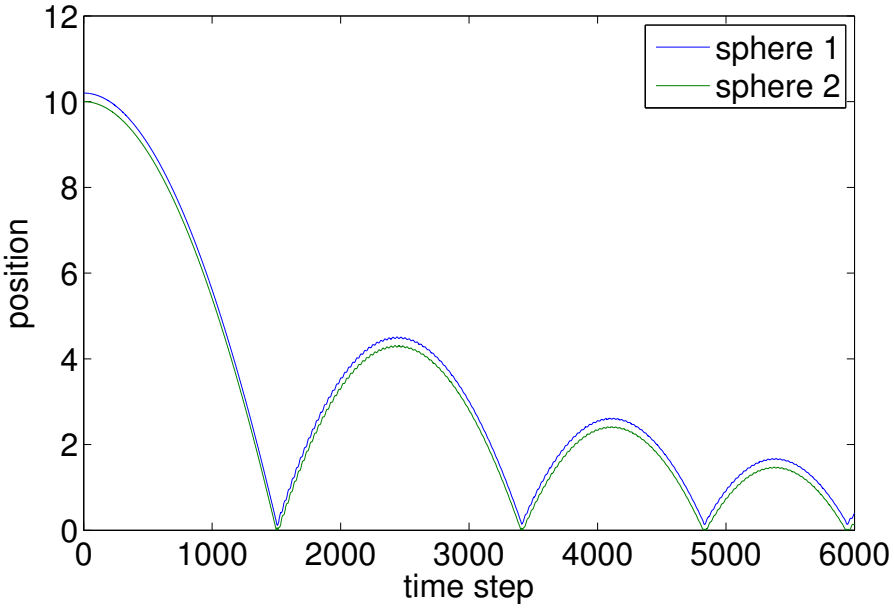


Figure 6.3: Simulation results of the drop experiment of two bonded spheres with an additional self-damping term

shown in Fig. 6.3. Compared to the first model, the damping effect is more pronounced, significantly so in the first rebound. This model may offer a decent approximation when several thousand spheres are bonded together in an enclosure, with the model parameters fitted to resemble the behavior of a filled sphere simulated in detail in a previous step.

Chapter 7

Conclusion

It was shown that to derive the coefficient of restitution of a particle filled hollow sphere, a drop experiment can be used. The time between bounces on the fundament serves as a measure for that coefficient. In this thesis, a method to numerically model and simulate the physical experiment was developed. The approach used is based on the molecular dynamics method which is able to simulate large particle numbers. The suggested method inherits the second-order accuracy of that approach. For the model of the particles, pseudo-friction was implemented as well as a moving, deformable simulation volume with a coherent potential-based approach for all interactions. With this framework, it is now possible to simulate and test a large number of particles inside a hollow sphere within a moderate amount of computing time. Different particle shapes and sizes can be included in the simulation, and the interaction potential can easily be modified or replaced to suit the need of the simulation, e.g., to simulate different materials. The simulation was initially developed and tested using a straightforward implementation with a double loop for the interaction checks. However, the quadratic nature of that algorithm makes the computation time of larger systems impractical. The linked cell method on the other hand showed a superior performance compared to the straightforward implementation and is an obvious choice for the considered scenario that uses up to around 10^5 particles. For this purpose, a flexible memory structure using pointers was implemented for the linked cell approach. Organizing the particles this way, the simulation is able to deal with the agglomeration effects that occur because of the gravity acting on the system as well as the changing distribution of the particles inside the volume of the sphere as the experiment commences.

For the first time, the method developed here is able to fulfill the require-

ments of the simulation of the considered scenario. Notably, a range of valid step sizes was identified to satisfy the requirement of numerical stability and energy conservation. While the straightforward approach is too slow for a large number of particles, it still has its uses in parameter training as it returns results quickly for smaller systems containing just a few particles. The numerical experiments support the findings of the physical experiments, with a clear damping visible as the number of particles grows. However, due to the heterogeneity of the physical particle mix, containing a myriad of different shapes and sizes, results remain a coarse approximation of the real world experiment. Upon closer inspection, Fraunhofer IFAM Dresden suspects the particle number to reach many millions and even up to a billion, with the material breaking up during the experiment, with most particles extremely small. Simulating such a system in very fine detail remains a task for future computers.

Future developments

The method and software developed in this work provides a versatile and powerful tool for conducting the drop experiment numerically. To simulate the lightweight material on a larger scale, many particle-filled spheres could be simulated together, e.g. using bonds as discussed in chapter 6. To comprehensibly simulate the composite material, thousands of spheres could be coupled running in parallel on a large computer system. Challenges that arise there include proper coupling of several spheres with each other and with other structures in the material, and with that a greater flexibility in deformation of the reflective layer. The coupling will have to be versatile to allow for the different manufacturing models used, such as glue, solder or the embedding in a resin matrix.

A possibility for that would be to revisit the phantom particle boundary discussed in chapter 3 to build arbitrary walls and thus differently shaped enclosures. The phantom particles would need to use some kind of exponential barrier potential to prevent hull breaches when particles accumulate near the reflective layer.

Additionally, moving away from the detailed simulation, it would be desirable to derive meta-models that approximate the damping effect caused by the particles, as discussed in the previous chapter. These would be able to help in the design and engineering process for new applications of the lightweight material on a broader scale.

Another challenge that remains is the analysis of new friction models. While there are highly detailed contact models available, they are not really applicable for large systems on today's computer hardware. If the computation of a single sophisticated interaction takes a few seconds or even minutes, a simulation with thousands of simultaneous contacts in each iteration will not finish in any reasonable timeframe. Thus, the potential based approach is very useful to model interactions of bodies. The present software has been built to allow for a quick exchange of the interaction potentials, so experiments can quickly be modeled in a completely different manner when another friction approach is to be studied.

In industrial applications, an optimization is always desirable. In the future, the framework presented here could be expanded by a coupled optimization to identify the most relevant design parameters in the lightweight material. These parameters will have to follow the requirements regarding solvent and temperature resistance and many others, so that a close cooperation with mechanical engineers is paramount for the success of such optimization efforts.

The linked cell algorithm presented here could potentially see some improvements in the workload caused by its structure. As many cells are empty periodically, it should be possible to only save the cells that have particles inside which would decrease the memory footprint. However, balancing performance with management overhead will always play a role with these kind of simulations.

Another improvement that can be made is deploying sophisticated packing strategies for a more efficient initial configuration of the particles inside the sphere. This is especially important for more complex particles or when dealing with mixtures. Since the current configuration builds upon the largest cut-off radius of any particle present in the simulation, a higher number of particles could be fitted inside the hollow sphere when smaller particles are packed more efficiently.

Ultimately, discrete element simulations have always come down to raw computing power. They have been highly dependent on the advances in computing technology. Only with today's computing power of supercomputers, is it possible to arrive at the threshold to real practical simulations. Some of the fastest machines can now simulate a tiny but macroscopic drop of liquid. It will be interesting to see the development that this field will take in the future.

Appendix A

Additional simulation data

This appendix complements the results obtained in Chapter 5.

Simulation 1: Damping behavior

N	ΔT in ms	N	ΔT in ms
10000	198.40	80000	96.20
20000	184.44	90000	86.84
30000	166.91	100000	78.70
40000	148.97	110000	71.65
50000	132.98	120000	65.53
60000	118.95	130000	60.44
70000	106.83		

Table A.1: Rebound time for an increasing number of particles

Simulation 2: Time step

N	$\Delta t = 0.01$ μs	$\Delta t = 0.05$ μs	$\Delta t = 0.1$ μs	$\Delta t = 1$ μs	$\Delta t = 2$ μs	$\Delta t = 5$ μs
50	34.58	34.56	34.69	34.69	34.69	32.03
100	34.54	34.56	34.41	32.04	29.36	32.02
250	34.56	34.56	34.71	34.66	34.70	32.01
500	34.55	34.57	34.65	34.66	34.68	45.25
750	34.53	34.47	34.37	34.54	34.67	31.95
1000	34.47	34.52	34.57	34.56	34.58	31.98
1500	34.42	34.36	34.47	34.52	34.45	31.88
2000	34.38	34.40	34.16	34.62	34.35	-
2500	34.25	34.26	34.44	34.46	34.46	44.88
5000	34.16	34.10	34.10	34.35	34.50	31.58
7500	34.07	34.05	34.15	34.33	34.23	-
10000	33.61	33.58	33.43	34.02	33.68	-
12500	33.12	33.11	33.07	32.92	32.59	-
15000	32.67	32.70	32.74	32.41	33.05	-
17500	32.17	32.10	32.07	31.97	32.25	-
20000	31.74	31.74	31.73	31.62	31.78	-

Table A.2: Rebound time (in ms) for varying time steps for the drop experiment from height $h_d = 1.5$ mm

Simulation 3: Particle size

mass in mg	$\sigma = 0.31 \cdot 10^{-4}$ m	$\sigma = 0.62 \cdot 10^{-4}$ m	$\sigma = 0.93 \cdot 10^{-4}$ m
0.123	205.64	205.91	205.30
0.246	202.62	202.23	201.66
0.369	200.52	198.15	197.42
0.492	194.81	194.64	193.11
0.615	186.10	190.75	188.80
0.738	176.99	186.44	184.35
0.861	168.28	180.37	179.98
0.984	160.46	175.58	175.37
1.107	153.16	169.84	170.59

Table A.3: Rebound time (in ms) for varying particle diameters and particle mass

Simulation 4: Impact of the friction parameters

σ in 10^{-4} m	N	mass in 10^{-5} mg
0.317	102326	4.404
0.400	51163	8.808
0.504	25575	17.620
0.635	12791	35.230
0.800	6395	70.464
1.008	3216	140.100
1.270	1598	281.900
1.600	799	563.700
2.016	399	1127.400

Table A.4: Number of particles, diameter and mass for the friction experiment

σ in 10^{-4} m	P=.05,k=.02	P=.1,k=.02	P=.15,k=.02	P=.1,k=.01	P=.1, k=.05	P=.1,k=.1
0.317	72.29	56.72	46.49	56.74	56.68	56.57
0.4	77.26	62.66	52.71	62.70	62.55	62.41
0.504	81.97	68.18	58.87	68.25	68.12	67.90
0.635	86.11	73.13	64.63	73.26	73.07	72.70
0.8	89.56	77.64	69.54	77.72	77.34	76.97
1.008	91.91	80.98	73.15	81.01	80.59	79.86
1.27	93.96	84.17	76.88	84.31	83.45	82.56
1.6	95.29	86.69	80.08	86.95	85.92	84.62
2.016	95.13	88.39	82.59	88.69	87.69	86.39

Table A.5: Rebound time (in ms) for varying friction parameters

Simulation 5: Particle shape

type	body-fixed coordinates (in 10^{-4} m)
L	(0,0,0), (0,0.2,0), (0,0.4,0), (0.2,0,0), (0.4,0,0)
V	(0,0,0), (0.2,0,0), (0.4,0,0), (0.1414,0.1414,0), (0.2828,0.2828,0)
I	(0,0,0), (0,0.2,0), (0,0.4,0), (0,0.6,0), (0,0.8,0)
O	(0,0,0), ($\pm$ 0.1414,$\pm$ 0.1414,0)
T	(0,0,0), (0,0.2,0), (0,0.4,0), (0,0.2,0.2), (0,0.2,0.4)

Table A.6: Definition of particle types composed of five spheres

Simulation 6: Vibration of the sphere

	frequency in Hz				
N	0	100	200	300	400
10000	199.19	199.46	198.36	197.94	197.82
20000	171.43	171.92	170.64	170.52	170.42
30000	146.44	146.81	146.01	145.63	145.69
40000	127.13	127.43	126.63	126.56	126.40
50000	111.50	111.60	111.16	111.03	110.97
60000	98.43	98.59	98.23	98.14	98.14
70000	87.38	87.52	87.30	87.21	87.24
80000	77.91	78.05	77.68	77.76	77.70
90000	69.57	69.70	69.48	69.48	69.48
100000	62.31	62.43	62.21	62.23	62.27
110000	56.02	56.14	55.96	56.00	55.95
120000	50.68	50.77	50.63	50.64	50.61
130000	46.22	46.28	46.14	46.13	46.12

Table A.7: Rebound time (in ms) for varying vibration

References

[1] B.J. Alder and T.E. Wainwright. Studies in molecular dynamics. *The Journal of Chemical Physics*, 31:459–466, 1959.

[2] M.P. Allen and D.J. Tildesley. *Computer Simulation of Liquids*. Oxford University Press, New York, 1990.

[3] T. Bajd, M. Mihelj, and M. Munih. Rotation and orientation. In *Introduction to Robotics*. Springer Netherlands, 2013.

[4] M.F. Beatty. *Principles of Engineering Mechanics, Volume 2, Dynamics - The Analysis of Motion*. Springer New York, 2006.

[5] D. Blase. *Simulation partikelgefüllter Hohlkugeln in zwei Raumdimensionen*. Diplomarbeit, Technische Universität Dresden, 2008.

[6] P.A. Cundall. A computer model for simulating progressive, large-scale movements in blocky rock systems. In *Rock fracture: Proceedings of the International Symposium on Rock Mechanics*, pages II–8. International Society for Rock Mechanics, Nancy, 1971.

[7] P.A. Cundall. *The measurement and analysis of accelerations in rock slopes*. PhD thesis, Imperial College, London, 1971.

[8] G. Dahlquist and Å. Björck. *Numerical methods*. Prentice-Hall, Englewood Cliffs, N.J., 1974.

[9] I. Djurdjev, R. Filimonov, M. Hjälle, A. Malyutina, and S. Zhu. Modeling the coupling of spheres in a drop experiment. Technical report, Universität Paderborn, 2014.

[10] D. Fincham. Leapfrog rotational algorithms. *Molecular Simulation*, 8:165–178, 1992.

[11] K.-F. Fischer, editor. *Taschenbuch der technischen Formeln*. Fachbuchverlag Leipzig, 1999.

[12] D. Frenkel and B. Smit. *Understanding Molecular Simulation*. Academic Press, San Diego, 2002.

[13] H. Goldstein. *Klassische Mechanik*. Akademische Verlagsgesellschaft, Frankfurt am Main, 1972.

[14] M. Griebel. *Numerische Simulation in der Moleküldynamik*. Springer, Berlin, 2005.

[15] R. Haberlandt, S. Fritzsche, G. Peinel, and K. Heinzinger. *Molekulardynamik*. Vieweg, Braunschweig, 1995.

[16] J.M. Haile. *Molecular Dynamics Simulation, Elementary Methods*. John Wiley & Sons, New York, 1992.

[17] E. Hairer, C. Lubich, and G. Wanner. *Geometric Numerical Integration*. Springer, Berlin Heidelberg, 2006.

[18] A. Heinecke, R. Bader, M. Brehm, N. Hammer, H. Huber, H.-G. Kleinhenz, J. Vrabec, H. Hasse, M. Horsch, M. Bernreuther, C. W. Glass, C. Niethammer, A. Bode, H.-J. Bungartz, and W. Eckhardt. 591 TFLOPS Multi-trillion Particles Simulation on Super MUC. *Lecture Notes in Computer Science*, 7905:1–12, 2013.

[19] P.H. Hünenberger. Thermostat algorithms for molecular dynamics simulations. *Advances in Polymer Science*, 173:105–149, 2005.

[20] U. Jehring, H. Göhler, P. Quadbeck, J. Meinert, R. Hauser, G. Stephani, and B. Kieback. Metallische Hohlkugelstrukturen - Systemleichtbau durch gezieltes Werkstoffdesign. In *LLC Proceedings*, pages 375–388. 6. Landshuter Leichtbau-Colloquium, 2013.

[21] B. Kieback, G. Stephani, P. Quadbeck, J. Courtois, K. Hahn, D. Blase, A. Walther, and U. Jehring. Lightweight-materials made from particle filled metal hollow spheres. In *Porous Metals and Metallic Foams: MetFoam 2009*, 2009.

[22] J.B. Kuipers. *Quaternions and Rotation Sequences*. Princeton University Press, Princeton, 1999.

[23] H. Meyer, G. Schumpich, and G. Holzmann. *Technische Mechanik, Teil 2, Kinematik und Kinetik*. B.G. Teubner, Stuttgart, 2000.

[24] J.J. Monaghan. Smoothed particle hydrodynamics. *Annual Review of Astronomy and Astrophysics*, 20:543–574, 1992.

[25] J.J. Monaghan and J.C. Lattanzio. A refined particle method for astrophysical problems. *Astronomy and Astrophysics*, 149:135–143, 1985.

[26] A. Perez and J. M. McCarthy. Dual quaternion synthesis of constrained robotic systems. *Journal of Mechanical Design*, 126:425–435, 2004.

[27] D.C. Rapaport. *The art of molecular dynamics simulation*. Cambridge University Press, Cambridge, 2007.

[28] M. Sánchez and L.A. Pugnaloni. Effective mass overshoot in single degree of freedom mechanical systems with a particle damper. *Journal of Sound and Vibration*, 330:5812–5819, 2011.

[29] F. Scheck. *Mechanik*. Springer, Berlin, 1996.

[30] B. Siciliano and O. Khatib, editors. *Springer Handbook of Robotics*. Springer, Berlin, 2008.

[31] T. Steinle, J. Vrabec, and A. Walther. Numerical simulation of the damping behavior of particle-filled hollow spheres. In H. G. Bock, X. P. Hoang, R. Rannacher, and J. P. Schlöder, editors, *Modeling, Simulation and Optimization of Complex Processes - HPSC 2012*, pages 233–243. Springer, Cham, 2014.

[32] A. Tasora and M. Anitescu. A convex complementarity approach for simulating large granular flows. *Journal of Computational and Nonlinear Dynamics*, 5:1–10, 2010.

[33] H. Vehkamäki. *Classical Nucleation Theory in Multicomponent Systems*. Springer, Berlin Heidelberg, 2006.

Selbständigkeitserklärung

Ich erkläre hiermit, dass ich die vorliegende Arbeit selbständig und nur unter Verwendung der angegebenen Literatur und Hilfsmittel angefertigt habe.

Paderborn, den 28. Oktober 2015 Tobias Steinle