RUS
Rechenzentrum
Universität Stuttgart

ICA I
Institut für
Computeranwendungen
Abt. Physik mit Hochleistungsrechnern

AIM
Anwendungen der Informatik
im Maschinenwesen

FORSCHUNGS - UND ENTWICKLUNGSBERICHTE

FAST ALGORITHMS FOR THE SIMULATION OF GRANULAR PARTICLES

Matthias S. Müller

ISSN 0941 - 4665 Juni 2001 RUS - 52

Rechenzentrum
Universität Stuttgart
Prof. Dr.-Ing. R. Rühle
Allmandring 30, 70550 Stuttgart

Institut für
Computeranwendungen
Abt. Physik mit Hochleistungsrechnern

Anwendungen der Informatik
im Maschinenwesen

FAST ALGORITHMS FOR THE SIMULATION OF GRANULAR PARTICLES

von der Fakultät Physik der Universität Stuttgart
zur Erlangung der Würde eines Doktors der Naturwissenschaften (Dr. rer. nat)
genehmigte Abhandlung

vorgelegt von

Matthias S. Müller

aus Sindelfingen

Hauptberichter:	Prof. Dr. H. J. Herrmann
Mitberichter:	Prof. Dr. H.-R. Trebin
Tag der Einreichung:	02. April 2001
Tag der mündlichen Prüfung:	28. Juni 2001

ISSN 0941 - 4665 Juni 2001 RUS - 52

To my mother and my father.

Contents

1. Introduction

Granular materials are ubiquitous in nature, industrial processing and everyday life. Examples range from small scale particles in dust, graphite powder, cement or flour over medium-sized sand grains, plastic granulates and cornflakes to the planetary rings on the astrophysical scale. Similarly broad are the physical phenomena controlling their behavior in transport, storage and processing.

In the last years, granular media have attracted a lot of attention [80]. Dissipative, many-particle systems far from equilibrium are one possible description of this fascinating granular "state of matter". Classical kinetic theories have been extended to account for dissipation and higher densities (see for example refs. [14, 21] and references therein). However, most of the classical and advanced approaches for a theoretical description of rapid granular flow are based on the assumption of molecular chaos – the assumption that the velocities and positions of all colliding pairs of particles in a gas are uncorrelated. In a gas, the errors introduced by this assumption are small. In dense granular flows, correlations between colliding particles may be important, leading to qualitative changes in behavior.

Other continuum-mechanical descriptions are describing granular materials for specific flows like static heaps [127], surface flow [12] or heap formation [35]. Despite their success in describing these flows, they are not applicable to general flows.

However, despite their importance, continuum or other large-scale modeling still shows severe deficiencies and our understanding of the mesoscopic physics in these systems, as exemplified by fragmentation, dissipative effects, sound propagation, etc. is incomplete since many theoretical methods otherwise applicable to many-particle systems do not apply. Therefore, large-scale computation is sometimes the only way to deepen our insight. The reason is that typical granular systems consist of millions of particles and most phenomena are only visible on long time scales.

For many years discrete methods have been used where the granular material is treated as an assemblage of particles. This method was introduced by Cundall to simulate the motion of rock masses [20]. The so called Molecular Dynamics (MD) has been applied to a wide range of flows and phenomena. Examples are size segregation [104], density waves in pipe flow [43, 103], pressure distribution in heaps [73] and shear flow [16]. Variants of MD that incorporate static friction have been used to examine the details of force distributions in granular assemblies [106, 107]. One problem of MD is the huge amount of computer time that is required to simulate large systems. Many time steps are necessary to cover long time scales when stiff particles are involved. Since in many cases it is justified to model the granular particles as infinitely rigid spheres, fast algo-

rithms for Hard Sphere Molecular Dynamics have been proposed [62] and applied to fields like vibrated granular materials [70], dense pipe flow [69] or freely evolving dissipative particles[77]. A problem of this algorithm is the so called "inelastic collapse"[76, 78]: the number of collisions per unit time may diverge and this makes it impossible to integrate the further evolution of the system. So far, there is also no efficient parallel version which limits the method to small and moderate particle numbers. Further simplifications are introduced by the lattice gas automaton (LGA) models. Here, the particles are located at the vertices of a two dimensional triangular lattice. LGA simulations [105, 122] have been applied to simulate pipe flow [97], heap formation [7] and outflow of a hopper [8, 9]. The simplicity of the model allows a simulation to follow a large number of particles. On the other hand, since the movement of particles is fixed to the lattice the isotropy and homogeneity of space is not maintained. Care has to be taken that the resulting artifacts do not limit the validity of results. Since all methods have their specific advantages and disadvantages there is clearly the need for methods that can efficiently simulate large particle numbers and can be applied to a wide range of granular flows.

A common approach to develop a computer simulation consists of several subsequent steps. First a model has to be developed that represents the physical system of interest. Most of the time this model consists of differential equations describing the dynamic behavior. In a second step an appropriate numerical scheme is required to solve these equations. This scheme has to be implemented, normally in a high level programming language like Fortran or C. A more direct approach is to use a numerical program like `matlab` that offers implementations for most standard numerical schemes. If the problems require large-scale computation a final step is the parallelization of the program to make use of high-end massive parallel or parallel vector computers. At the end the resulting program has of course to be verified for correctness. The task of this thesis included all of the described phases. The idea was to model the physical system from the very beginning in a way that makes it suitable for fast parallel computation. This approach comprises the five steps described above into one design process. The name *Hybrid Simulation Monte Carlo* or HSMC was selected, because the resulting algorithm combines several elements of different methods. The abreviation HSMC also resembles DSMC (Direct Simulation Monte Carlo), the method that served as a starting point in the development of the algorithm. The name HSMC should not be confused with hybrid Monte Carlo simulation [125], which is a general term for methods that combine Monte Carlo elements with other approaches.

A crucial part of this work was the selection of appropriate test cases that could first serve as a guideline in the development of the algorithm and later be used as verification. Homogeneous cooling was selected as first test case, because it treats the most basic phenomenon: dissipation. The correct behavior of HSMC is verified by comparing with theoretical results. The next case already shows the demand for large-scale computation. The studied clustering instabilities occurs only in systems that are large enough. Due to the lack of sufficient theoretical knowledge the results of HSMC are compared with results of event driven simulations. In both systems the granular particles are freely evolving, without external forces. In the next two cases driven granular flows are treated. Here energy is brought into the system either by the walls or by gravity

acting on the particles. The next system that is considered is even more difficult to treat with a stochastic method like HSMC. So far, the systems have all been kept dynamic, either freely evolving or kept alive by a continuous supply of energy. In the case of heaps however, the system comes to rest in a well defined state. The heap does not only show a well defined angle of repose, but also small deviations at its tail. Because a transition from dynamics to statics is typical for granular systems the correct treatment of this final test case was not only difficult, but also very important.

Chapter 3.6 looks at one of the assumptions HSMC is based upon: the correlations that exists between different particles in a dissipative system and to what extent they are reflected by HSMC. The section about vibrated beds is an additional case study where the results of the stochastic DSMC are compared with results from hard sphere molecular dynamics. This does not only show once again the correctness of the method, but also answers the question whether correlations between particles in a narrow two dimensional systems are important for the correct scaling behavior.

The structure of this thesis is as follows: at the beginning there is a short introduction into the theory that is the basis of the algorithm that was developed in this thesis. Afterwards this simulation algorithm is presented. Since it is based on the Direct Simulation Monte Carlo method (DSMC), several variations that exist in literature are described in the next chapter. One important issue before and during the course of developing a simulation method is to learn from other simulation methods. Chapter 2.5 shortly describes some methods that are used to simulate granular matter and were candidates for fast parallel methods. The next chapters treat the phenomena that were chosen to be representative for granular matter.

The last two chapters treat computational issues. Although HSMC was designed to run efficiently on parallel supercomputers, the ever growing demand for larger and more precise simulations asks for coupling these supercomputers. This so called metacomputing poses new threads to the algorithms used in this field. Chapter 4.1 describes the experiences made in several testbeds and why DSMC and HSMC are well suited for metacomputing. Finally chapter 4.2 talks about the implementation of the algorithm. There are several reasons to talk about the implementation. First it is an important issue for the performance of the resulting program, second it supplies another viewpoint of the algorithm and finally the program was implemented in C++. This is still uncommon in the scientific community where due to performance considerations Fortran and C are still the most used languages.

2. Theory and Models

2.1. Statistical Mechanics

The size of granular particles range from a fraction of a millimeter in dust to kilometers in planetary rings. In experiments a small setup contains thousands of particles. Industrial systems are much larger and consist of millions of particles. Due to the size and numbers of particles granular systems are subject to be described with classical statistical mechanics.

All systems in this thesis consist of N identical particles. Each particle has d translational degrees of freedom and is described by its position $\mathbf{x}$ and momentum $\mathbf{p} = m\mathbf{v}$. The whole system is completely specified by its dN coordinates $\mathbf{x}^N \equiv \{\mathbf{x}_1, ..., \mathbf{x}_N\}$ and dN momenta $\mathbf{p}^N \equiv \{\mathbf{p}_1, ..., \mathbf{p}_N\}$. These values determine a point in the $2dN$ dimensional phase space.

The motion of the system in the phase space is described by the Hamilton equation

$$\dot{\mathbf{x}}_i = \frac{\partial H}{\partial \mathbf{p}_i} \tag{2.1}$$

$$\dot{\mathbf{p}}_i = -\frac{\partial H}{\partial \mathbf{x}_i}, \tag{2.2}$$

where H is the Hamiltonian of the system:

$$H_N(\mathbf{x}^N, \mathbf{p}^N) = \sum_{i=1}^{N} \frac{1}{2m_i}|\mathbf{p}_i|^2 + V_N(\mathbf{x}^N). \tag{2.3}$$

The solution is determined by the $2dN$ initial condition on the coordinates and momenta. The aim of statistical mechanics is not to find a solution for a single initial condition, but to calculate observable properties of the system as either time-average or ensemble average. The later is called Gibbs' method. In Gibbs' formulation the distribution of an ensemble is described by a phase-space probability density $f^{(N)}(\mathbf{x}^N, \mathbf{v}^N)$. The time evolution of f is governed by the Liouville equation

$$\frac{\partial f^{(N)}}{\partial t} = \sum_{i=1}^{N} \left(\frac{\partial H_N}{\partial \mathbf{x}_i}\frac{\partial f^{(N)}}{\partial \mathbf{p}_i} - \frac{\partial H_N}{\partial \mathbf{p}_i}\frac{\partial f^{(N)}}{\partial \mathbf{x}_i} \right). \tag{2.4}$$

The full phase space probability density $f^{(N)}$ provides a very detailed description of the system. For many purposes it is sufficient to describe only a subset of n particles.

The unnecessary information can be eliminated by integrating $f^{(N)}$ over the coordinates and momentum of the remaining $N-n$ particles, resulting in a reduced phase-space distribution function $f^{(n)}$, which is defined as

$$f^{(n)}(\mathbf{x}^n, \mathbf{p}^n, t) = \frac{N!}{(N-n)!} \int \int f^{(N)}(\mathbf{x}^N, \mathbf{p}^N, t) d\mathbf{x}^{(N-n)} d\mathbf{p}^{(N-n)}. \tag{2.5}$$

The factor $\frac{N!}{(N-n)!}$ is the number of different ways to choose n particles out of N.

If the interaction between the particles is limited to a pairwise interaction, the total force $\mathbf{F}_i$ acting on a particle i can be written as sum of an external Force $\mathbf{F}_i^{ext}$ and the pair forces $\mathbf{F}_{ij}$. The Liouville equation then becomes

$$\frac{\partial f^{(N)}}{\partial t} + \frac{1}{m} \sum_{i=1}^{N} \mathbf{p}_i \frac{\partial f^{(N)}}{\partial \mathbf{x}_i} + \sum_{i=1}^{N} \mathbf{F}_i^{ext} \frac{\partial f^{(N)}}{\partial \mathbf{p}_i} = \sum_{i=1}^{N} \sum_{j=1}^{N} \mathbf{F}_{ij} \frac{\partial f^{(N)}}{\partial \mathbf{p}_i}. \tag{2.6}$$

We can now integrate over $(N-n)$ positions and momenta and insert definition (2.5). By exploiting the symmetry of $f^{(N)}$ with respect to interchange of particle labels and the fact that $f^{(N)}$ vanishes when either $\mathbf{p}_i \to \infty$ or $\mathbf{x}_i$ lies outside the volume occupied by the system, we find that [38]

$$\begin{aligned}
&\frac{\partial f^{(N)}}{\partial t} + \frac{1}{m} \sum_{i=1}^{N} \mathbf{p}_i \frac{\partial f^{(N)}}{\partial \mathbf{x}_i} + \sum_{i=1}^{N} \mathbf{F}_i^{ext} \frac{\partial f^{(N)}}{\partial \mathbf{p}_i} \\
&= -\frac{N!}{(N-n)!} \sum_{i=1}^{N} \sum_{j=1}^{N} \int \int \mathbf{F}_{ij} \frac{\partial f^{(N)}}{\partial \mathbf{p}_i} d\mathbf{x}^{(N-n)} d\mathbf{p}^{(N-n)} \\
&= -\sum_{i=1}^{n} \sum_{j=1}^{n} \mathbf{F}_{ij} - \frac{N!}{(N-n)!} \sum_{i=1}^{n} \sum_{j=n+1}^{N} \int \int \mathbf{F}_{ij} \frac{\partial^{f(N)}}{\partial \mathbf{p}_i} d\mathbf{x}^{(N-n)} d\mathbf{p}^{(N-n)} \\
&= -\sum_{i=1}^{n} \sum_{j=1}^{n} \mathbf{F}_{ij} - \sum_{i=1}^{n} \int \int \mathbf{F}_{i,n+1} \frac{\partial f^{(n+1)}}{\partial \mathbf{p}_i} d\mathbf{x}_{n+1} d\mathbf{p}_{n+1}.
\end{aligned} \tag{2.7}$$

Collecting the terms with $f^{(n)}$ on the left hand side shows, that $f^{(n)}$ and $f^{(n+1)}$ are linked by the formula

$$\begin{aligned}
&\left\{ \frac{\partial}{\partial t} + \sum_{i=1}^{n} \left[\frac{1}{m} \mathbf{p}_i \frac{\partial}{\partial \mathbf{x}_i} + \left(\mathbf{F}_i^{ext} + \sum_{j=1}^{n} \mathbf{F}_{ij} \right) \frac{\partial}{\partial \mathbf{p}_i} \right] \right\} f^{(n)}(\mathbf{x}^n, \mathbf{p}^n, t) \\
&= -\sum_{i=1}^{n} \int \int \mathbf{F}_{i,n+1} \frac{\partial}{\partial \mathbf{p}_i} f^{(n+1)}(\mathbf{x}^{(n+1)}, \mathbf{p}^{(n+1)}, t) d\mathbf{x}_{n+1} d\mathbf{p}_{n+1}.
\end{aligned} \tag{2.8}$$

This set of equations is known as the BBGKY hierarchy (Bogolyubov, Born, Green, Kirkwood, Yvon). This hierarchy cannot be treated for any realistic particle number N, therefore at some point an approximation has to be made. An equation for the one

particle distribution function $f^{(1)}$ is obtained by setting $n = 1$ in equation (2.8):

$$\begin{aligned}&\left(\frac{\partial}{\partial t} + \frac{1}{m}\mathbf{p}_1\frac{\partial}{\partial \mathbf{x}_1} + \mathbf{F}_1^{ext}\frac{\partial}{\partial \mathbf{p}_1}\right) f^{(1)}(\mathbf{x}_1, \mathbf{p}_1, t) \\ &= \int\int \mathbf{F}_{12}\frac{\partial}{\partial \mathbf{p}_1} f^{(2)}(\mathbf{x}_1, \mathbf{p}_1, \mathbf{x}_2, \mathbf{p}_2, t) d\mathbf{x}_2 d\mathbf{p}_2.\end{aligned} \tag{2.9}$$

The problem of this equation is, that you need to know the pair distribution function $f^{(2)}$ to solve it. The simplest approximation is to ignore the pair correlation by writing

$$f^{(2)}(\mathbf{x}_1, \mathbf{p}_1, \mathbf{x}_2, \mathbf{p}_2) = f^{(1)}(\mathbf{x}_1, \mathbf{p}_1) f^{(1)}(\mathbf{x}_2, \mathbf{p}_2). \tag{2.10}$$

This approximation is also called the Molecular Chaos Assumption, because it assumes that the position and velocities of two particles are uncorrelated. Combining (2.9) and (2.10) leads to the Vlasov equation:

$$\left(\frac{\partial}{\partial t} + \frac{1}{m}\mathbf{p}_1\frac{\partial}{\partial \mathbf{x}_1} + \mathbf{F}_1^{ext}\frac{\partial}{\partial \mathbf{p}_1}\right) f^{(1)}(\mathbf{x}_1, \mathbf{p}_1, t) = -\bar{\mathbf{F}}(\mathbf{x}, t), \tag{2.11}$$

where

$$\bar{\mathbf{F}}(\mathbf{x}, t) = \int\int \mathbf{F}(\mathbf{x}, \mathbf{x}_2) f^{(1)}(\mathbf{x}_2, \mathbf{p}_2, t) d\mathbf{x}_2 d\mathbf{p}_2 \tag{2.12}$$

is the average force from other particles in the system acting on a particle at position $\mathbf{x}$ at time t.

2.2. Boltzmann Equation

If the mean force $\bar{\mathbf{F}}(\mathbf{x}, t)$ is caused by collisions with other particles, the Vlasov equation (2.11) results in the Boltzmann equation (2.13).

In the framework of the Boltzmann equation the system is described with the single particle distribution function f, where $f(\mathbf{x}, \mathbf{v}, t)$ describes the probability density to find a particle at position $\mathbf{x}$, velocity $\mathbf{v}$ and time t.

$$\frac{\partial f(\mathbf{x}, \mathbf{v}, t)}{\partial t} = -\underbrace{\mathbf{v}\nabla_{\mathbf{r}} f}_{\text{flow}} - \underbrace{\frac{\mathbf{F}}{m}\nabla_{\mathbf{v}} f}_{\text{external force}} + \underbrace{\left(\frac{\partial f}{\partial t}\right)_{\text{coll}}}_{\text{collisions}} \tag{2.13}$$

with

$$\left(\frac{\partial f}{\partial t}\right)_{\text{coll}} = \int d\mathbf{x}' \int d\mathbf{v}' \frac{\partial U(\mathbf{x}, \mathbf{x}')}{\partial \mathbf{x}} \frac{\partial}{\partial \mathbf{v}} f_2(\mathbf{x}, \mathbf{v}, \mathbf{x}', \mathbf{v}') \tag{2.14}$$

f_2 is the two particle distribution function. Writing down equations for f_2 leads to formulas involving three particle distribution functions. In general, the equation for

the n particle distribution functions contains the n+1 particle distribution function. This results in the so called BBGKY hierarchy of equations [39]. A closure is needed to end this infinite recursion. The simplest approach is to replace the two particle distribution function $f_2(\mathbf{x}, \mathbf{v}, \mathbf{x}', \mathbf{v}')$ with the product of two single particle distribution functions $f_1(\mathbf{x}, \mathbf{v}) f_1(\mathbf{x}', \mathbf{v}')$. This is the molecular chaos assumption, because $f_2 = f_1 f_1$ holds if the movement of two particles is completely uncorrelated. With this so called Stosszahlansatz the collision term can be written as

$$\left(\frac{\partial f(\mathbf{x}, \mathbf{v}_1, t)}{\partial t}\right)_{\text{coll}} = \frac{\sigma_T}{2} \int\limits_{\hat{\mathbf{k}}\mathbf{v}_{12}>0} d\hat{\mathbf{k}} d\mathbf{v}_2 (\hat{\mathbf{k}}\mathbf{v}_{12}) \left(\underbrace{f(\mathbf{v}_1^*) f(\mathbf{v}_2^*)}_{\text{gain}} - \underbrace{f(\mathbf{v}_1) f(\mathbf{v}_2)}_{\text{loss}} \right). \tag{2.15}$$

In equation (2.15) σ_T denotes the scattering cross section, $\hat{\mathbf{k}}$ the direction of momentum transfer and $\mathbf{v}_i^*$ are the post collision velocities.

$$\begin{aligned} \frac{\partial f(\mathbf{x}, \mathbf{v}, t)}{\partial t} &= -\left(\mathbf{v}\nabla_{\mathbf{r}} + \frac{\mathbf{F}}{m}\nabla_{\mathbf{v}}\right) f + \left(\frac{\partial f}{\partial t}\right)_{\text{coll}} \\ &= -\hat{D} f + \hat{C} f \end{aligned} \tag{2.16}$$

Equation (2.16) describes the time evolution of the single particle distribution function f. The time evolution contains the changes due to advection $\hat{D}$ and collisions $\hat{C}$ between the particles.

Two examples where the Boltzmann equation has been applied to systems of industrial relevance are the flow of air around space vehicles in the upper atmosphere [117] and the flow around heads of hard disks [4]. Since in the the mean free path in these cases is of the same size than the typical length of the system, the flow cannot be described with the Navier-Stokes equation, instead the Boltzmann Equation is used.

2.2.1. Analytical Treatment of Boltzmanns Equation

The major problem of an analytical treatment is the nonlinear collision term in equation (2.15):

$$\left(\frac{\partial f(\mathbf{x}, \mathbf{v}_1, t)}{\partial t}\right)_{\text{coll}} = \frac{\sigma_T}{2} \int\limits_{\hat{\mathbf{k}}\mathbf{v}_{12}>0} d\hat{\mathbf{k}} d\mathbf{v}_2 (\hat{\mathbf{k}}\mathbf{v}_{12}) \left(\underbrace{f(\mathbf{v}_1`) f(\mathbf{v}_2`)}_{\text{gain}} - \underbrace{f(\mathbf{v}_1) f(\mathbf{v}_2)}_{\text{loss}} \right).$$

Several approximations are possible to get a linear equation:

1. If two different kinds of particles collide, e.g. electrons and protons, the particles are described by two different probability distribution functions F_{proton} and f_{electron}. The momentum of a proton remains almost unchanged when it collides with an

electron. With the further approximations that the electrons scatter only with protons the product ff in the integral of equation (2.15) is replaced by Ff, where F is constant. The resulting equation is thus linear in f.

2. If a known equilibrium distribution f_0 exists, the solution can be written as f_0 plus a small deviation Φ.

$$f = f_0(1 + \Phi)$$

$$\begin{aligned} f(\mathbf{v}_1)f(\mathbf{v}_2) &= f_0(\mathbf{v}_1)(1 + \Phi(\mathbf{v}_1))f_0(\mathbf{v}_2)(1 + \Phi(\mathbf{v}_2)) \\ &\approx f_0(\mathbf{v}_1)f_0(\mathbf{v}_2)(1 + \Phi(\mathbf{v}_1) + \Phi(\mathbf{v}_2)) \end{aligned}$$

If the deviation Φ is small, $\Phi(\mathbf{v}_1)\Phi(\mathbf{v}_2)$ can be neglected and the equation is linear in the new unknown Φ.

3. Another possible approximation is that the momentum transfer in a particle interaction is small. The post collision velocity v^* is expressed as the pre collision velocity v plus a small change ξ. The value of $f(v^*)$ is calculated with a Taylor expansion of f around the pre collision velocity.

$$\begin{aligned} \left(\frac{\partial f}{\partial t}\right)_{\text{coll}} &\sim \int_\sigma \int_{v_2} (f(v_2^*)f(v_1^*) - f(v_2)f(v_1)) \\ &\sim \int \int (f(v_1 - \xi)f(v_2 + \xi) - f(v_2)f(v_1)) \\ &\sim \int \int \sum_{n=0}^{\infty} \frac{\xi^n}{n!}\frac{\partial^n}{\partial v^n}\, f(v)|_{v_1}\, f(v)|_{v_2} - f(v_1)f(v_2) \\ &\sim \int \int \sum_{n=1}^{\infty} \frac{\xi^n}{n!}\frac{\partial^n}{\partial v^n}\, f(v)|_{v_1}\, f(v)|_{v_2} \\ &\approx \int \int \frac{\xi}{1}\frac{\partial}{\partial v}\, f(v)|_{v_1}\, f(v)|_{v_2}\, \frac{\xi^2}{2}\frac{\partial^2}{\partial v^2}\, f(v)|_{v_1}\, f(v)|_{v_2} \end{aligned}$$

This equation has now the form

$$\left(\frac{\partial f}{\partial t}\right)_{\text{coll}} = \frac{\partial}{\partial v} M_1 f(v) + \frac{1}{2}\frac{\partial^2}{\partial v^2} M_2 f(v),$$

with

$$M_i = \int \int \frac{\xi^n}{n!} f(v_2).$$

If we approximate further and calculate M_i with f_0 instead of f, M is constant. This results in the Fokker-Planck-Equation

$$\frac{\partial f}{\partial t} = A\frac{\partial f}{\partial v} + B\frac{\partial^2 f}{\partial v^2},$$

with $A = M_1$ and $B = \frac{1}{2}M_2$.

Unfortunately none of this ways is applicable to granular materials in general. First, they normally consist of particles with about the same size and mass. Thus, the first approach is not applicable. Due to the stiffness of the particles the momentum transfer in a single collision is of the order of the momentum itself, making the third approach invalid. The second approach does not work in general, because frequently no known equilibrium distribution exists. Also, the equilibrium of a undriven dissipative system contains no energy. This makes the equilibrium distribution function inappropriate for an expansion, because the deviation to the solution of the non-equilibrium state will be large.

2.2.2. Enskog Theory

With all the possible approximations for the Boltzmann equation we should not forget that already the Stosszahlansatz (2.15) contains huge simplifications. Besides the assumption of molecular chaos it is the fact, that the particles interact at the same place. If the size of the particles cannot be neglected this is not longer true, and the two single particle distributions functions of the interacting particles have to be evaluated at different locations $\mathbf{x}$ and $\mathbf{x}+\mathbf{D}$, where $\mathbf{D}$ is the distance between the two particles. This results in the collision term

$$\left(\frac{\partial f(\mathbf{x},\mathbf{v}_1,t)}{\partial t}\right)_{\text{coll}} = \frac{\sigma_T}{2}\int\limits_{\hat{\mathbf{D}}\mathbf{v}_{12}>0} d\hat{\mathbf{D}}d\mathbf{v}_2(\hat{\mathbf{D}}\mathbf{v}_{12})\left(\chi(\mathbf{x},\mathbf{x}-\mathbf{D})f(\mathbf{x},\mathbf{v}_1^*)f(\mathbf{x}-\mathbf{D},\mathbf{v}_2^*)\right.$$
$$\left. -\,\chi(\mathbf{x},\mathbf{x}+\mathbf{D})\,f(\mathbf{x},\mathbf{v}_1)f(\mathbf{x}+\mathbf{D},\mathbf{v}_2)\right).$$

Because the particles have a finite volume, the number of collisions is increased compared to the dilute gas limit. The so called Enskog factor χ takes this into account. It is the value of the pair correlation function at the point of contact. This is again a function of the two particle positions ($\chi(\mathbf{x},\mathbf{x}+\mathbf{D})$). If χ is taken to be the value of the equilibrium pair correlation function at contact of the density in the middle between the two particles ($\chi(\mathbf{x},\mathbf{x}+\mathbf{D})=\chi(n(\mathbf{x}+0.5\mathbf{D}))$), this is the so called Standard Enskog Theory(SET).

If $\chi(\mathbf{x},\mathbf{x}+\mathbf{D})$ is a **local** equilibrium pair correlation function of a **non**uniform state obtained by density functional theory, it is the so called Revised Enskog Theory (RET).

2.3. Hybrid Simulation Monte Carlo

Hybrid Simulation Monte Carlo or HSMC is the simulation method that was developed in this thesis for the fast simulation of granular materials. It should not only be able to simulate a wide range of phenomena as described in the chapters about the test cases, but should also be capable to simulate millions of particles in a reasonable time frame. Of course certain simplifications have to be made to obtain a method that needs to be an order of magnitude faster than traditional methods like molecular dynamics. The test cases were used to check whether this simplifications are justified and HSMC gives valid results. Direct Simulation Monte Carlo (DSMC), a particle method that was first

used to simulate rarefied gases [11], served as a starting point to develop the algorithm. DSMC was not chosen, because it was the best suitable method for granular material, but due to its computational advantages like speed and its suitability for a parallel implementation [47].

For several reasons particle methods like DSMC or SPH (smoothed particle hydrodynamics [81, 85]) are becoming more and more popular. Unlike finite element methods that could also be used to solve the underlying equations, they are not based on a grid. Therefore they are also called mesh-less methods and do not suffer from the anisotropy and other artifacts introduced by the mesh. As described below, DSMC is not really a mesh-less method, because the collisions are based on a grid, therefore there will be visible lattice artifacts for dense systems. Another advantage of particle methods is the treatment of boundary conditions. The straight forward treatment of free boundaries is one reason why SPH is successfully used in the calculation of astrophysical flows.

2.3.1. Numerical Treatment of Boltzmann Equation

For the numerical solution of Eq. (2.16) it has to be discretized in space and time. In a first step the continuum equation is discretized in time

$$\begin{aligned} f(t+\Delta t) &= f(t) + \left(\frac{\partial f(t)}{\partial t}\right)\Delta t + O(\Delta t^2) \\ &\approx f(t) + \left(\frac{\partial f(t)}{\partial t}\right)\Delta t \\ &\approx \left(1 - \Delta t\hat{D} + \Delta t\hat{C}\right) f(t). \end{aligned} \tag{2.17}$$

This is a Taylor series expansion to first order. It will only give a good approximation if Δt is small enough. In general this means that the time step has to be of the order of the mean time between collisions. Up to the order $O(\Delta t^2)$ a finite time step will result in a modification of the transport coefficients. Together with the fact that there is no numerical instability, it makes the method useful for the application of large time steps.

Because the advection and collision operator act simultaneously this equation is still difficult to treat with a numerical algorithm. The two processes can be decoupled with the so called operator splitting. It consists of the following approximation:

$$1 - \Delta t\hat{D} + \Delta t\hat{C} \approx (1 + \Delta t\hat{C})(1 - \Delta t\hat{D}) + O(\Delta t^2). \tag{2.18}$$

The time evolution operator is now split in a product. At first the advection $1-\Delta t\hat{D}$ acts on f and afterwards the collision operator $1 + \Delta t\hat{C}$

$$f(t+\Delta t) \approx \underbrace{(1 + \Delta t\hat{C})}_{\text{collisions}}\underbrace{(1 - \Delta t\hat{D})}_{\text{advection}} f. \tag{2.19}$$

The advection step can be solved with any integration scheme. For the collision step a space discretization is used by introducing cells in which the collisions take place (see Sec. 2.4.1 and 2.3.2).

2.3.2. Description of the HSMC algorithm

The major disadvantage of all DSMC variants described in chapter 2.4 is, that there is no limitation of the particle density. This is a consequence of the fact that during the advection step the particles move completely independently of each other. During the interaction step only stochastic collisions take place. One example where the straightforward application of DSMC will result in unrealistic high particle densities is a heap of dissipative particles. In such a heap the collisions need to be highly correlated in order to support the particles at the top. A further problem is, that with vanishing kinetic energy and therefore velocity the momentum transfer Δp in a collision also disappears. Because the force $F = \frac{\Delta p}{\Delta t}$ needs to balance gravity, the number of collisions per time diverges. The idea of HSMC is to change the advection step of the DSMC algorithm to achieve an excluded volume. A real excluded volume would however result in an algorithm similar to Molecular Dynamics or Event Driven, because every particle and its neighbors have to be inspected. We therefore check the constraints only on the coarse grained level of the cells that are used by the original DSMC to choose collision partners quickly. Whenever a particle tries to enter a different cell, the move is rejected if it would result in a physically unrealistic high density.

After the basic idea is clear we give now a detailed description of the algorithm.

Like with DSMC the system is integrated in discrete time steps τ. Each time step consists of two phases: the advection step, where the particles are moved, and the interaction step where the particles collide with each other.

Advection step: At each time step every particle is first moved, according to the equation of motion. Unlike in DSMC the particles do not move independently of each other, but interact via the cells they belong to. The particles are sorted into spatial cells. Every cell that contains enough particles that no additional particle can enter is marked as "full" (see Fig. 2.2). Whenever in subsequent steps a particle tries to enter a full cell, this move is rejected and the particle keeps its old position. In reality, the particle will enter the cell with volume fraction ν with a possibility $p(\nu)$. The probability will be zero for a density ν equal to the density ν_{dense} of a close packing of disks and one for an empty cell. In the HSMC algorithm this unknown function is approximated by a function that is one for $\nu < \nu_{max}$ and zero otherwise (see Fig. 2.1), ν_{max} was choosen to be equal to the density of a random close packing. As a possible improvement of HSMC this simple aproximation could be replaced by a more realistic distribution, which could be obtained from ED simulations or experiments.

So far the algorithm prevents unrealistic high densities but nevertheless yields unphysical results. Up to this point there is no interaction across cell boundaries, i.e. a particle trying to enter a full cell does not collide with particles inside this cell. To allow a momentum transfer across cell boundaries between full cells we perform collisions between these cells first. For the collisions the cells are treated like particles with mass

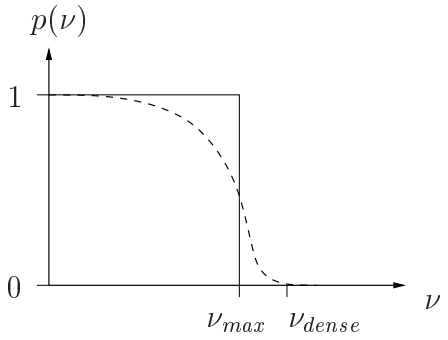


Figure 2.1.: Probability $p(\nu)$ of a particle to enter a cell with volume fraction ν. The unknown function (dashed line) is approximated with a heavyside function (solid line).

M_c and velocity $\mathbf{v}_c$.

$$M_c := \sum_i m_i \tag{2.20}$$

$$\mathbf{v}_c := \frac{1}{M_c} \sum_i m_i \mathbf{v}_i. \tag{2.21}$$

The definition of the cell velocity is chosen so that the total momentum of the particles is equal to the cell momentum $\mathbf{p}_c$:

$$\mathbf{p}_c = M_c \mathbf{v}_c = \frac{M_c}{M_c} \sum_i m_i \mathbf{v}_i = \sum_i \mathbf{p}_i \tag{2.22}$$

The collision partner (an adjacent cell) is not chosen randomly but the cell with the maximum relative velocity is picked (see Fig. 2.3). A checkerboard update is used to make the update independent of the order of execution on different processors on a parallel platform. Because every cell is either picked by the checkerboard update directly or a neighbor of such a cell, it is possible for every cell to participate in a collision. To avoid differences between "white" and "black" cells we exchange their role in the next time step.

The dissipation in the cell collisions is also governed by the coefficient of restitution. The dissipated energy is distributed among the particles that constitute the cell by increasing its "thermal energy" – the velocity relative to $\mathbf{v}_c$ is rescaled. With the post collision velocity $\mathbf{v}_c'$ of the cell we can calculate the post collision velocity $\mathbf{v}_i'$ of particle i:

$$\Delta \mathbf{v}_c := \mathbf{v}_c' - \mathbf{v}_c \tag{2.23}$$

$$\tilde{\mathbf{v}}_i' := \mathbf{v}_i + \Delta \mathbf{v}_c. \tag{2.24}$$

We now rescale the relative velocities of the particles with a factor k. The tilde denotes

values before the rescaling takes place, but after the cell collision.

$$\tilde{\mathbf{v}}_i' = \mathbf{v}_c' + \tilde{\mathbf{v}}_i^{rel'} \tag{2.25}$$

$$\mathbf{v}_i' = \mathbf{v}_c' + k\tilde{\mathbf{v}}_i^{rel'}. \tag{2.26}$$

Because of

$$\mathbf{p}_c = \sum_i m_i \mathbf{v}_i \tag{2.27}$$

$$= \sum_i m_i (v_c + v_i^{rel}) \tag{2.28}$$

$$= \underbrace{M_c v_c}_{=\mathbf{p}_c} + \underbrace{\sum_i m_i v_i^{rel}}_{=0} \tag{2.29}$$

rescaling the relative velocity does not change the momentum.

The kinetic energy of the paritcles insides the cell is

$$E_c' = \frac{1}{2}\sum_i m_i \mathbf{v}_i^{'2} \tag{2.30}$$

$$= \frac{1}{2}\sum_i m_i (v_c' + v_i^{rel'})^2 \tag{2.31}$$

$$= \underbrace{\frac{1}{2}\sum_i m_i \mathbf{v}_c^{2'}}_{\tilde{E}_{trans}} + \underbrace{\frac{1}{2}\sum_i m_i \mathbf{v}_c' \mathbf{v}_i^{rel'}}_{=0} + k^2 \underbrace{\frac{1}{2}\sum_i m_i \tilde{\mathbf{v}}_i^{rel'^2}}_{\tilde{E}_{therm}}, \tag{2.32}$$

and with $E_c' = \tilde{E}_{therm} + \tilde{E}_{trans} + \Delta E_c$

$$k = \sqrt{\frac{\tilde{E}_{therm} + \Delta E_c}{\tilde{E}_{therm}}} \geq 1, \tag{2.33}$$

where ΔE_c is the energy the cell lost in collisions with other cell due to a coefficient of restitution < 1. The strategy of this cell collision is depicted in Fig. 2.4. Because the energy is redistributed as thermal energy among the particles and thus increases the granular temperature, the interaction between cells is conservative. Only successive collisions between particles will damp the energy. An obvious question is, whether this will result in the correct cooling behavior for dense systems (see Sec. 3.1).

Afterwards the movement and acceleration of the single particles are calculated. To calculate the movement of the particles one can either use an analytical solution of the equation of motion or apply a standard numerical integration scheme to solve it. Like DSMC this method is less restricted than event driven (ED) simulations, where an analytical solution is required for a fast calculation of the evolution of the system. If the particle tries to enter a full cell the move is rejected: the particle keeps its old position and no acceleration due to external forces is considered.

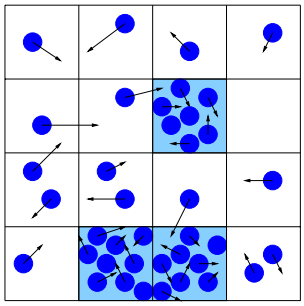


Figure 2.2.: Underlying grid of the HSMC algorithm. Moves into "full" cells (marked grey) are rejected.

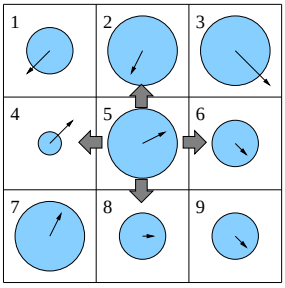


Figure 2.3.: In the HSMC algorithm collisions are performed between neighboring cells. The cells with the highest relative velocities are picked for a collision. The cells 2,4,6 and 8 are possible collision partners for cell 5.

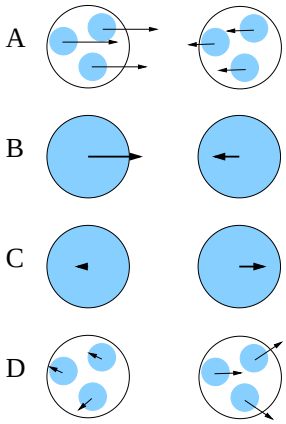


Figure 2.4.: Collision between cells of the HSMC algorithm. From A to B the total momentum and mass of the particles are summed up. Between B and C the collision between the cell pseudo particles take place. The energy dissipated in this collision is distributed among the particles and visible as random relative velocity in D.

Interaction step: Next, we take the particle-particle interactions, *i.e.* the collisions into account. In contrast to ED simulations, the exact times and places of these collisions are not calculated, but a stochastic algorithm is applied as described in the following.

The particles are sorted into spatial cells of linear size L and volume $V_c = L^d$, where d is the dimensionality of the system. Collisions occur only between the particles in the same cell, which ensures that only particles which are close to each other may collide. In every cell with more than one particle, we choose randomly

$$M_c = \chi \frac{N_c(N_c-1)\sigma v_{max}\tau}{2V_c} \tag{2.34}$$

pairs of particles. Here, N_c is the number of particles in the cell, σ the scattering cross section (for spherical particles, $\sigma_{2D} = 4R$, $\sigma_{3D} = 4\pi R^2$) and v_{max} is an upper limit for the relative velocity between the particles. To get v_{max} we sample the velocity distribution from time to time and set v_{max} to twice the maximum particle velocity found. χ is the Enskog factor by which the number of collisions are increased due to the finite extent of the particles. In order to determine the correct number of collisions, we apply an acceptance-rejection method: For a pair of particles i and j the collision is performed if $\frac{|\mathbf{v}_i - \mathbf{v}_j|}{v_{max}} < Z$, where Z is independent uniformly distributed in the interval $[0;1]$. This method leads to a collision probability proportional to the relative velocity of the particles.

Since the collision takes place regardless of the particle positions in the cell, we have to choose an impact parameter b in order to calculate the post collision velocities. Molecular chaos is assumed here; b is drawn from a uniform distribution in the interval $[-2R, 2R]$ in 2D or in a circle with radius $2R$ in 3D. It is also possible to use a different distribution as long as it is known (see Sec. 3.6). The post collision velocities $\mathbf{v}'_i$ and $\mathbf{v}'_j$ are now calculated as if the two particles collided with that impact parameter, i.e. like in event driven simulations.

$$\mathbf{v}'_i = \mathbf{v}_{cm} + \frac{m_j}{m_i + m_j}|\mathbf{v}_{rel}|\mathbf{e} \tag{2.35}$$

$$\mathbf{v}'_j = \mathbf{v}_{cm} - \frac{m_i}{m_i + m_j}|\mathbf{v}_{rel}|\mathbf{e} \tag{2.36}$$

$\mathbf{e}$ is the unit vector pointing in the direction of the relative velocities after the collision. For hard spheres $\mathbf{e}$ is uniformly distributed over the unit sphere [93]. Finally, the dissipation can be introduced by changing the normal component of the post collision velocity to $\mathbf{u}^{(n)} = -e\mathbf{v}^{(n)}$, whereas the tangential component remains unchanged.

Like in the CBA or CUBA algorithm (see Sec. 2.4.2) we apply an additional offset in every collision: each of the two colliding particles is moved by $2R$ into the direction of its momentum change (see Fig. 2.10). This offset is suppressed if the collisions takes place in a full cell. The reason is, that in dense packings the particles cannot move in reality and the artificial diffusion that is introduced by this offset is unrealistical high. To show one consequence if the offset is also performed in full cells a simulation of a static heap was made, where the offset was not suppressed inside the heap. The result

is visible in Fig. 2.5. The heap does not show a definite angle of repose, but a strongly curved surface.

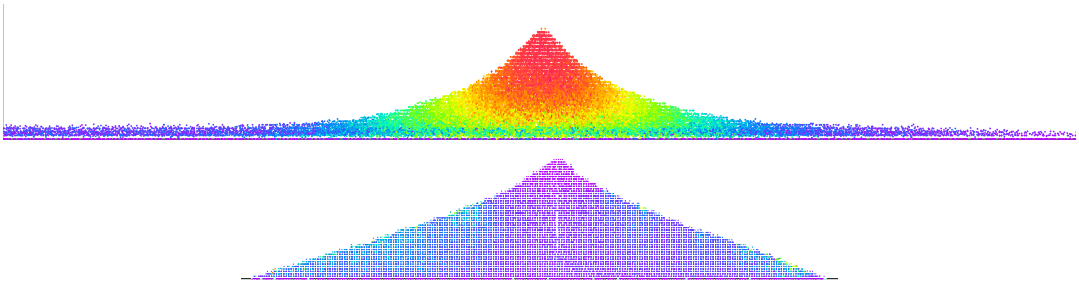


Figure 2.5.: Shape of a heap with (top) and without (bottom) offset at collisions inside the heap. The heap at the top does not show a well defined angle of repose in contrast to the heap at the bottom, where the offset is suppressed in the inner part.

Normally the time step is set to a value $\tau = r_s \frac{L}{v_{max}}$, with $r_s < 1$. Since L is choosen to be smaller than the mean free path of the particles, this ensures that the temporal and spatial resolution of the method is better than the mean free path or mean free time between collisions. This time step determines the mean number of collisions in a cell that occurs in one time step. On the other hand it is also possible to fix the desired number of collisions in one time step and choose the time step accordingly. This has the advantage that the time step is independent of the choosen reference frame, because a different abolute value of the velocity does not change the number of collisions between particles.

Example To demonstrate that the HSMC algorithm provides the excluded volume we look at a system of dissipative particles under gravitation. Fig. 2.7 shows the time evolution of the center of mass of this system. The results of four different simulation methods are shown: Event Driven (ED), Molecular Dynamics (MD), Direct Simulation Monte Carlo (DSMC) and Hybrid Simulation Monte Carlo (HSMC). Up to $t = 0.05$ all four simulation methods agree. Already at $t = 0.1$ the center of mass in the DSMC simulation is too low. Up to $t = 0.3$ the deviations between HSMC and ED are of the same size as the deviations between ED and MD. At $t = 0.3$ the ED simulations stops because an inelastic collapse occurs. For $t > 0.3$ all particles in the DSMC simulations are on the ground, this corresponds to unrealistic high densities. See also Fig. 2.6 for a sketch of the final configuration. HSMC gives results that are comparable to the MD simulation. The smaller value of HSMC is related to the fact, that within a cell there is no excluded volume enforced and therefore the particles tend to overlap and to be located at the bottom of the cells. The length of a cell in this simulation was 0.02 compared to a deviation of ≈ 0.004.

Like the other DSMC algorithm (see chapter 2.4) HSMC treats the Boltzmann equation numerically. In addition to other improvements made in this algorithms HSMC

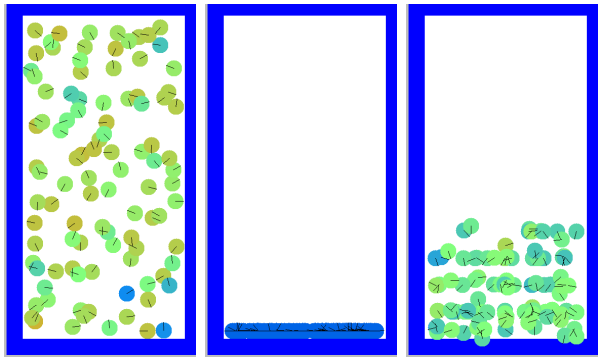


Figure 2.6.: Snapshots of a system with dissipative particles under gravity. On the left the initial conditions is shown. In the final configuration of DSMC (middle) consists of all particles lying at the bottom. The HSMC algorithm at the right shows a final state consistent with excluded volume.

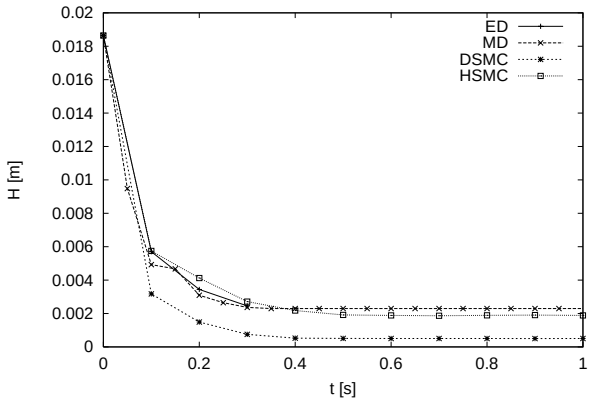


Figure 2.7.: Time evolution of the center of mass for a system of dissipative particles under gravity. The system consists of $N = 100$ particles with diameter 0.001 in a system of size 0.02×0.04.

enforces the constraint that the density of the particles must not exceed a certain limit. This is done by rejecting moves resulting in densities above this threshold. In addition to enforcement of this constraint a pseudo dynamic is applied by introducing the collisions between cells described above. This pseudo dynamic only acts in regions with densities close to the maximum density allowed. This is true for all modifications of the algorithms. For low densities the original DSMC or more precisely the CBA/CUBA algorithm is recovered.

2.4. Direct Simulation Monte Carlo methods

Because HSMC is based on DSMC, the method is presented here together with modified versions of the original algorithm. Most of these modifications aimed to improve the method for higher densities, the same target HSMC is aiming at.

2.4.1. Direct Simulation Monte Carlo (DSMC)

The DSMC method was first used for the simulation of rarefied gas flows[11], recently it was also applied to dry granular media [14, 90, 88].

Several variants of DSMC exist. In the original algorithm of Bird[11] first the collision partners are picked and the time is advanced accordingly. In the modification of Nanbu[93] the time step is fixed and the collision are performed according to this time step. The two methods give equivalent results[34] for systems where DSMC is applicable. Because a fixed time step offers advantages for a parallel implementation Nanbus algorithm is used here.

The system is integrated in time steps τ. One of the basic assumptions of DSMC is that the movement and interaction of the particles can be decoupled (operator splitting, see equation (2.18)). This is a good approximation if the time step is smaller than the mean time between two collisions of one particle. At each time step every particle is thus first moved, according to the equation of motion, without interaction with other particles. External forces, such as gravitation, are taken into account here. To calculate the movement of the particles one can either use an analytical solution of the equation of motion or apply a standard numerical integration scheme to solve it. In this respect this method is less restricted than event driven (ED) simulations, where an analytical solution is required for a fast calculation of the evolution of the system.

Next, we take the particle-particle interactions, *i.e.* the collisions into account. In contrast to ED simulations, the exact times and places of these collisions are not calculated, but a stochastic algorithm is applied as described in the following:

The particles are sorted into spatial cells of linear size L and volume $V_c = L^d$, where d is the dimensionality of the system. Collisions occur only between the particles in the same cell, which ensures that only particles which are close to each other may collide. In every cell with more than one particle, we choose randomly

$$M_c = \frac{N_c(N_c-1)\sigma v_{max}\tau}{2V_c} \tag{2.37}$$

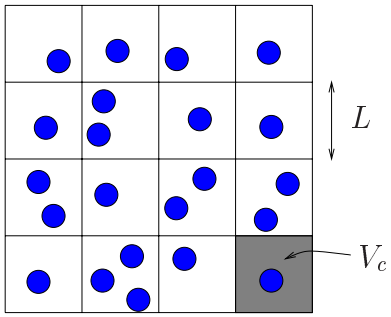


Figure 2.8.: Cells of the DSMC algorithm. The cells have length L and Volume V_c. Particles in the same cell can collide with each other.

pairs of particles. Here, N_c is the number of particles in the cell, σ the total scattering cross section (for spherical particles, $\sigma_{2D} = 4R$, $\sigma_{3D} = 4\pi R^2$) and v_{max} is an upper limit for the relative velocity between the particles. To get v_{max} we sample the velocity distribution from time to time and set v_{max} to twice the maximum particle velocity found. In order to determine the correct number of collisions, we apply an acceptance-rejection method: For a pair of particles i and j the collision is performed if

$$\frac{|\mathbf{v}_i - \mathbf{v}_j|}{v_{max}} < Z, \tag{2.38}$$

where Z is independent uniformly distributed in the interval $[0; 1]$. This method leads to a collision probability proportional to the relative velocity of the particles.

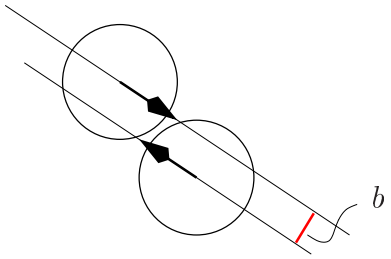


Figure 2.9.: Description of the impact parameter b.

Since the collision takes place regardless of the particle positions in the cell, we have to choose an impact parameter b in order to calculate the post collision velocities. Molecular chaos is assumed here; b is drawn from a uniform distribution in the interval $[-2R, 2R]$ in 2D or in a circle with radius $2R$ in 3D. The post collision velocities $\mathbf{v}'_i$ and $\mathbf{v}'_j$ are now calculated as if the two particles collided with that impact parameter, i.e. like in event driven simulations or HSMC.

$$\mathbf{v}_i' = \mathbf{v}_{cm} + \frac{m_j}{m_i + m_j}|\mathbf{v}_{rel}|\mathbf{e} \tag{2.39}$$

$$\mathbf{v}_j' = \mathbf{v}_{cm} - \frac{m_i}{m_i + m_j}|\mathbf{v}_{rel}|\mathbf{e} \tag{2.40}$$

$\mathbf{e}$ is the unit vector pointing in the direction of the relative velocities after the collision. For hard spheres $\mathbf{e}$ is uniformly distributed over the unit sphere[93]. Finally, the dissipation can be introduced by changing the normal component of the post collision velocity to $\mathbf{u}^{(n)} = -e\mathbf{v}^{(n)}$, whereas the tangential component remains unchanged.

The algorithm can be generalized for soft sphere potentials. It is also possible to introduce rotations[94].

Like every simulation method, DSMC is also based on certain simplifications and approximations. One is that the interaction between the particles can be modeled by binary collisions. Furthermore, neither the location nor the time of a collision is calculated exactly. To keep the error small, three conditions must be fulfilled: (i) the system should be in the collisional regime, (ii) the mean free path should be larger than the cell size, (iii) the mean time between two collisions should be larger than the time step. In particular, at present these limitations restrict the validity of the method to relatively low densities.

2.4.2. Consistent Boltzmann Algorithm (CBA)

In DSMC only ideal gas properties have been considered; the particles have a scattering cross section but no real volume. One consequence of the excluded volume is an increase in the number of collisions. Several approaches exist to take this into account. One possibility is to multiply the right hand side of Eq. (2.37) with the Enskog factor χ. One can either use a Padé kind of approximation for $\chi(\nu)$[3] or an analytical expression [56].

F. J. Alexander et al. [2] also introduced an additional advection process after a collision: each of the two colliding particles is moved by $2R$ into the direction of its momentum change (see Fig. 2.10). The hand waving argument is that a particle with real extension would on average travel $2R$ less distance between two successive collisions than a point like particle. This advection step leads to the correct equation of state for all densities. The reason is explained below. In a particle system the stress tensor $\sigma_{\alpha\beta}$ contains two contributions (see equation (2.41)). First, momentum is transferred, because the particles are moving. Second, a collisional transfer takes place, due to the interaction between the particles. For the original DSMC, the later contribution to the stress tensor will be zero on average because the momentum transfer Δp_α^{ij} and the separation r_α^{ij} of the two colliding particles are uncorrelated. The offset of $2R$ introduces the correlation existent in a hard core gas. By adjusting this offset it is also possible to simulate particles with different potentials, i.e. van der Waals interaction [5].

$$\sigma_{\alpha\beta} = \frac{1}{V_c}\left(-\sum_i m_i v_\alpha^i v_\beta^i + \sum_{ij} r_\alpha^{ij} f_\alpha^{ij}\right) \tag{2.41}$$

$$= \frac{1}{V_c}\left(\underbrace{-\sum_i m_i v_\alpha^i v_\beta^i}_{\text{movement}} + \underbrace{\sum_{ij} r_\alpha^{ij}\frac{\Delta p_\alpha^{ij}}{\Delta t}}_{\text{collisions}}\right) \tag{2.42}$$

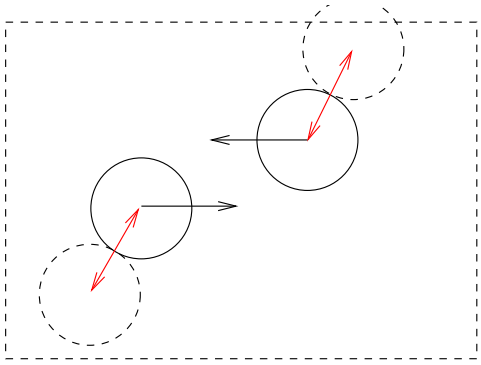


Figure 2.10.: Additional advection process after a collision of two particles in the direction of the respective momentum transfer.

It is clear that this offset will influence the transport coefficients of the simulated system. But as long as the mean free path is much larger than the particle diameter the impact should be negligible. Adding a random offset with expectation value zero will not change the pressure tensor on average. This can be used to adjust the transport properties[30]. The resulting algorithm is called Generalized Consistent Boltzmann Algorithm or Consistent Universal Boltzmann Algorithm (CUBA).

2.4.3. Enskog Simulation Monte Carlo (ESMC)

The Enskog Simulation Monte Carlo addresses another deficiency of DSMC. Whereas DSMC is based on the Boltzmann Equation, ESMC[83] is based on the standard Enskog theory (SET) or revised Enskog theory (RET). In this theories the collision term $\left(\frac{\partial f}{\partial t}\right)$ is described by the expression:

$$D^2 \int d\mathbf{v}_1 \int d\hat{\mathbf{D}}\Theta(\hat{\mathbf{D}}\mathbf{g})(\hat{\mathbf{D}}\mathbf{g})\left[\chi(\mathbf{r},\mathbf{r}-\mathbf{D})f(\mathbf{r},\mathbf{v}',t)f(\mathbf{r}-\mathbf{D},\mathbf{v}_1',t) - \chi(\mathbf{r},\mathbf{r}+\mathbf{D})f(\mathbf{r},\mathbf{v},t)f(\mathbf{r}+\mathbf{D},\mathbf{v}_1,t)\right]. \tag{2.43}$$

Here $\Theta(x)$ is the Heavyside function, $\mathbf{g} = \mathbf{v} - \mathbf{v}_1$ and the primes denote post collision values. SET and RET differ in the way the Boltzmann factor χ is calculated. In SET

$\chi(\mathbf{r}, \mathbf{r}+\mathbf{D}) = \chi(n(\mathbf{r}+1/2\mathbf{D}))$, where $\chi(n)$ is the equilibrium pair correlation function at contact corresponding to a uniform density n. In RET $\chi(\mathbf{r}, \mathbf{r}+\mathbf{D})$ is identified with the local pair correlation function of the nonuniform state (see [83] and references therein). The main difference to the Boltzmann equation is that the two collision partners are drawn from different single particle distribution functions $f(\mathbf{r})$ and $f(\mathbf{r}+\mathbf{D})$. The fact that the two positions differ makes the methods computationally much more expensive. Like in Molecular Dynamics a neighbor search has to be performed. If the spatial resolution is Δr, every particle has an interaction zone of $| \, ||\mathbf{x}-\mathbf{r}|| - D| < \Delta r$. The time step must be small enough to detect a particle when it crosses this zone. Although a collision *may* occur while a particle crosses this zone, there is nothing that guarantees that a particle can't come closer. The method therefore cannot prevent that the density grows above physical realistic values.

2.5. Other particle methods

For the development of an algorithm it is important to have an overview of methods that are already applied to the problems that should be solved by the new method. This is not only necessary to check whether an algorithm already exists that has the required properties, but also to learn from the advantages and deficiencies of the existing solutions. Many different methods are used to simulate granular materials[44]. Also far from being complete the next three sections cover not only a wide range of methods, but also the methods that are most frequently applied to granular matter.

2.5.1. Molecular Dynamics and Contact Dynamics

Molecular Dynamics is often regarded as the most precise method to simulate granular materials, because it models the interaction between the particles with a detailed force law.

Molecular Dynamics is an algorithm to solve Newtons equation of motion

$$\frac{d\mathbf{p}_i}{dt} = \mathbf{F}_i. \tag{2.44}$$

The momentum $\mathbf{p}_i$ of particle i changes due to the force $\mathbf{F}_i$ on that particle. The force is composed of an external force $\mathbf{F}_{ext}$ and forces between the particle and other particles

$$\mathbf{F}_i = \mathbf{F}_{ext} + \sum_{i \neq j} \mathbf{F}_{ij}. \tag{2.45}$$

The calculation of equation (2.45) results in a N^2 loop over all possible pairs if N particles are interacting with each other. For short range forces several optimization techniques like Verlet neighbor lists or linked cell algorithm exist[6, 26], that result in an $O(N)$ algorithm. Short-range Molecular Dynamics has the additional advantage that it is well suited for parallel computers[102].

In the linked cell algorithm the system is divided into cells of length L_c, where L_c is the so called cut-off length. All interactions beyond this range are neglected. It is therefore sufficient to look for interaction partners in the same cell and neighbor cells (see Fig. 2.11).

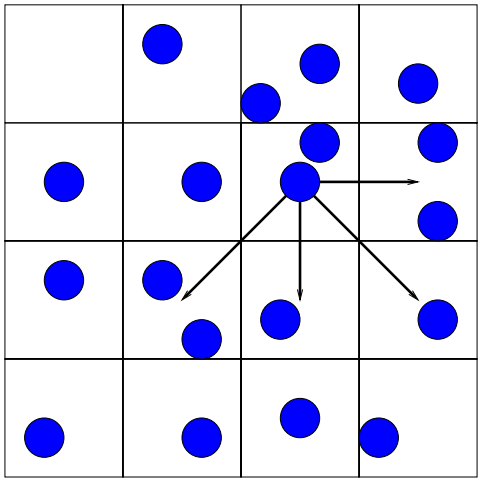


Figure 2.11.: Linked cell algorithm of Molecular Dynamics. The search for interaction partners is limited to cell and its neighbor cells.

The main difficulty of Molecular Dynamics applied to granular matter is the modeling and implementation of static friction. One model is the law of Coulomb, which can be written as

$$F_t = \begin{cases} -\operatorname{sgn}(v_t)\mu_d F_n & \text{if} \quad v_t \neq 0 \\ -\mu_s F_n \ldots \mu_s F_n & \text{if} \quad v_t = 0. \end{cases} \tag{2.46}$$

F_t and F_n denote the tangential and normal components of the contact force, while μ_d and μ_t are the dynamic and static coefficients of friction. The tangential force is a non-smooth function which results in a non-smooth mechanics.

A method widely used in literature is the so called Cundall-Strack spring[19]. They introduced a additional tangential force F_t, with

$$F_t = -\operatorname{sgn}(\xi_t)\min(k_t|\xi_t|, \mu F_n) \tag{2.47}$$

$$\xi_t = \int_{t_0}^{t} v_t dt'. \tag{2.48}$$

Here ξ_t can be regarded as the elongation of an imaginary tangential spring that is attached to the particles at the time of contact formation t_0. As soon as the contact breaks the spring is also broken. While this method gives realistic results it does not implement Coulombs law of friction (2.46) correctly.

The main problem with Coulomb law of friction is that it is not a single valued function (see figure 2.12). Instead it is a constraint, that sets the relative velocity to zero

as long as the force required to maintain the resting contact is not larger than a threshold $\mu_s F_n$. Because in standard molecular dynamics the force is needed to calculate the motion of the particles, static friction according to equation (2.46) cannot be simulated correctly by MD. An improvement of MD that solves this problem is the so called Contact

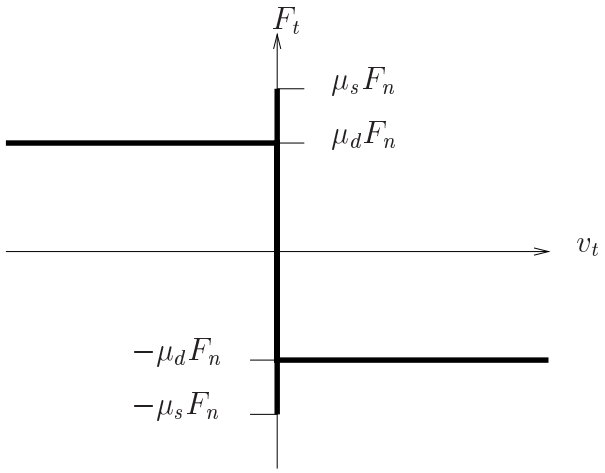


Figure 2.12.: The graph of Coulomb's law of friction representing allowed pairs of (v_t, F_t).

Dynamics (CD). CD calculates the forces between the particles by analyzing the contact network. It determines a self consistent solution for the forces by an iterative method. Thus it calculates the motion of a system with real Coulomb friction. The drawback is, that CD is a rather expensive method, which normally limits the system size to a few thousand particles.

Another problem of the MD method is that the time step has to be smaller than the time of the fastest physical process that needs to be resolved. For explicit integration schemes additional stability criteria may further decrease the time step. In granular systems this time is normally the contact time between two colliding particles. The fact that the contact may be arbitrarily short leads to the so called "brake failure" problem, i.e. no matter how small the time step is chosen, there will always be some grazing collisions with contact times below this time step. Absolutely rigid particles can not be handled by Molecular Dynamics, but have to be approximated by stiff but elastic particles. For dense systems this may lead to artificial multiple particle contacts and accordingly less dissipation [65, 67].

2.5.2. Event Driven or Hard Sphere Molecular Dynamics

This method is also known as hard sphere molecular dynamics. It addresses the problem of Molecular Dynamics with stiff particles. It uses the fact that the outcome of a collision of two perfectly rigid particles is ruled by momentum and energy conservation and known analytically. Because the duration of a collision is zero, the time evolution of the system is calculated from binary collision to binary collision.

For a fast calculation of the collision times the particles should be either freely moving or under constant acceleration. An optimized serial algorithm was proposed by Lubachevsky [62]. The efficient parallel implementation of this algorithm is still an unsolved issue. One problem is that the knowledge of the next collision is a global information that has to be shared among the processors. To eliminate the need for this global knowledge a division of the domain into cells can be introduced. Every time a particle crosses a cell boundary this is a new event. However, this leads to additional events and synchronization points and thus additional effort.

Dissipation due to inelastic collisions is introduced via the restitution coefficient r. This coefficient is the ratio between the normal velocities between the particle before and after the collision. The limits $r = 1$ and $r = 0$ correspond to elastic and completely inelastic collisions respectively. For details concerning the interaction model for granular matted used in the ED method see refs. [63, 68]. Since the ED method readily handles the excluded volume constraint of the particles, it is used to check the results of methods like DSMC or HSMC.

A problem of the zero collision time is the so called "inelastic collapse"[76, 78] : the number of collisions per unit time may diverge and this makes it impossible for the algorithm to integrate the future time evolution of the system. Several work arrounds for this deficiency exists to make the method better suited for dissipative granular systems[71], but they all change the physics of the interaction.

2.5.3. Lattice Gas Automata

The lattice gas automaton (LGA) was introduced as a numerical scheme to solve the Navier-Stokes equation[27]. This method has also been applied to simulate granular media in pipe flow [97], heap formation [7] and others [8, 9, 105, 122].

A LGA proceeds in discrete time steps. The particles are located at the vertices of a two dimensional triangular lattice. In addition to the fixed six directions the absolute value of the velocity is also discretized. A particle can be either at rest or moving in one of the six directions with fixed velocity. This seven states can be either occupied or empty. Therefore, the state of each vertex can be represented by a seven bit state variable.

The time evolution of a LGA consists of a collision step and a propagation step. In the first step collision rules are applied to calculate the outcome of collisions between the particles. In the advection step the particles move to the nearest neighbor sites according to their velocity.

Because of the fixed absolute value of the velocity, dissipation can only be introduced by transforming moving to rest particles. Momentum conservation further limits the choice of suitable collision rules.

The limitation to discrete states and the resulting simple collision rules allow a very efficient implementation of the algorithm concerning memory consumption and speed. However, due to this discrete states the information contained in one state is also limited. Informations about the flow field like velocity can only be obtained by a coarse graining across many states of the LGA.

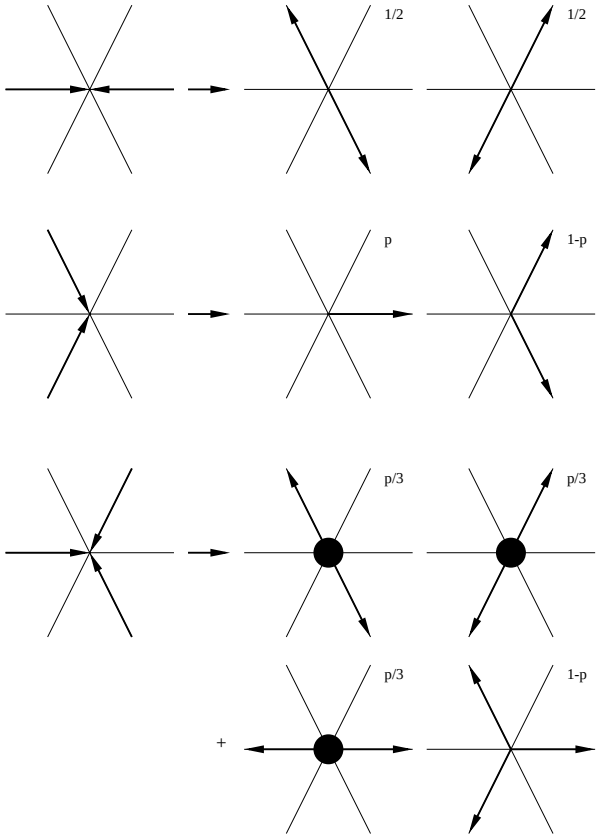


Figure 2.13.: Collision rules of a Lattice Gas Automaton.

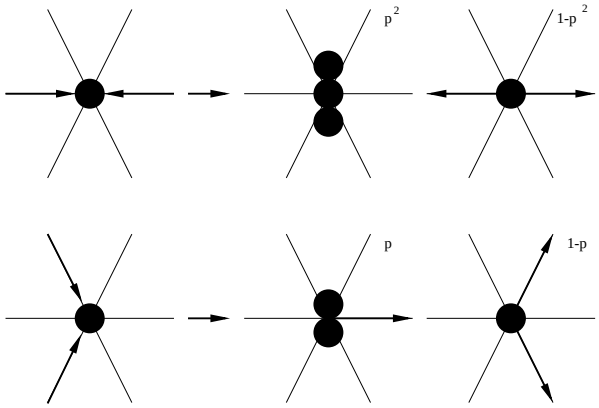


Figure 2.14.: Dissipative collision rules of a Lattice Gas Automaton.

A similar algorithm that extends the information contained in one state is the Boltzmann Lattice Gas method. Like a LGA a Boltzmann Lattice Gas Automaton (BLGA) discretizes space with a triangular lattice. But the seven states of each vertex cannot only be occupied or unoccupied, but a probability is assigned to each state to be occupied. Instead of dealing with particles directly the BLGA treats probabilities that a state is occupied by a particle.

The time evolution of an BLGA also consists of a collision step and a propagation step. In the advection step the probabilities are transferred to the nearest neighbor sites according to their velocities.

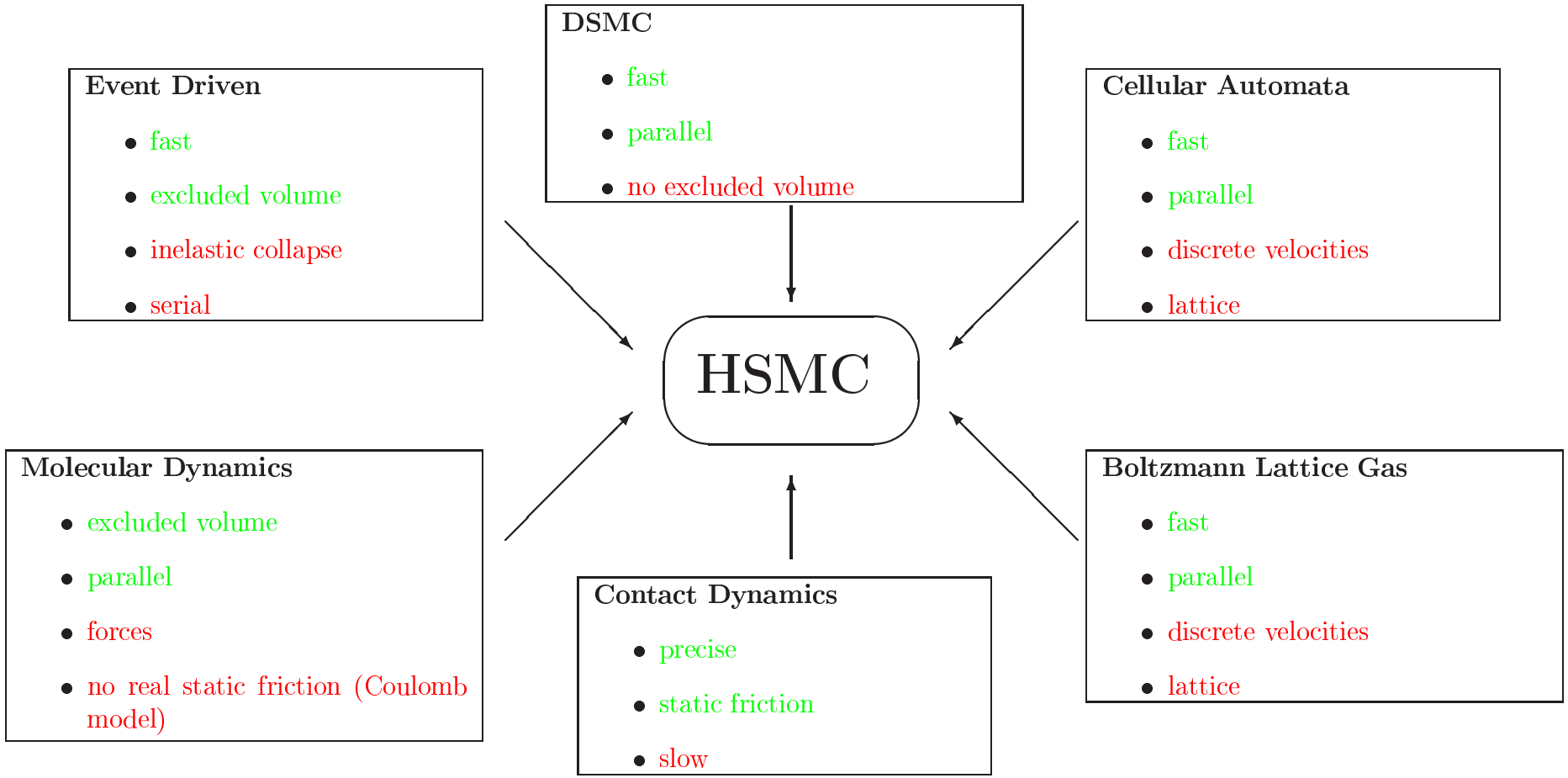


Figure 2.15.: Comparison of different simulation methods and their advantages and disadvantages. The goal was to include most of the advantages into HSMC without incorporating the disadvantages.

3. Physical Test Cases

This chapter contains description and results of the test cases that have been selected to verify the applicability of HSMC to granular systems. They have been chosen to cover a wide range of phenomena. Homogeneous cooling was selected as first test case, because it treats the most basic phenomenon: dissipation. By comparing with theoretical results the correct behavior of HSMC is verified. The next case already shows the demand for large-scale computation. The studied clustering instability occurs only in systems that are large enough. Due to the lack of sufficient theoretical knowledge the results of HSMC are compared with results of event driven and DSMC simulations. In both systems the granular particles are freely evolving, without external forces. In the next two cases driven granular flows are treated. Here energy is brought into the system either by the walls or by gravity acting on the particles. The next system that is considered is even more difficult to treat with a stochastic method like HSMC. So far, the systems have all been kept dynamic, either freely evolving or kept alive by a continuous supply of energy. In the case of heaps however, the system comes to rest in a well defined state. The heap does not only show a well defined angle of repose, but also small deviations at its tail. Because a transition from dynamics to statics is typical for granular systems the correct treatment of this final test case was not only a difficult, but also a very important one.

Section 3.6 looks at one of the assumptions HSMC is based upon: the correlations that exists between different particles in a dissipative system and to what extent they are reflected by HSMC. The section about vibrated beds is an additional case study where the results of the stochastic DSMC are compared with results from hard sphere molecular dynamics. This does not only show once again the correctness of the method, but also answers the question whether correlations between particles in a narrow two dimensional system are important for the correct scaling behavior.

3.1. Homogeneous cooling

In the case of homogeneous cooling we look at a granular gas of dissipative particles that is freely evolving. The total energy is dissipated through inelastic collisions. In a collision between two inelastic hard spheres the particles looses a fraction $\epsilon = 1 - r^2$ of their relative kinetic energy, where r is the normal coefficient of restitution and ϵ is called the inelasticity.

In the homogeneous cooling state[32, 31, 77] the kinetic energy $E(t)$ of the system

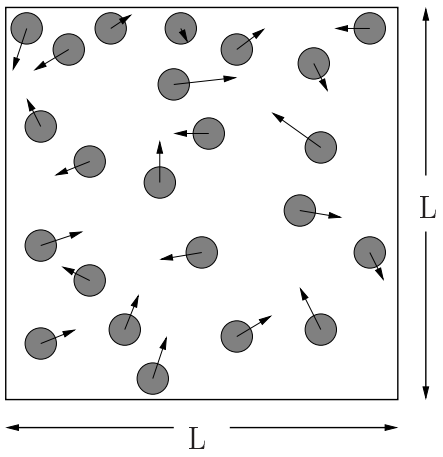


Figure 3.1.: Setup for homogeneous cooling.

decays with time and follows Haff's cooling law[36]

$$\frac{E(t)}{E_0} = \left(\frac{1}{1 + t/t_0}\right)^2 \quad . \tag{3.1}$$

The theoretically expected time scale

$$t_0 = \frac{\sqrt{\pi} d \chi(\nu)}{\sqrt{2}(1 - r^2)\nu \bar{v}} \tag{3.2}$$

is a function of the initial energy $\bar{v} = \sqrt{2E_0/Nm}$, the particle diameter d, the restitution coefficient r, and the volume fraction ν, with $\chi(\nu) = (1 - \nu)^2/(1 - 7\nu/16)$. Equation (3.1) holds as long as the system stays homogeneous. In section 3.2 differences from this behavior are described.

Sometimes the number of collisions per particle τ offer a more natural time scale. In this case Haff's cooling law has the simple form $E(\tau) = E_0 \exp(-2\gamma_0 \tau)$. The relation between "collision time" τ and real time t is given by $d\tau = \omega(E(t))dt$, with the collision frequency $\omega(E) \sim \sqrt{E}$.

3.1.1. Simulation results

Cooling rate

Fig. 3.2 shows the expected agreement between kinetic theory and the HSMC simulation method. HSMC is based on this theory and should therefore reproduce its results. This is not self evident for dense systems where the original method was modified. Fig. 3.3 shows that even for area fraction up to 0.9 Eq. (3.1) is reproduced. One could have anticipated that the dissipation would be underestimated by the HSMC algorithm because the interaction between crowded cells is conservative (see section 2.3). The

agreement shows that the transformation of translational to "thermal" energy will lead to the correct dissipation in subsequent time steps.

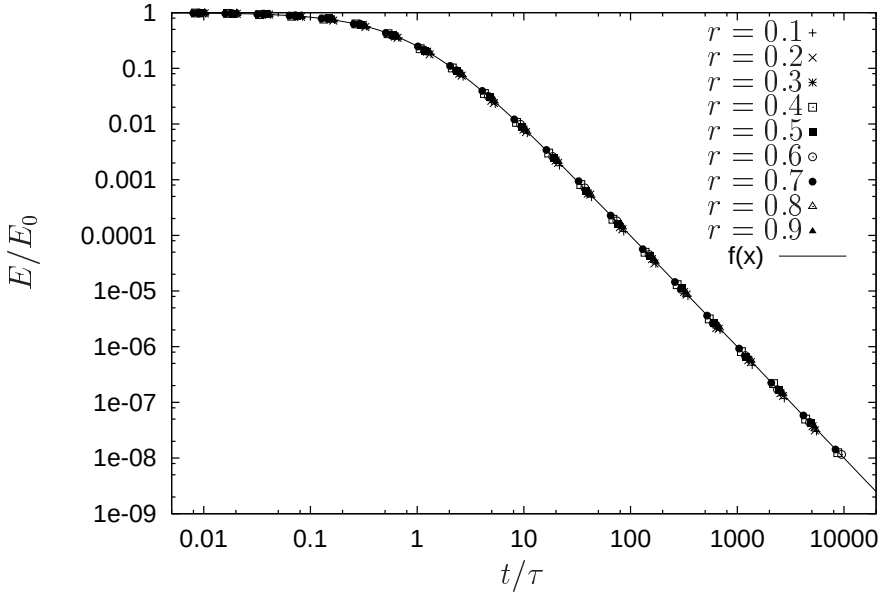


Figure 3.2.: Homogeneous cooling of a freely evolving granular gas for different restitution coefficients r. The area fraction ν was $3 \cdot 10^{-5}$, f(x) refers to Eq. (3.1).

Non-Maxwellian velocity distribution

The initial state of the system is carefully equilibrated and the velocities of the particles follow a Maxwellian distribution function Φ_0. This does not hold for the dissipative system. However, the exact form of the velocity distribution function Φ is not known, even in the case of a dilute system. Approximations of the solution can be obtained by expanding Φ in Hermitian or Sonine polynomials with Φ_0 as a starting point [14].

In contrast to many analytical approaches, the validity of HSMC is not based on a specific velocity distribution function.

Because the deviations from the Maxwellian distribution functions are small, we do not compare Φ_0 and Φ directly, but measure the moments of the distributions. The moments are used to characterize a distribution function and are defined as follows:

$$\alpha_i = \int v^i \Phi(v) dv \tag{3.3}$$

For a Gaussian distribution $\Phi(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\bar{x})^2}{2\sigma^2}}$ we can calculate the moments with

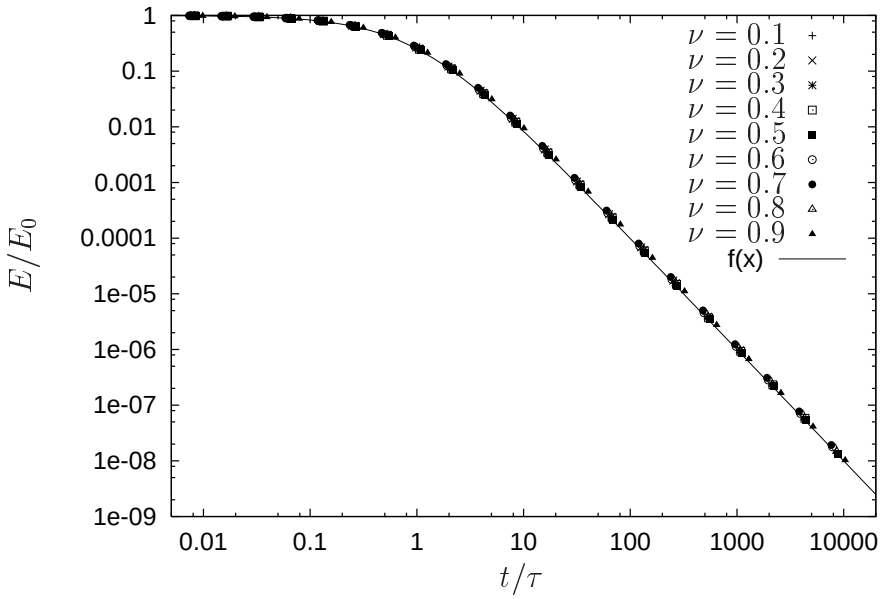


Figure 3.3.: Homogeneous cooling of a freely evolving granular gas for different volume fractions ν and coefficient of restitution $r = 0.99999$. f(x) refers to Eq. (3.1).

the relation

$$\int_0^\infty x^n \mathrm{e}^{-ax^2} dx = \frac{\Gamma\left(\frac{n+1}{2}\right)}{2a^{\frac{n+1}{2}}} \text{ if } a > 0, n > -1. \tag{3.4}$$

In this case $\bar{x}$ equals 0 and a is $\frac{1}{2\sigma^2}$, this results in the following moments:

i	0	1	2	3	4	5	6	7	8
α_i^0	1	0	σ^2	0	$3\sigma^4$	0	$15\sigma^4$	0	$105\sigma^6$

For the simulation results we simulated a system of $N = 1000$ particles and a system size $L = 10\lambda$. To increase the accuracy the results have been averaged over 100 runs with different initial conditions. Fig. 3.4 shows the time evolution of the moments and their deviation from the Maxwellian values. The results demonstrate that although HSMC is based on kinetic theory there is no implicit assumption about the velocity distribution of the particles. In contrast to many simplified theoretical approaches where a Maxwellian distribution is assumed, HSMC can also be used where the velocity distribution is unknown. In other words, the correct distribution is an outcome of the simulation not a prerequisite or input.

3.1.2. Suitability of other simulation methods

The energy of the homogeneous cooling systems simulated here varies over eight orders of magnitude. Because the mean free path l stays constant, the mean time between collisions $\tau = \frac{l}{\bar{v}} \sim \frac{l}{\sqrt{E}}$ changes over four order of magnitudes. This is no problem for

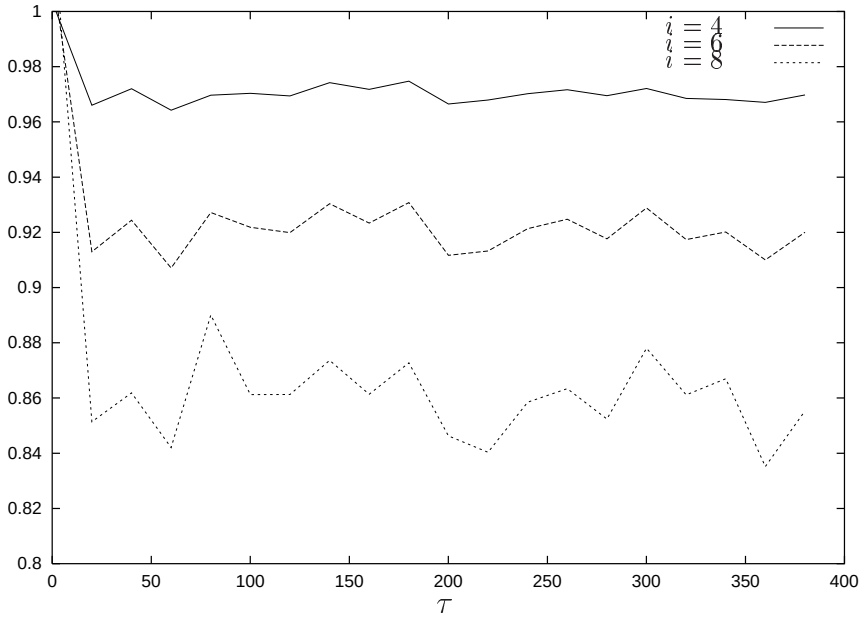


Figure 3.4.: Time evolution of the moments of the velocity distribution function. The moments α_i are given as ratios to the moments α_i^0 of a Maxwellian distribution of velocities.

simulation methods like hard sphere molecular dynamics, because there is no time step, but the simulation proceeds from collision to collision. DSMC and HSMC also contain no intrinsic time. The time step can be adjusted to a constant fraction of the mean time between collisions. The efficiency of the method stays the same, regardless of the systems energy. This is different for the soft sphere molecular dynamics, where the contact time between two particles introduces a time scale. The time step of a MD simulation has to resolve the contact time. Thus the MD cannot follow the dynamic of the system, but gets more and more inefficient as the system becomes slower. The same argumentation holds for LGA models, because a fixed time step is explicitly part of these algorithms. In addition the velocity of a particle can only be one or zero. The slow cooling of a system can only be reproduced by averaging over large particle numbers, which reduces the computational efficiency significantly.

3.2. Clustering instability

The system of interest in this chapter is almost the same as in chapter 3.1. The initial condition consists of a periodic two dimensional system of size $L \times L$ of dissipative particles. At $t = 0$ the system is in equilibrium with a homogeneous density and the particles have a Maxwellian velocity distribution. The only difference is, that the systems of this chapter are larger. Under this circumstance the so called clustering instability [32, 31, 77] may occur.

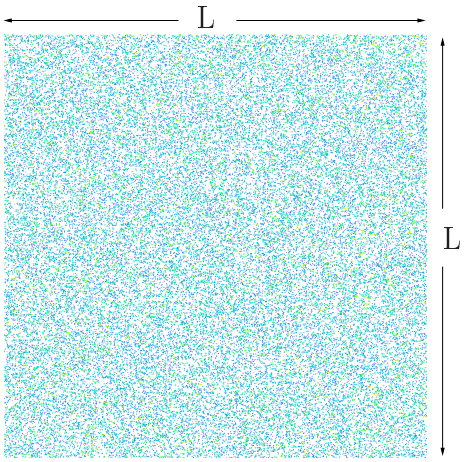


Figure 3.5.: Setup for clustering.

The mechanism for this instability can be described as follows. If due to density fluctuation the density in a region gets higher, the number of collisions is increased. Since in this collisions energy gets lost, this results in a decrease of the granular temperature. More particles flow into the region, because due to the lower temperature the pressure is lower than in the surrounding. This tendency exists also in small systems, but is counterbalanced by the diffusion. If the system gets too large, the diffusion mechanism is too slow, and the system is dominated by the hydrodynamic clustering instability.

In this section we present three simulations, starting with the same initial condition, using the same parameters, but carried out with the three different methods ED, DSMC and HSMC. The simulation involves $N = 99856 = 316^2$ dissipative particles in 2D with restitution coefficient $r = 0.8$ in a periodic quadratic system with volume fraction $\nu = 0.25$. In order to reach an equilibrated initial condition, the system is first allowed to evolve with $r = 1$ for about 10 collisions per particle so that a Maxwellian velocity distribution and a rather homogeneous density distribution is reached. Then, at $t = 0$s, dissipation is set to $r = 0.8$ and the quantities of interest are calculated as functions of time.

3.2.1. Simulation Results

Time Evolution of Kinetic Energy

In the homogeneous cooling state (see section 3.1) we expect that the energy $E(t)$ of the system decays with time and follows the functional form of equation (3.1). This holds as long as the system stays homogeneous and HSMC was able to reproduce this result for any densities. In this section we will focus on a system which will undergo the so called cluster instability.

In Fig. 3.6(a) we present the normalized kinetic energy $K(t)/K(0)$ as a function of the normalized time t/t_0. At the beginning of the simulation we observe a perfect agreement

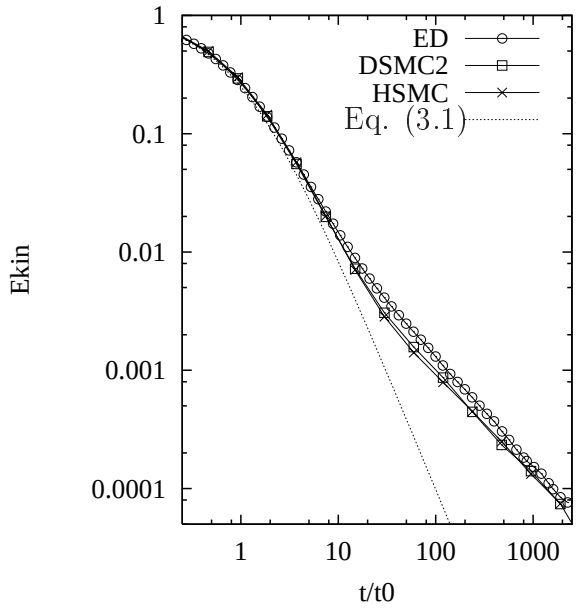


Figure 3.6.: Normalized kinetic energy vs. normalized time from ED, DSMC and a HSMC simulations in 2D with $N = 99856$, $\nu = 0.25$, and $r = 0.8$. The dotted line represents Eq. 3.1.

between the theory for homogeneous cooling and the simulations. At $t/t_0 \approx 2$ both simulation methods show substantial deviations from the homogeneous cooling behavior, and only at $t/t_0 \approx 10$ we evidence a difference between ED and DSMC. After that time the kinetic energy obtained from the DSMC simulation is systematically smaller than $K(t)$ from the ED simulation. We relate this to the fact that the molecular chaos assumption of a constant probability distribution of the impact parameter b is no longer valid [72]. Since dissipation acts only at the normal component of the relative velocity, DSMC dissipates more energy than ED as soon as the number of central collisions is overestimated. To verify this assumption we take a closer look at the impact parameter and its probability distribution in section 3.2.1.

Distribution of Impact Parameter

One basic assumption connected to molecular chaos is a uniform probability distribution of the impact parameter. We define $P(b/d)$ to be the probability distribution of b and normalize it such that $\int_0^1 \mathrm{d}(b/d)P(b/d) = 1$. In ED the probability distribution may deviate from the case expected for molecular chaos, whereas DSMC and HSMC always use the function $P(b/d) = 1$. We find from ED simulations with elastic particles the normalized probability distributions $P(b/d) = 1$ in 2D and $P(b/d) = 2b/d$ in 3D, as expected for the case of molecular chaos.

The ED simulation of Fig. 3.6 leads to $P(b/d) = 1$ for short times only. For larger times we observe an increasing (decreasing) probability of grazing (central) collisions. In Fig. 3.7 we present data of the probability distribution at different times during the simulation. As obvious from the data, more and more grazing collisions occur with increasing simulation time. Evidently, the assumption of a homogeneous probability distribution of the impact parameter is violated.

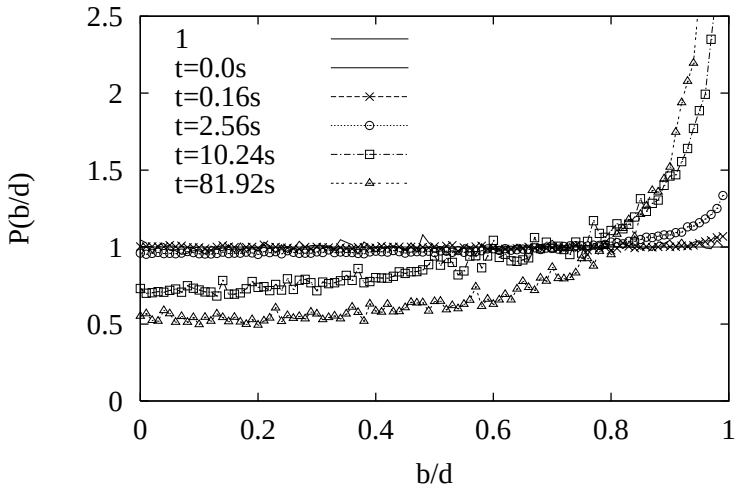


Figure 3.7.: Normalized probability distribution of the impact parameter from an ED simulation in 2D with $N = 99856$, $\nu = 0.25$, and $r = 0.8$ at different times.

One can imagine at least two possible reasons for the deviation of $P(b/d)$ from the constant value. The first is, that $P(b/d)$ might be a function of the density, and that due to density fluctuations the form of $P(b/d)$ changes. Thus we calculate $P(b/d)$ in smaller systems with $N = 240$, $r = 1$, and different volume fractions, ranging from very dilute to extremely dense systems. From Fig. 3.8(a) we learn that $P(b/d)$ is not sensitive to the density, as long as the system is elastic. The stronger fluctuations for low density come only from a comparatively worse statistic. The second reason for $P(b/d)$ to deviate from unity might be dissipation. In Fig. 3.8(b) the restitution coefficient is varied for fixed $\nu = 0.7495$. For weak dissipation, i.e. $r \geq 0.9$, the distribution is homogeneous. For stronger dissipation $r = 0.80$ the probability of grazing contacts increases.

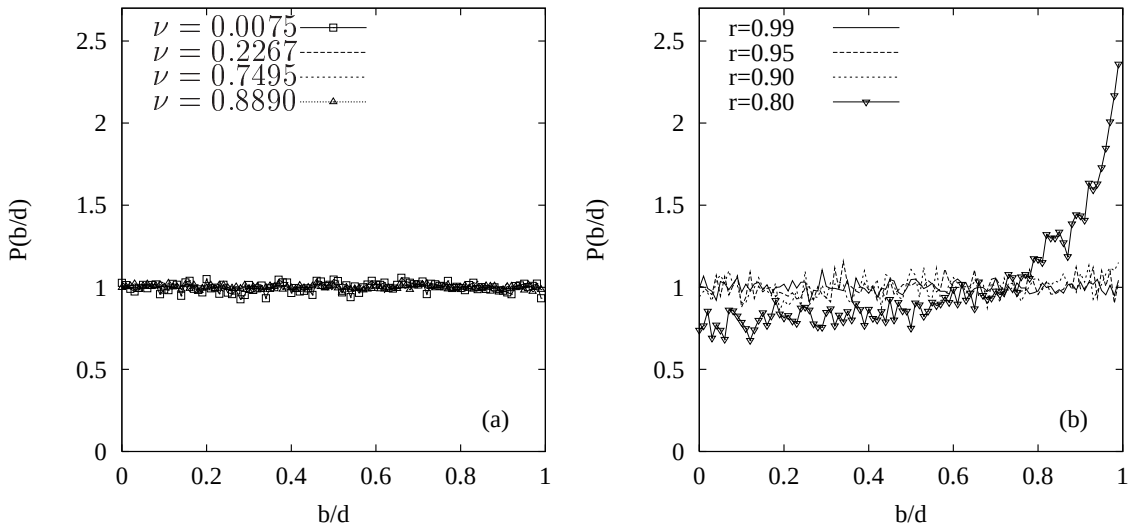


Figure 3.8.: (left) Normalized probability distribution of the impact parameter from ED simulations in 2D with $N = 240$, $r = 1$, and different V_o. (right) Normalized probability distribution from ED simulations in 2D with $N = 240$, $V_o = 0.7495$ and different r.

The assumption $P(b/d) = 1$ is true in elastic systems for arbitrary density. For inelastic systems, $P(b/d)$ is almost constant for sufficiently weak dissipation, but this is not valid for strong dissipation. The breakdown of molecular chaos is *not* due to high density. Furthermore, dissipation alone is *not* the reason for an inhomogeneous probability distribution, since the dissipation must be strong enough to cause the inhomogeneous distribution.

The remaining question is, why we observe this increasing probability of grazing contacts. Looking in more detail at the simulations in Fig. 3.8(right), we observe that the inhomogeneous distribution for $r = 0.8$ is connected to shear motion of the particles, whereas no visible shear motion occurs for $r \geq 0.9$. The shear motion, can be understood as the geometrical reason for the higher probability for grazing contacts.

The structure factor

One difference between ED, HSMC and DSMC simulations is the handling of excluded volume by the two methods. While the ED method models hard spheres with a well defined excluded volume, the DSMC method models point particles and excluded volume is introduced by the approximations described in subsection 2.4.1. As expected we obtain dramatic differences in the particle-particle correlation function $g(r)$: At large times ED simulations lead to a $g(r)$ with a rich structure for short distances, indicating a rather close packing of mono-disperse spheres. In contrast, the DSMC simulations show no short range correlations between particle positions throughout the whole simulation.

We would like to know if this difference has consequences at longer length scales.

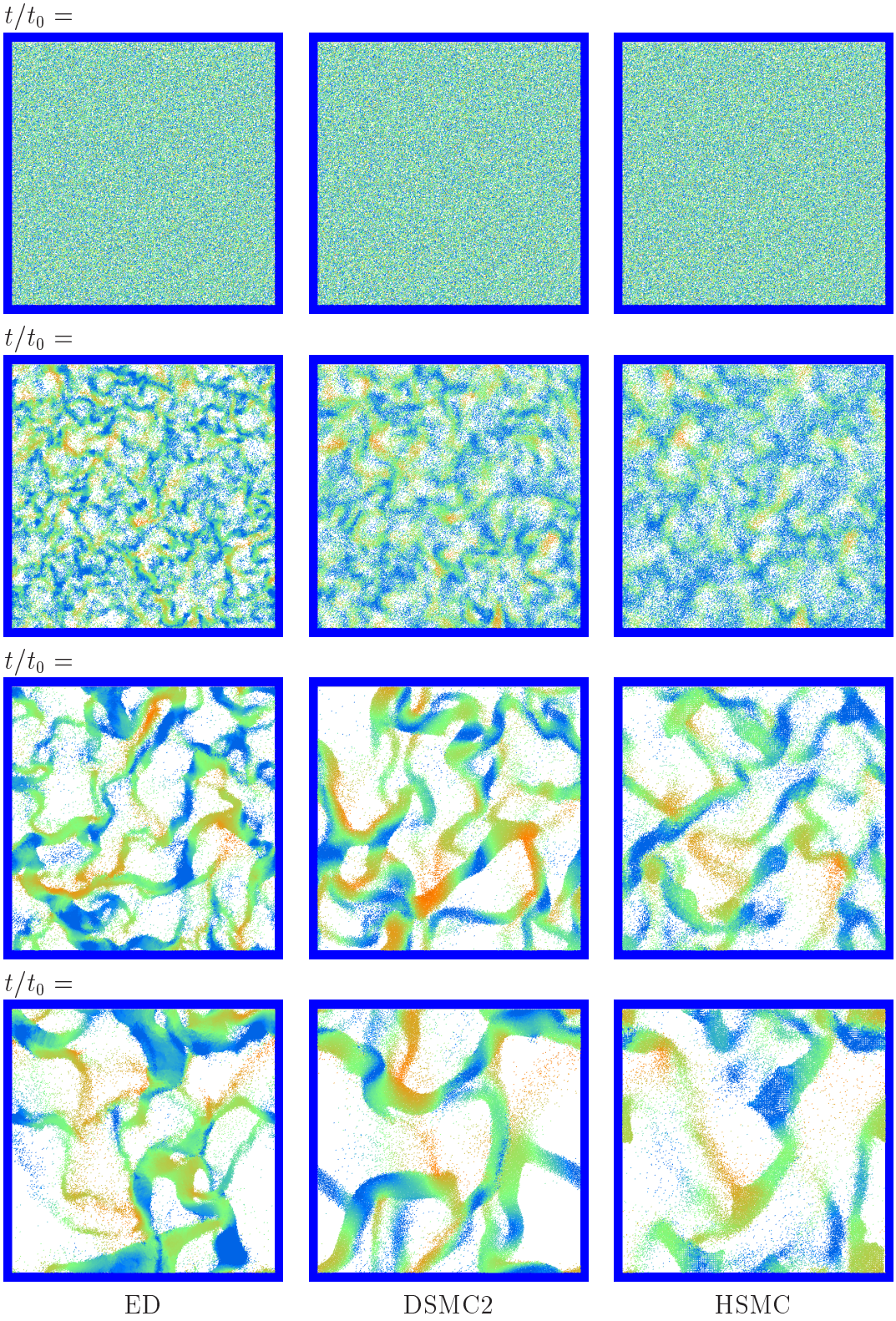


Figure 3.9.: Evolution of the system of Fig. 3.7. Different simulation methods: ED (left), DSMC2 (middle), HSMC (right).

At a first look the results of the different simulation methods look very similar on the scale of the whole system (see Fig. 3.9). The formation and growth of large clusters [31, 32, 77] is quantified by $g(r)$ at large r, or equivalently, the structure factor $S(k)$ at small k. We calculate $S(k)$ by a direct FFT of the two dimensional density. Before we apply the FFT we map the particles onto a $M \times M$ lattice, where M is the closest power of 2 that gives a lattice box size of about one diameter.

We plot the structure factors obtained by ED in Fig. 3.10(a) and those obtained by DSMC in Fig. 3.10(b). Different symbols correspond to different times. We observe an increase of $S(k)$ for short wavenumbers $k < 25$, until the structure factor ceases to change for $t \geq 20$.

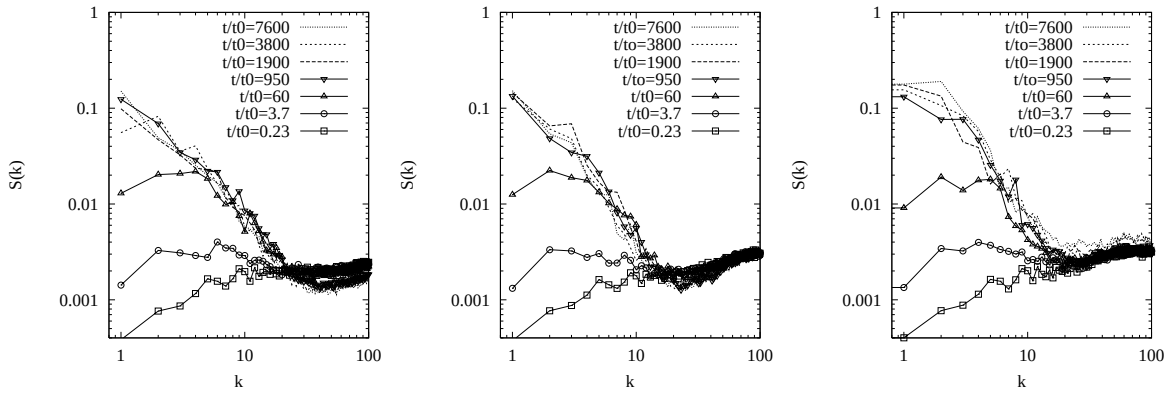


Figure 3.10.: (left) Structure factor obtained from the ED simulations of Fig. 3.6 as function of the wavenumber $k = L/\lambda$, with wavelength λ and system size L. (middle) Structure factor obtained from the corresponding DSMC simulation. (right) Structure factor obtained from the corresponding HSMC simulation.

The structure factor agrees reasonably well for all three simulation methods and for large enough time it does not change further. This proves that both HSMC and DSMC simulations are capable to reproduce the more realistic, but computationally more expensive, ED results that account for the excluded volume by construction. Even without short-range correlations, the information about large wavelengths is well reproduced by the Monte Carlo simulations.

3.2.2. Suitability of other simulation methods

In this chapter the suitability of ED, DSMC and HSMC for the simulation of clustering systems has been demonstrated.

Even with no short range correlation in the Monte Carlo Methods, both methods agree well with respect to long range correlations quantified by the structure factor. This indicates that kinetic theories can succeed in describing the formation and growth of clusters, even though the assumption of molecular chaos is not satisfied. Furthermore,

the faster DSMC and HSMC method can be used to study cluster formation in larger or three dimensional systems.

In the clustering systems the kinetic energy and therefore the time scale changes over several orders of magnitude. Thus, like for the homogeneous cooling systems (see chapter 3.1), neither MD nor LGA methods are efficient for this kind of flows. In addition the systems contain about 1000 times more particles.

Once the system is dominated by clusters most of the kinetic energy is stored in slowly moving clusters. In this aggregates the relative motion of the particles or granular temperature is very low. This state of the system is very difficult to model for LGA methods, where the value of the velocity is limited to a few directions and one absolute value.

3.3. Bagnold shear flow

One of the standard experiments to gain insight into the behavior of materials is the shear experiment. In its classic form it was used for the definition of viscosity μ.

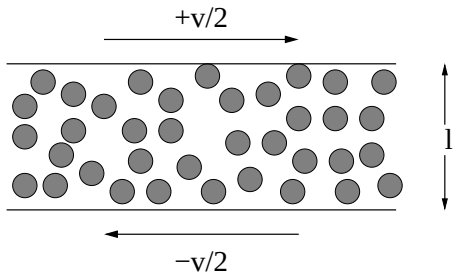


Figure 3.11.: Setup for Bagnold shear flow.

The force F needed to drive two plates of Area A separated a distance l with relative velocity v was found to follow the law

$$F = \mu A \frac{v}{l}. \tag{3.5}$$

For non dissipative particles this system does not have a steady state: due to the viscous heating the temperature of the system is increasing.

In its general form $\frac{v}{l}$ is replaced by the velocity gradient and (3.5) is divided by A, so that the equation has the form

$$\sigma = \mu \nabla \mathbf{v}, \tag{3.6}$$

where σ is the shear stress. The special case of a constant gradient is called uniform shear. This special case is homogeneous in a Lagrangian frame. Recent studies have shown that choosing the correct reference frame is crucial in shear flows[119]. Simulation methods that are not invariant under Galilean transformations will not yield the correct results. In the simulations Lee Edwards boundary conditions are often applied to avoid finite size effects. However, care has to be taken when comparing this kind of simulations with experiments. It is clear that the density and velocity profile will depend on the boundary condition[16]. In the simulations shown in this chapter no Lee Edwards boundary conditions were applied, but rigid moving walls have been implemented. I.e. the system was driven by moving the walls with a constant velocity $v_w = \pm\frac{1}{2}v$ as shown in Fig. 3.11. An algorithm simulating perfectly rigid particles will never show perfect agreement with the experiment, because among other reasons the velocity distribution depends on the interaction potential between the particles[82]. However, properties like Bagnold's law follow theoretical considerations and thus an algorithm like HSMC can be verified without direct comparison with the experiment.

When rigid walls are used instead of Lee Edwards Boundary conditions a collision rule with the walls has to be selected. The rule applied here for perpendicular and parallel velocity components is

$$v'_{\perp} = -r_w v_{\perp} \tag{3.7}$$
$$v'_{\parallel} = 2v_w - v_{\parallel}, \tag{3.8}$$

where $v_{\perp}$ is the velocity component perpendicular to the wall, and $v_{\parallel}$ the component parallel to the wall. This corresponds to a perfectly rigid wall, where in a collision the velocity component parallel to the wall is reversed in the reference frame of the moving wall.

3.3.1. Simulation Results

Velocity profile

The special case of uniform shear discussed here is characterized by a constant velocity gradient and thus a linear velocity profile. This is not necessarily true close to the boundary. If the wall is elastic, there will be less energy dissipation close to the wall and the higher granular temperature will decrease the density and influence the velocity profile. In the extreme case there will be an almost rigid dense region in the center of the system and high velocity gradients close to the wall. Even if the restitution coefficient of the wall is equal to the one of the particles there will be an influence. The reason is, that in this case the wall will behave like a dense collection of particles. The increased dissipation will result in a lower granular temperature with subsequent higher density. To reduce this influence of the wall the coefficient of restitution of the wall r_w was chosen to fulfill

$$(1 - r_w) = (1 - r)\rho_l. \tag{3.9}$$

Where ρ_l is the optical density of the system, i.e. the area fraction of a line which is covered by particles. Thus the dissipation at the wall resembles the dissipation inside the system.

Fig. 3.12 shows that HSMC reproduces the expected linear profile. A result that is also valid for DSMC, because the system is dilute enough that the differences between the two methods are negligible.

Viscosity

For non dissipative particles in a system without walls this system does not have a steady state: due to the viscous heating the temperature of the system is increasing. For dissipative particles the temperature of the system will scale with the velocity of the driving plate and so does μ. Therefore the driving force F will follow Bagnold's law and scale with v^2:

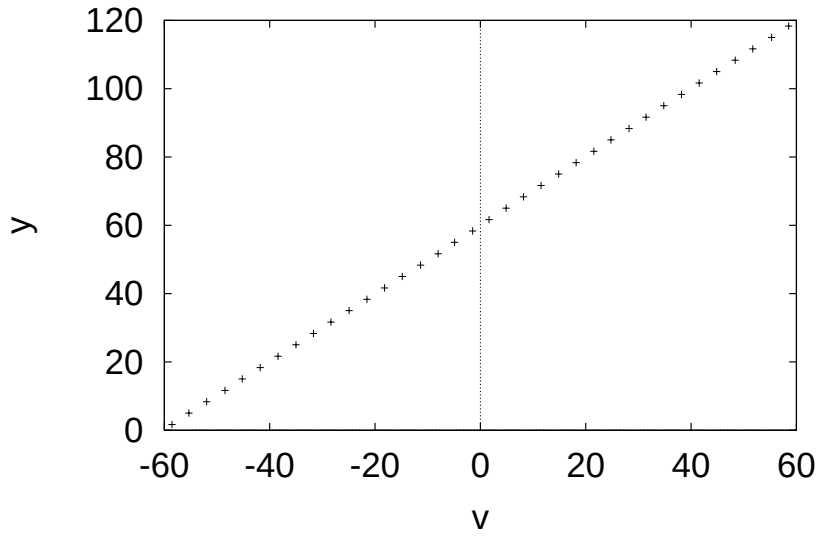


Figure 3.12.: Velocity profile of a sheared granular system.

$$F \sim v^2. \tag{3.10}$$

Fig. 3.13 shows the verification of this scaling behavior by HSMC across three orders of magnitude in v. Eq. 3.10 only holds if no shear induced clustering occurs[118], therefore the area fraction was chosen to be low enough that the system stays homogeneous.

3.3.2. Suitability of other simulation methods

The event driven hard sphere molecular dynamics is equally well suited for the simulation of this flow. Again, the soft sphere molecular dynamics is computationally more expensive, but in this case there is no evolution of the involved time scale throughout the simulation. This makes MD very appropriate for the Bagnold shear flow, this is especially true for dense flows, where the detailed interaction between the particles and the interaction between particles and wall get more and more important. This suitability was demonstrated for Bagnold shear flow and similar flows like Couette Flow [60].

The LGA methods do have the problem, that the velocity in a shear flow ranges continuously over several orders of magnitude across the system. This makes this method again inappropriate.

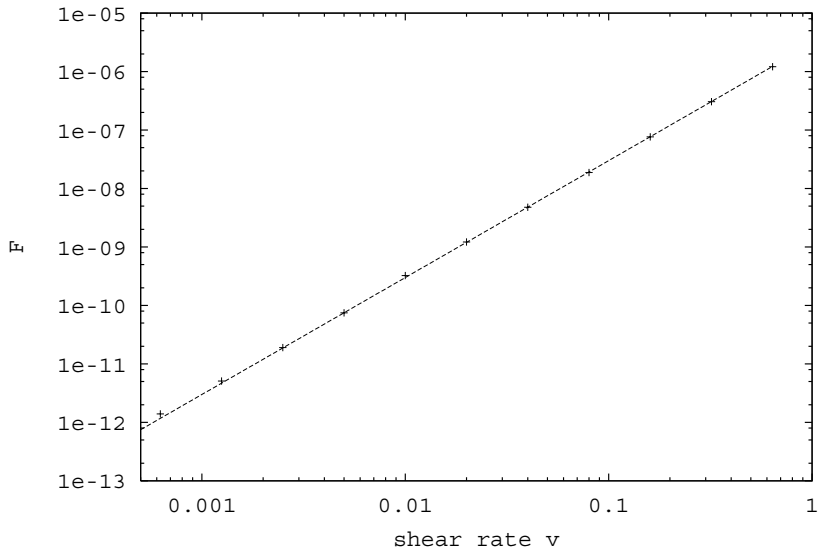


Figure 3.13.: Scaling behavior of driving force F in a Bagnold shear flow. The driving velocity v is varied over three order of magnitudes. Results of HSMC simulations (crosses) are compared with $F \sim v^2$ (line).

3.4. Pipe flow

The vertical flow of granular media through a pipe under the influence of gravity is one of the standard flows that has attracted a lot of attention [1, 84, 99]. One reason for the importance of this flow is its relevance for industrial processing, another is the fact that it is a quasi one-dimensional problem. This makes it accessible for theoretical models and also results in similarities to models of traffic flow [40, 42, 79, 128].

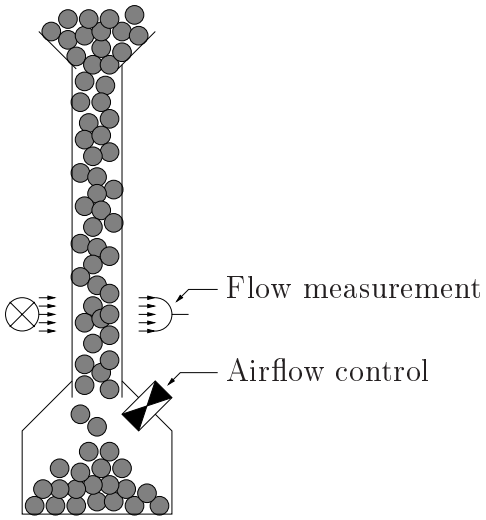


Figure 3.14.: Experimental setup for pipe flow.

A typical experimental setup is shown in figure 3.14. The granular material is poured into the hopper at the top and flows down the pipe. The flow is measured at an appropriate distance from the hopper to allow the flow to equilibrate.

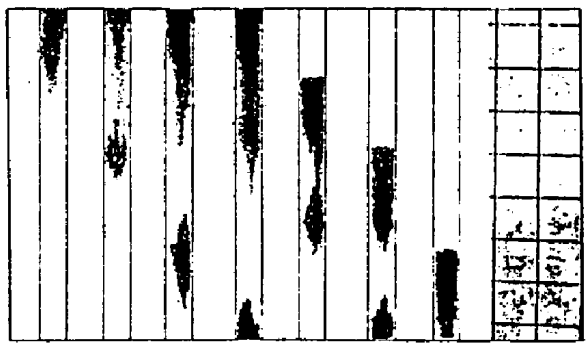


Figure 3.15.: Sequential snapshots of the density waves in pipe flow from [54]. The time is increasing from left to right.

3.4.1. Simulation Results

Spontaneous Density Waves

The first question is, whether HSMC will reproduce the spontaneous creation of density waves. In order to reduce finite size effects periodic boundary conditions have been applied in the direction of gravity. The main difference to a semi-infinite system with a hopper at the top is, that the average volume fractions of particles can be set arbitrarily. In the experiments the area fraction is indirectly determined by the flux of particles entering the pipe.

The roughness of the wall is modeled with a statistic approach. Whenever a particle hits the wall, its velocity component parallel to the wall is reversed with a probability p. A value of $p = 0$ corresponds to a smooth wall. This approach was successfully taken in the case of LGA simulations [97].

For comparison with the experimental results figure 3.16 shows snapshots from a system that starts with a homogeneous density. It is clearly visible that density waves are almost immediately created.

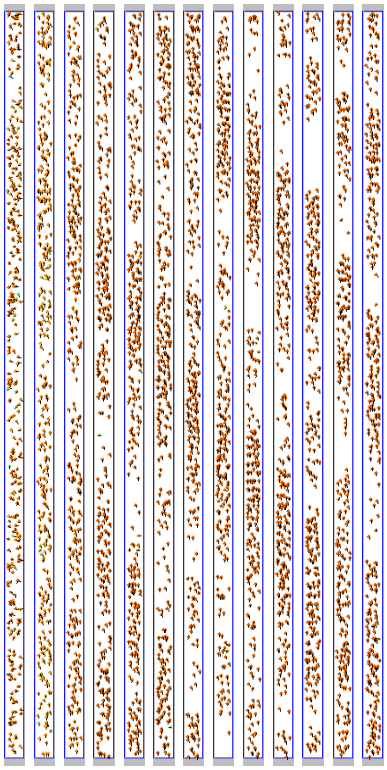


Figure 3.16.: Snapshots of the density waves in pipe flow from a HSMC simulation.

To follow this density fluctuations over a longer time period, figure 3.17 shows the time evolution of the density in two different systems. To measure the vertical density profile, the system is cut into horizontal slices with the height of one HSMC cell. The

average density of every slice is transformed to a gray scale (white equals to area fraction zero, black to one). The density is plotted from top to bottom (direction of gravity) and the time is increasing from left to right.

For the system with small dissipation (restitution coefficient $r = 0.99$) no significant density waves are observable, whereas for the system with strong dissipation ($r = 0.0$) spontaneous density waves occur from the initially homogeneous system. It is also visible that the system reaches an equilibrium state after an initial acceleration. In this case the density waves travel downwards in the direction of gravity.

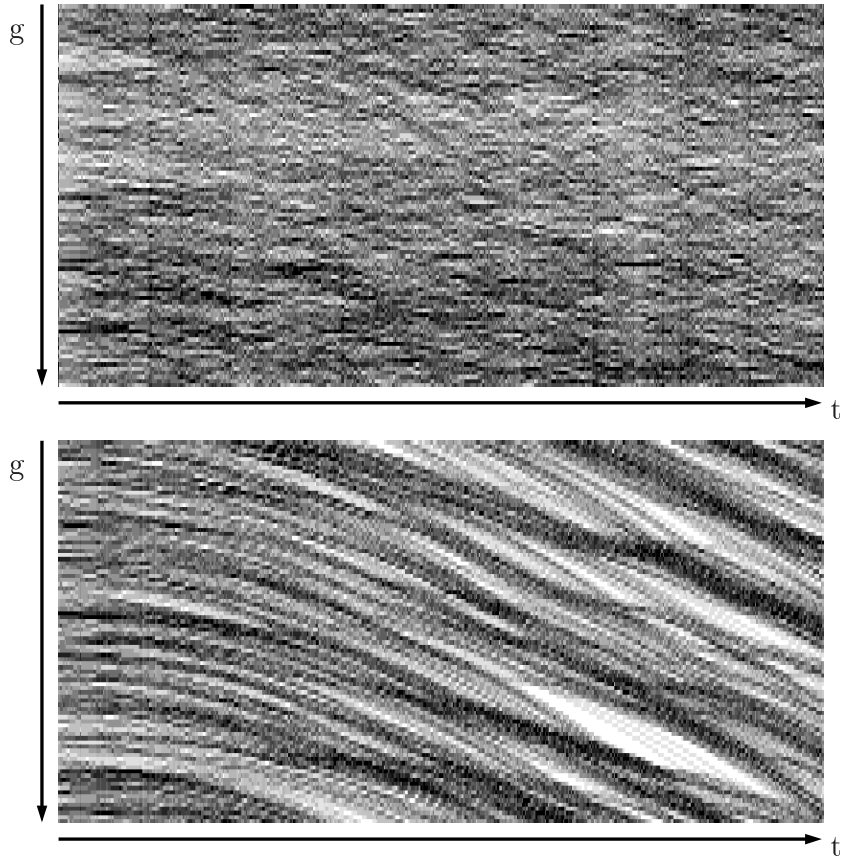


Figure 3.17.: Density fluctuations in pipe flow. The system at the top has a restitution coefficient $r = 0.99$, for the system at the bottom r is 0.0. For the system with small dissipation the density fluctuations do not result in density waves.

Power Spectrum of Density Fluctuations

One of the central points of various studies has been the characterization of the density fluctuations. One possibility is to characterize them by their power spectrum $P(f)$, where f is the frequency of the fluctuations. If from the function $\rho(t)$ an N-point

sample is taken at equal intervals, the power spectrum $P(f)$ is defined as

$$P(f_k) = \frac{1}{N^2}\left[|C_k|^2 + |C_{N-k}|^2\right] \tag{3.11}$$

with

$$C_k = \sum_{j=0}^{N-1} c_j e^{2\pi ijk/N} \quad k = 0, ..., N-1. \tag{3.12}$$

The power spectrum is only defined at discrete frequencies $f_k \equiv 2f_c\frac{k}{N}$, where f_c is the Nyquist frequency. A Fast Fourier Transform can be used to calculate C_k.

LGA simulations of pipe flow found $\alpha \approx -4/3$ [98]. This result was also reproduced by experiments [53] and theoretical considerations [84]. Horikawa et al.[53] measured $\alpha \approx -1.5$ and, in agreement with Moriyama [84] concludes, that the back flow of air is necessary for the observed density waves. The strong influence of the fluid can also be seen when the air is replaced by silicon oil. In this case the power law has an exponent of $\alpha = -0.81 \pm 0.01$[92]. On the other hand, Beshadskii [10] argues that the exponent $-4/3$ is universal for dissipative systems with scalar fluctuations convected by stochastic velocity fields. In addition, LGA does not contain a surrounding fluid and still yields $\alpha \approx -4/3$. However, in order to get a power law for the power spectrum of the density fluctuations, a viscous drag force between the particles and the surrounding fluid was added by Moriyama. This viscous drag results in the introduction of a velocity, where the gravity acting on the particle is balanced by the viscous drag. HSMC does not only contain no surrounding fluid, but also has no intrinsic velocity that may play a similar role. This is in contract to LGA, which has an intrinsic velocity build in, that cannot be exceeded by the particles. If HSMC reproduces the power law it would be a strong indication that no surrounding fluid is neccessary.

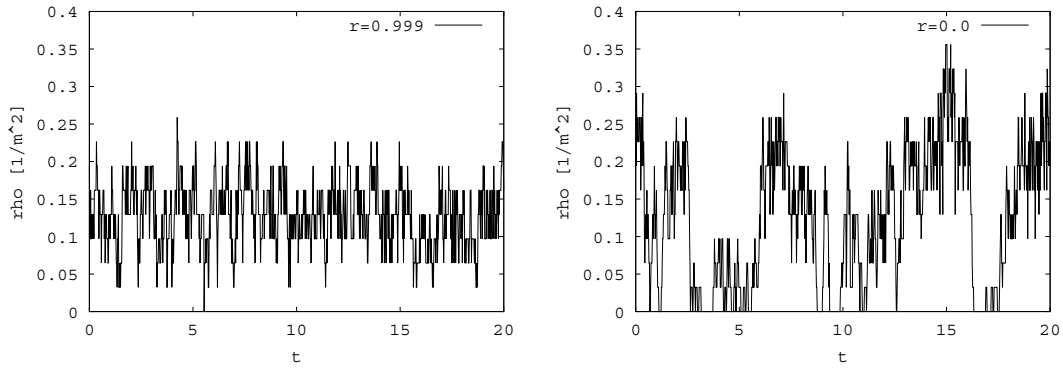


Figure 3.18.: Density ρ vs. time in a pipe flow for systems with different coefficient of restitution r. At the left the restitution coefficient r is equal to 0.999, at the right r is 0.0.

To obtain $P(f)$, the density $\rho(t)$ is measured across a horizontal slice in the system (see Fig. 3.14) during the experiment or simulation. Fig. 3.18 shows $\rho(t)$ for two different

systems. For small dissipation ($r = 0.999$) the density fluctuations are similar to white noise, whereas for strong dissipation ($r = 0.0$) they have a more intermittent behavior.

Fig. 3.19 shows the resulting power spectrum for different coefficients of restitution. The straight lines indicate that $P(f)$ follows a power law $P(f) \sim f^{\alpha}$ over one to two orders of magnitude. There are two cut-offs of this law. For low frequencies the behavior is dominated by the fact, that the system is periodic. There can be no strong fluctuations on time scales above the average time it takes a particle to cross the system length. High frequency fluctuations are dominated by short range particle particle interaction, because they are in the time scale of single particles entering and leaving the slice used for density measurement.

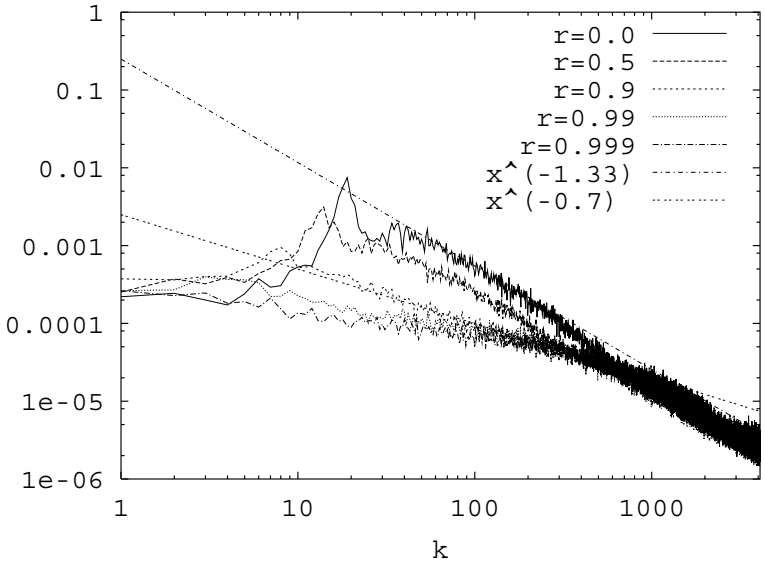


Figure 3.19.: Power spectrum of density fluctuations in a pipe for different restitution coefficients.

Fig. 3.20 shows how the coefficient α changes for different restitution coefficients r. Below a restitution coefficient of 0.6 the slope remains constant at about -1.3. Above $r = 0.6$ α approaches 0 with $r \to 1$, this corresponds to white noise fluctuations as it is expected for elastic particles. This behavior was also observed by LGA simulations [98].

Fig. 3.21 shows the dependency of α from the wall roughness p and area fraction ν. The LGA simulations found a region of constant slope and another region where $\alpha \longrightarrow 0$ for $p \longrightarrow 0$. Here, $\alpha(p)$ shows a more complicated behavior. The fact, that $\lim_{p \to 0} \alpha(p)$ is equal to zero is visible in the diagram. A minimum value of α is reached for $p \approx 0.1$, before α reaches ≈ 0.4 for $p \to 1$. A value of $p = 1$ means that whenever a particle hits the wall, its vertical velocity component points upwards after the collision. There is no variation in the interaction between the wall and the particles. This corresponds to a wall that has constant roughness throughout the system. On the other hand, a value of $p \neq 1$ models a wall that is rough on a fraction of p and smooth on all other parts. The

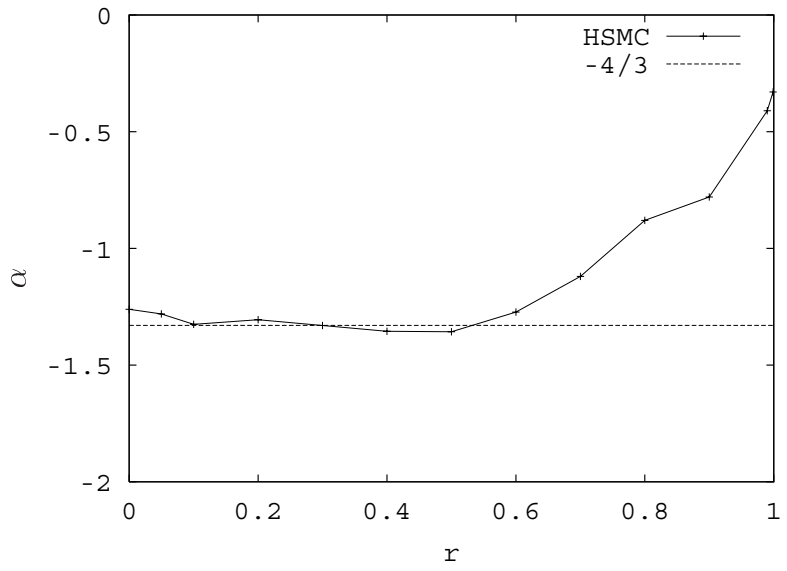


Figure 3.20.: Slope α of power law for power spectrum of density fluctuations in pipe flow for different restitution coefficients r ($\nu = 0.38$, $p = 0.1$). Below a restitution coefficient of 0.6 the slope remains constant at about -1.3. Above $r = 0.6$ α approaches 0 with $r \to 1$, and thus white noise fluctuations as expected for elastic particles.

fact that the wall roughness has to be varied in order to see strong density fluctuations is known from molecular dynamics simulations, where the wall roughness is modeled by attaching particles to the wall [103]. Only if the distance between the "wall particles" is not evenly spaced, there are strong density waves.

Fundamental diagram

One of the major questions in traffic flow is, how the flow rate Φ changes with the density ν of traffic. The resulting diagram $\Phi(\nu)$ is called fundamental diagram. Fig. 3.22 shows the result of the HSMC simulation. The graph shows three different regions. In the first, the flow grows proportional to the particle density ($\nu = 0 \cdots 0.6$). In the second ($\nu = 0.6 \cdots 0.75$), the flow grows faster than the average particle density ν. For high density ($\nu > 0.75$) the flow rate does not grow anymore or is even reduced for increasing density. In this region jamming occurs, which also makes the measurement difficult, because depending on the initial conditions the flow may jam within the simulation period or not.

The three regions with different slope $\beta = \frac{\partial \Phi}{\partial \nu}$ correspond to three different velocities of the propagating density waves. The different velocities can be measured in the diagrams showing the time evolution of the density along a vertical cross section through the system (Fig. 3.23). The diagram for $\nu = 0.88$ also shows the limitation of the HSMC method. The velocity of the density waves is drastically reduced, but does not reach

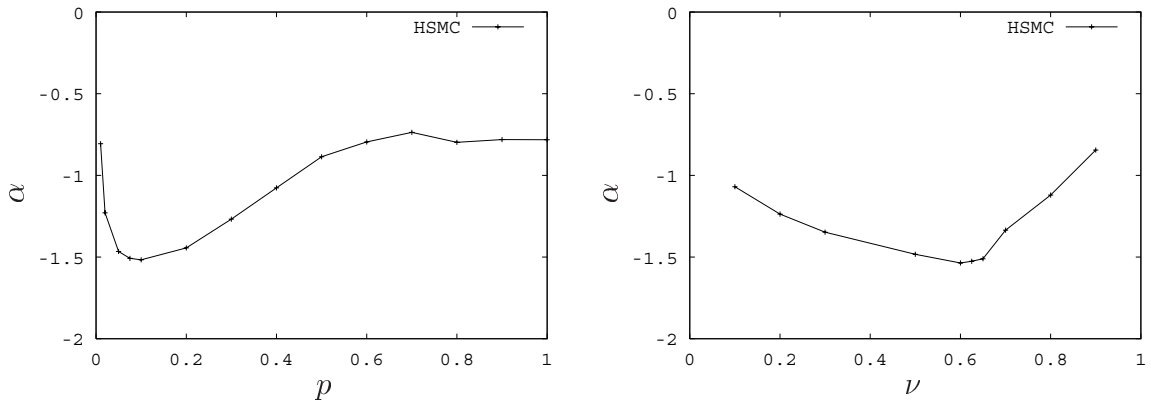


Figure 3.21.: Slope α of power law for power spectrum of density fluctuations in pipe flow for different wall roughness p ($\nu = 0.6$, $r = 0.0$) on the left, and for different area fractions ν ($p = 0.1$, $r = 0.0$) on the right.

negative values as could have been expected by looking at the fundamental diagrams for very high values of ν. For these high densities effects like dynamic arching or spin organizations [69] are important, that are not modeled correctly by HSMC.

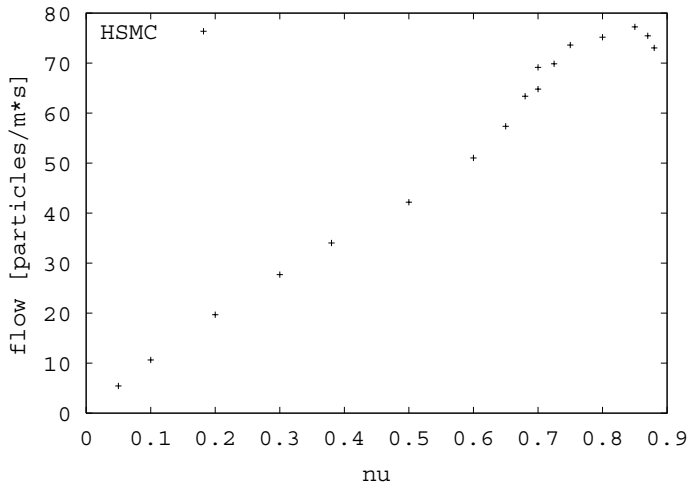


Figure 3.22.: Fundamental diagram for pipe flow (p=0.1, r=0.0).

3.4.2. Suitability of other simulation methods

Various publications demonstrate the suitability of LGA[43, 58, 98], ED [22, 69] and MD[103]. For high densities, methods like ED and MD are preferable to HSMC, because they include effects like arching or spin organizations, that play a dominant role in the case of high densities.

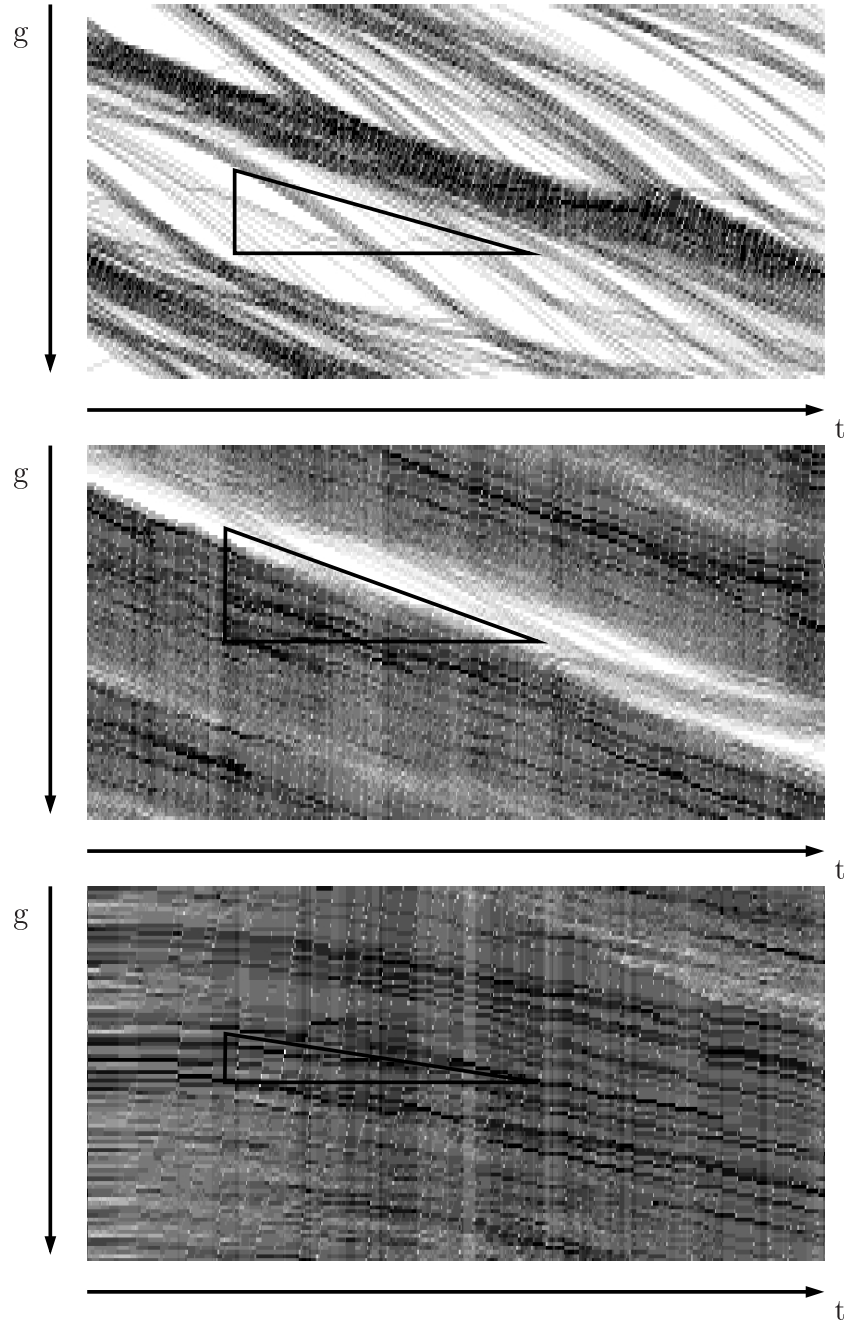


Figure 3.23.: Density fluctuations in pipe flow. The area fractions of the three systems are 0.3, 0.75 and 0.88 (from top to bottom). The triangles mark the velocity of the density wave.

Since DSMC cannot handle the simulation of the system in the dense regions it is not appropriate for the simulation of pipe flows where strong density fluctuations occur.

None of the above methods handles the simulation of the fluid between the particles, especially the back flow of air, that is considered to be crucial in the outcome of the experiments.

3.5. Heaps

A flow where granular matter behaves like a fluid as well as a solid is the formation of heaps. You can pour the grains out of a container (like a fluid) on a table and they form a stable pile with a finite slope (like a solid). Static friction is one of the reasons why granular material behaves like a solid. HSMC however does not include static friction and the question is to what extent the formation of heaps can be reproduced. Using soft sphere molecular dynamics it has been shown that heap formation is possible without static friction[64] due to the steric effects caused by the excluded volume. For the same reason it is also possible to simulate heaps using hard sphere molecular dynamics. Since excluded volume is built in on the coarse grained grid level into HSMC, an important question is to what extent artifacts of the induced anisotropy are visible in the simulations.

In order to build a heap, particles were put into the system at a constant rate (one per time unit). The freshly created particle is put a fixed length above the particle with the highest position in the system. The disadvantage of course is that the simulation time grows linear with the number of particles N. This algorithm however resembles more closely the experimental way of pouring particles out of a funnel than other solutions. To reach an equilibrium state we wait 500 time units after the last particle has been added. Fig. 3.24 shows a typical result. On a first glance the major feature of a heap like the finite slope are reproduced well. In the following we will have a closer look at the shape of the heap.

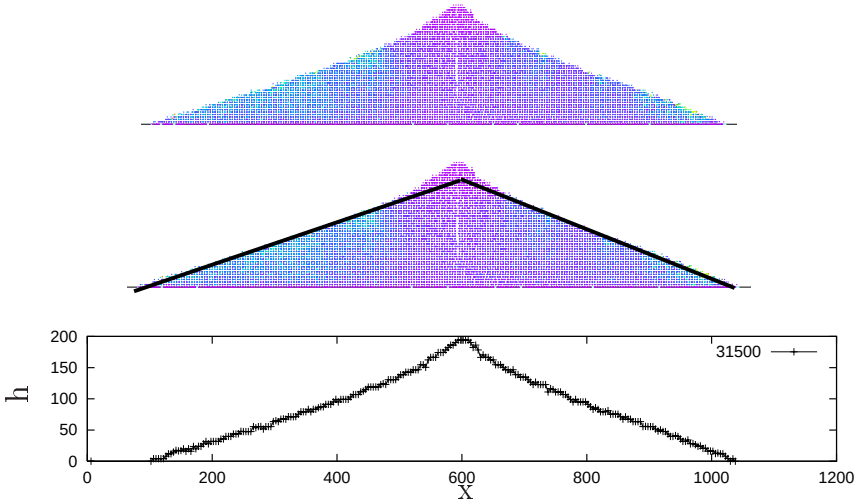


Figure 3.24.: Heap of 30500 particles simulated with HSMC. The picture at the top is a snapshot of the heap. In the middle two lines are added to guide the eye. At the bottom the surface of the heap is drawn, this data is used for further analysis, like fitting linear functions.

The shape of a heap is defined by its surface. In the case of a two dimensional heap

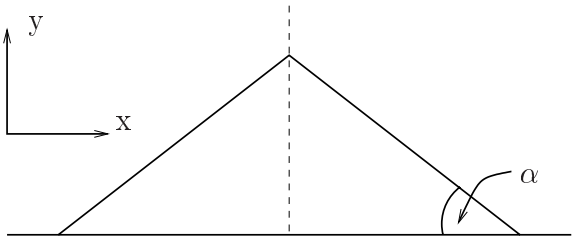


Figure 3.25.: Basic shape of a heap. The surface has an approximately constant slope and forms the so called angle of repose α with the horizontal bottom.

the surface can be described as a function $h(x)$, where h is the height at position x. To measure $h(x)$ the system is divided into bins of width Δx and $h(x_0)$ is taken as the maximum y coordinate of all particles between $x_0 - 0.5\Delta x$ and $x_0 + 0.5\Delta x$. Fig. 3.24 shows the result of this procedure compared to a picture of the same heap. Since the basic shape remains unchanged when the heap grows it is possible to rescale the shape at different times. If $h(x,t)$ denotes the shape at time t the following should apply:

$$\frac{1}{\sqrt{\gamma t_1}} h\left(\frac{x}{\sqrt{\gamma t_1}}, t_1\right) = \frac{1}{\sqrt{\gamma t_2}} h\left(\frac{x}{\sqrt{\gamma t_2}}, t_2\right) \tag{3.13}$$

The total mass $M = \gamma t$ of the system is proportional to the volume of the heap if the density is constant. Fig. 3.26 shows the growing process of a heap together with the rescaled shapes at different times. The curves collapse quite well on a single line, only for small heaps deviations are visible.

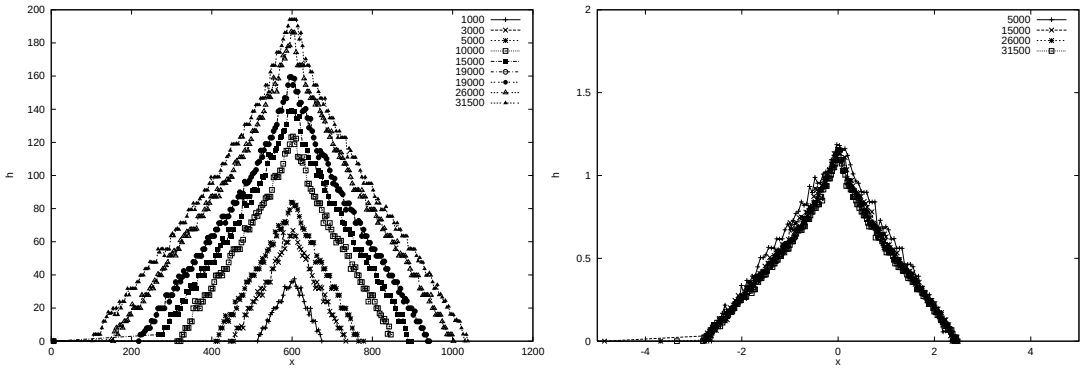


Figure 3.26.: Snapshots of a growing sandpile. The numbers denote the numbers of particles. On the right the data is collapsed as described in the text. The heap is steeper at the top, this is mainly caused by the small input velocity (see Fig. 3.32 and explanations there).

The angle of repose depends also on the micro-mechanical properties of the grains. The parameter of interest here is the coefficient of restitution r. In Fig. 3.27 we plot the

slope of a heap of 30,000 particles for r ranging from 0.97 to 0. The error bars indicate the statistical error as given by a least square fit of a straight line to the surface. Since the tail of the heap deviates from this line (see 3.5.1), only the straight central part is considered. If the underlying grid dominated the result we would see that the slope would preferably be an integer ratio like $\frac{1}{2}$, $\frac{1}{3}$. Regarding the slope there are no dominant lattice effects. However we observe artifacts at the point where particles are poured into the system. If the restitution coefficient is close to zero sometimes sharp peaks are visible that are two grid cells wide. It almost looks like the particles are stacked on each other. There are several reasons contributing to this behavior. First, there are no forces that would in reality push the particles apart under load. Second, within a cell the position of the particles are not considered for the particle interaction. Thus there is again no reason for them to be pushed out of a cell. Third, because the average horizontal momentum of the particles is zero there can be no horizontal movement once the kinetic energy has vanished. Finally one should keep in mind that within the framework of HSMC the particles have a scattering cross section but no shape. The association of the particles with spheres is strictly speaken arbitrary. Several optimizations are possible to reduce the lattice artifacts. First, the lattice could be rotated by 45 degrees to achieve a better adaption to the orientation of the heap surface. This improvement was successfully applied in LGA simulations of heaps [7]. Another possibility would be to apply a random displacement of the lattice between the time steps.

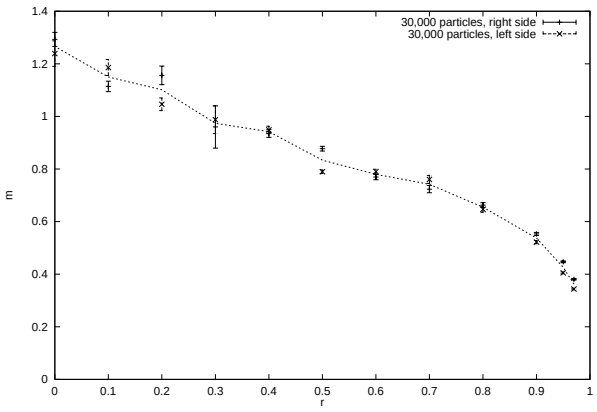


Figure 3.27.: Angle of repose α as a function of the coefficient of restitution ($m = \tan(\alpha)$).

3.5.1. Shape of a heap

Although the surface of a heap is close to a straight line, it is known that the tail of the heap deviates from that line. One possible model of a pile is the idea of incomplete layers of particles[7]. The surface is consequently not flat, but consists of kinks. If the top of the heap is located at $x = 0$ and its height is $h(0) = h_m$, an ideal right side of the

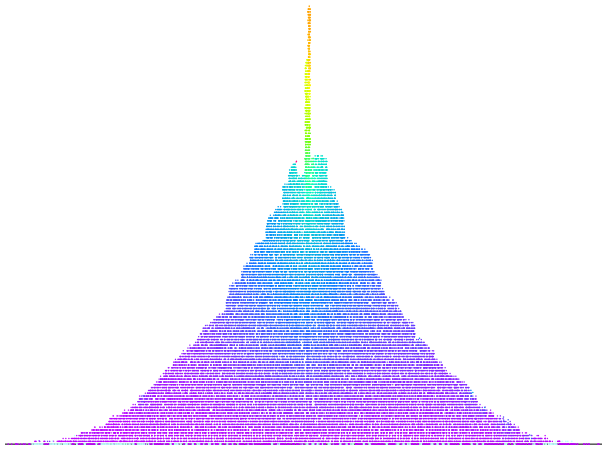


Figure 3.28.: Lattice artifact of HSMC. For small coefficient of restitution (here 0.0) sharp vertical peaks are visible.

heap follows the law $h(x) = h_m - mx$. Every kink increases this value of x by a value l, where l is typically of grain size. If ρ is the number of kinks per unit length in vertical direction we get the following differential equation:

$$\frac{dh}{dx} = -\frac{m}{1 + lm\rho}. \tag{3.14}$$

We can relate the density of kinks to the probability r of a particle to be stopped at a kink. The flux of moving particles $\Phi(x)$ at height h is diminished by this according to

$$d\Phi/dh = r\Phi\rho. \tag{3.15}$$

On a plate the heap grows by a horizontal shift of its surface. This is only possible if the aggregation rate of particles is independent of the height: $\partial\Phi/\partial h = B = const.$ Since $\Phi(0) = 0$ we obtain $\Phi = Bh$. Replacing Φ with Bh in Eq. (3.15) we get $\rho = 1/rh$ and thus

$$\frac{dh}{dx} = -\frac{m}{1 + lm/rh}. \tag{3.16}$$

With the solution

$$x = \frac{h_m - h}{m} + \frac{l}{r}\ln(\frac{h_m}{h}). \tag{3.17}$$

Fig. 3.29 shows a comparison of the simulation with formula (3.17). The logarithmic tail is reproduced very well, the HSMC method confirms the results of LGA simulations [7, 45].

The horizontal plate is a necessary boundary condition for the formation of a heap. In the direction perpendicular to gravity several boundaries are possible. So far the

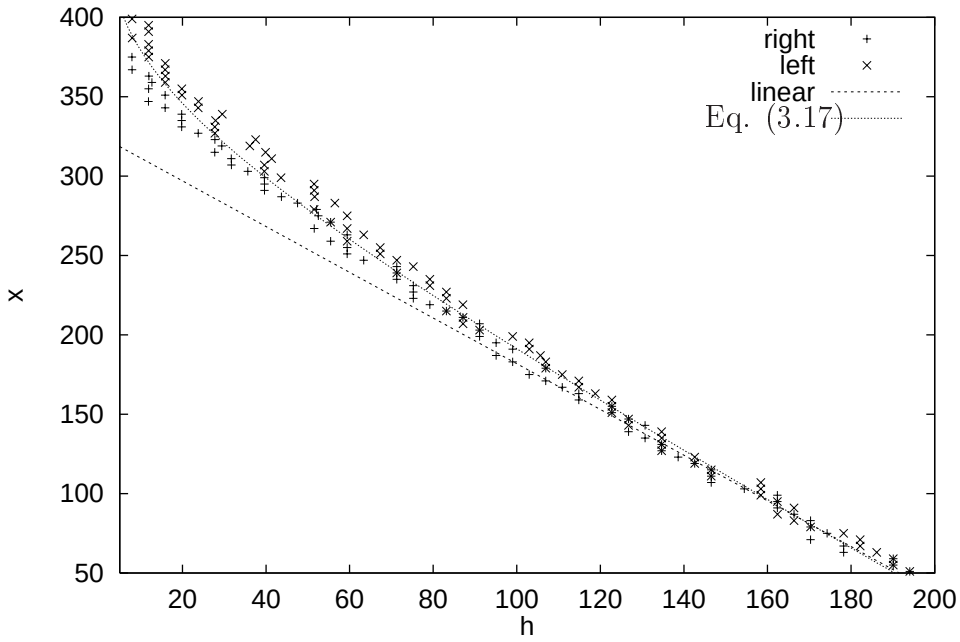


Figure 3.29.: Comparison of the shape of a heap on a plane with an analytical solution according to Eq. (3.17). $N = 30500$, $r = 0.9$

system was large enough to avoid an influence of these boundaries on the heap. Thus the result is the same as in a semi-infinite system. Another possibility is a system of fixed size like a box, and the third is a finite table (see Fig. 3.30 for a scheme of the three possibilities). The difference between a box and a table is that in the first all particles are contained inside the system whereas in the second every particles that reaches the horizontal border is removed ("falls of the table"). The different boundaries influence the shape of a heap. Fig. 3.31 shows the result for the discussed cases. The most striking difference is visible for the table. Instead of a logarithmic tail the surface is bent in the opposite direction. Near the boundaries the angle of repose is increased. A possible explanation is that the plate stabilizes the particles close to it. In the case of the infinite plate the bottom was covered with a thick layer of particles, effectively decoupling the plate from the particles at the top of this layer.

The final deviation from a straight line is visible at the head of the heap (see Fig. 3.32). Close to the top there is a transient regime where the initial velocity of the particles is still noticeable. Further down the acceleration due to the gravity and the dissipation due to collisions are balanced. If the particles are put into the system with a high velocity the head of the heap becomes flat, because the incoming particles will have a high probability to kick a resting particle out of the heap. The closer the local slope is to the maximum angle of stability the higher is the probability to release a particle. Therefore the slope is reduced until an equilibrium is reached.

The fact that the top of the heap becomes flater for high impact energies is known

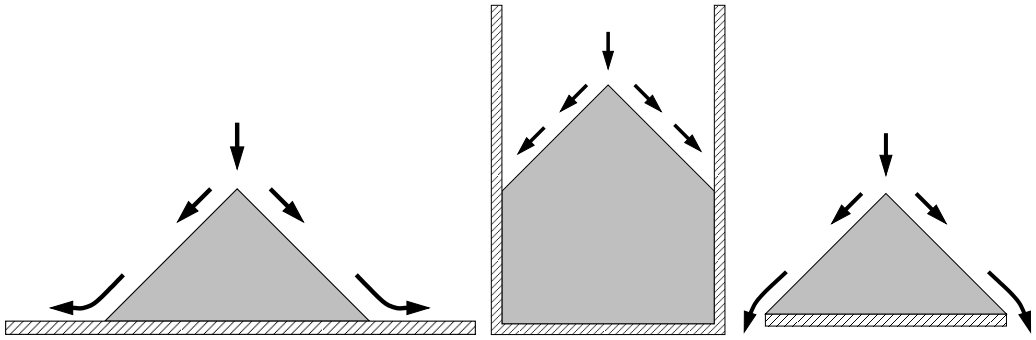


Figure 3.30.: Different boundary conditions for heaps. Semi-infinite table (left), box (middle) and finite table (right).

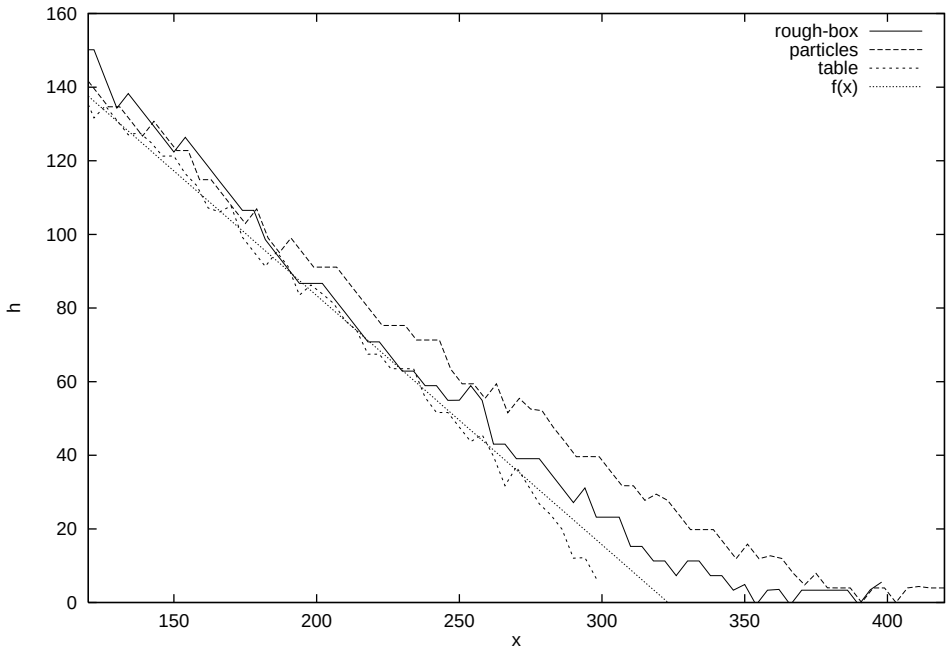


Figure 3.31.: Different shapes of heaps that result from different boundary conditions.

from experiments [33]. The artefact of the steeper surface for small impact energies relates again to the fact, that there is no steric interaction between the particles. A sufficiently high kinetic energy of the particles is necessary to force them out of a filled HSMC-cell.

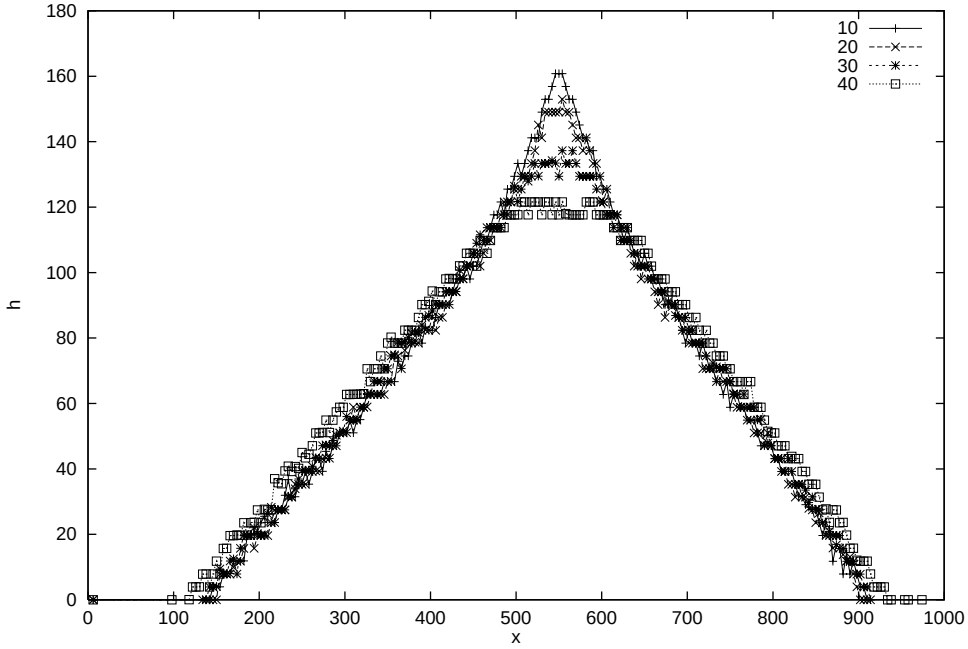


Figure 3.32.: Shape of the head of a heap for different pouring violence. The numbers are the vertical pouring speed in m/s.

In this section we have demonstrated that HSMC is able to simulate stable heaps due to the effects of excluded volume introduced on the grid level. It turned out that artifacts of this grid are only visible for very low coefficient of restitution. Besides the basic features of a heap like the constant slope of its surface more detailed structures like the logarithmic tail are also reproduced.

3.5.2. Suitability of other simulation methods

In the area of heaps molecular dynamics is the classical method, last but not least because it can treat several models of static friction. This is especially true for Contact Dynamics, where the Coulomb law of friction is incorporated. With MD it is possible to examine the effects of the particle shape on the properties of the heap[74].

LGA models have also been successful applied to simulate granular heaps[41], this includes the logarithmic deviation at the tail of the heap [7]. Care has to be taken that the inherent anisotropy of LGA models does not lead to artifacts in properties like the angle of repose.

With appropriate boundary conditions, e.g. particles at the bottom that have fixed positions, it is also possible to simulate heaps with hard sphere molecular dynamics. In this case the inelastic collapse problem (see chapter 2.5) has to be treated with an appropriate work around.

3.6. Correlations in dissipative gases

In this chapter we will take a closer look at the correlations existant in dissipative gases. The motivation to have a look into this can be seen in Fig. 3.34. For low dissipation ($r = 0.98$) there is perfect agreement between the predicted time evolution of the energy. For larger dissipation ($r = 0.85$) however DSMC and HSMC underestimate the kinetic energy compared to the results of an event driven simulation.

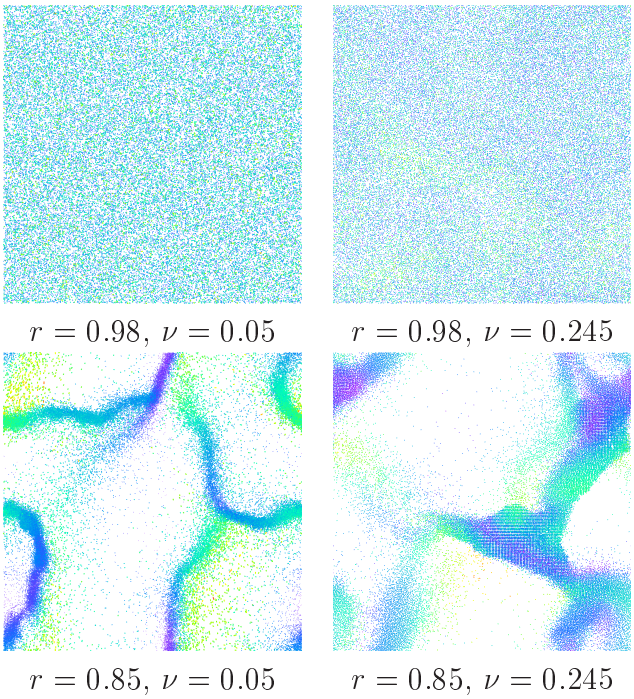
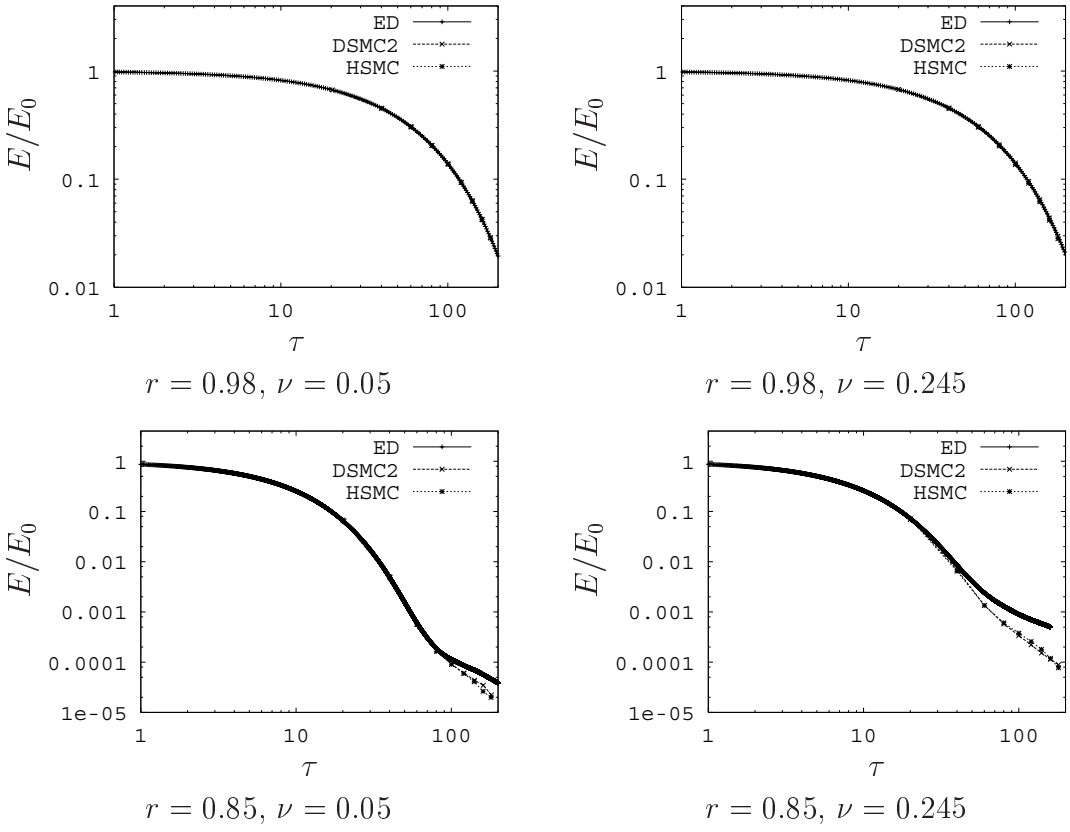


Figure 3.33.: Snapshots after $\tau = 200$ collisions per particle. The systems are characterized by the area fraction ν and coefficient of restitution r.

To test possible reasons for the discrepancies we have chosen four systems with different densities and dissipation. The systems with high dissipation ($r = 0.85$) show the clustering instability, whereas the two systems with $r = 0.98$ stay homogeneous. Fig. 3.33 shows a snapshot after every particle participated in 200 collisions. The area fraction $\nu = 0.05$ of the dilute system was chosen to make it well suited for the Monte Carlo Methods. The denser systems ($\nu = 0.245$ are still accessible for extensions to the theories of dilute systems, another area where the interest for this type of correlations stems from. All systems consist of $N = 50000$ particles. Throughout this chapter we use the number of collision per particle τ as natural time scale. Because one collision involves two particles it is related to the total number of collisions C in the system according to $\tau = 2C/N$.



$r = 0.98$, $\nu = 0.05$ $r = 0.98$, $\nu = 0.245$

$r = 0.85$, $\nu = 0.05$ $r = 0.85$, $\nu = 0.245$

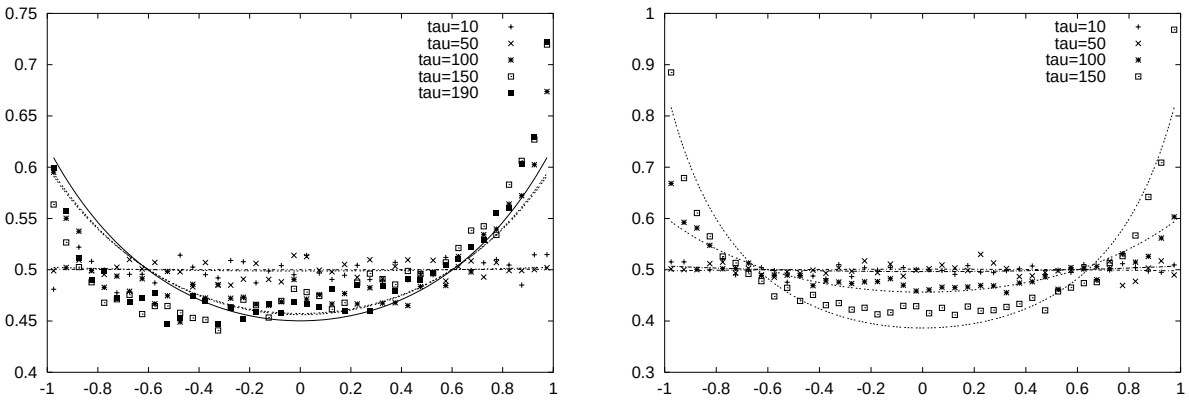
Figure 3.34.: Time evolution of the energy for the four different discussed systems.

3.6.1. Impact Parameter

The major assumption of both DSMC and HSMC is that the probability distribution $p(b)$ of the impact parameter b is constant. Since in the simulations only the normal relative velocity of two particles is damped, more energy is dissipated if the number of central collisions is overestimated. Fig. 3.35 shows the distribution measured with ED at different times. As soon as the evolution of the energy deviates ($\tau \approx 80$ for $\nu = 0.05$, and $\tau \approx 30$ for $\nu = 0.245$) we also observe deviations from the constant probability distributions. There are more grazing collisions than assumed by the Monte Carlo simulations. In order to correct the distribution used in the simulations we approximate the measured values with

$$p(b) = \frac{a}{1 - cb^2} \quad \text{with } b \in [-1, 1]. \tag{3.18}$$

The lines in Fig. 3.35 show the result of least square fits of expression (3.18). The limit $c \to 0$ corresponds to the constant distribution $p(b) = 0.5$. The coefficient c indicates therefore the deviation from the constant distribution. Fig. 3.36 shows c versus τ. In consistency with the energy data we see that c starts to deviate from zero at $\tau \approx 80$ for $\nu = 0.05$ and $\tau \approx 40$ for $\nu = 0.245$. With the knowledge of $c(\tau)$ we can use $p(b, \tau)$ in



$r = 0.85$, $\nu = 0.05$ $\qquad$ $r = 0.85$, $\nu = 0.245$

Figure 3.35.: Distribution of the impact parameter b for two systems showing clustering.

the Monte Carlo simulations to draw the impact parameter from the correct distribution and achieve a higher accuracy. In Fig. 3.36 the simulations using the adjusted $p(b, \tau)$ are labeled "corrected". For the dilute system ($\nu = 0.05$) there is a slight improvement for medium times ($\tau = 80 \ldots 120$). For the dense system the results of HSMC are closer to the ED results for all times, but deviations are still strong. For the DSMC2 simulations the dissipation is even increased. Because we examine $E(\tau)$ and not $E(t)$ an incorrect number of collisions is not a possible reason. Instead the average energy loss in a single collision must be wrong. This issue will be discussed in the following.

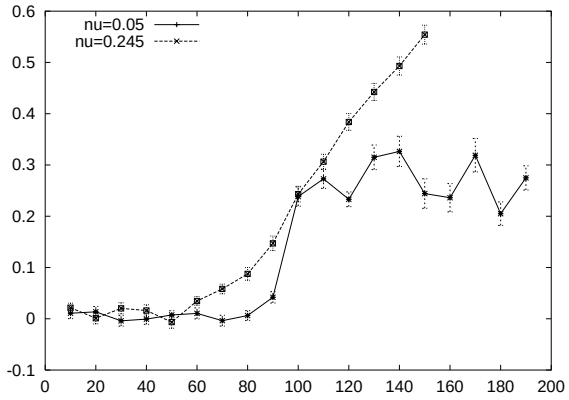
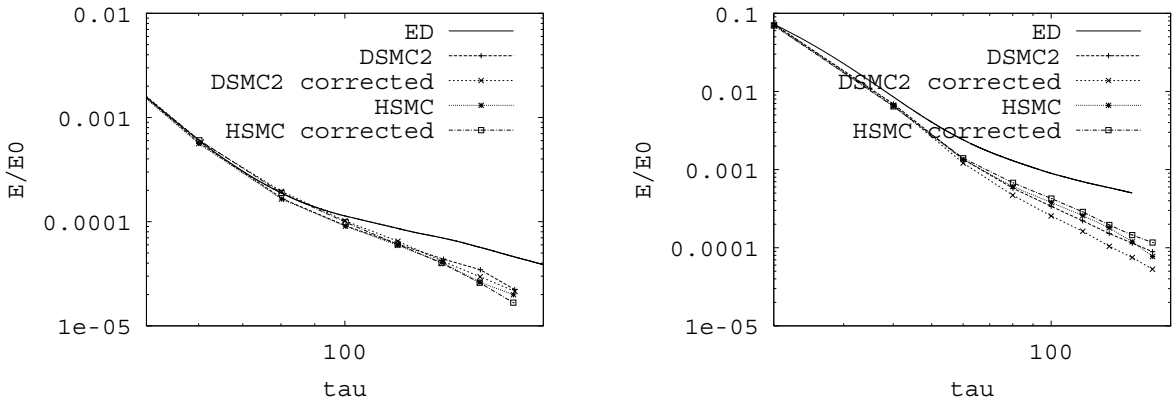


Figure 3.36.: Time evolution of the coefficient c from Eq. (3.18) for the systems with coefficient of restitution $r = 0.85$.

3.6.2. Density and Velocity Correlations

The outcome of a collision with fixed coefficient of restitution depends on two parameters. First the impact parameter and second the relative velocity. The correction of the first did only slightly improve the results. We therefore conclude that the relative velocity of particles is different in the Monte Carlo Simulations. Among others we will therefore take a more detailed look at the velocity correlation in the next section.



$r = 0.85$, $\nu = 0.05$ $r = 0.85$, $\nu = 0.245$

Figure 3.37.: Comparison of the time evolution of the energy for simulations with and without corrected impact parameter distribution.

The measured quantities are the spatial density correlations

$$G_{nn}(\mathbf{r}, t) = \frac{1}{V} \int d\mathbf{r}' < \delta n(\mathbf{r} + \mathbf{r}', t) \delta n(\mathbf{r}', t) > \tag{3.19}$$

and velocity correlations [120]

$$G_{\alpha\beta}(\mathbf{r},t) = \frac{1}{V}\int d\mathbf{r}' < u_\alpha(\mathbf{r}+\mathbf{r}',t)u_\beta(\mathbf{r}',t) > . \tag{3.20}$$

The rank two tensor field $G_{\alpha\beta}(\mathbf{r},t)$ has the transverse ($\perp$) and parallel components ($\|$)

$$G_\| = \hat{\mathbf{r}_\alpha}\hat{\mathbf{r}_\beta}G_{\alpha\beta}(\mathbf{r},t) \text{ and} \tag{3.21}$$
$$G_\perp = \hat{\mathbf{r}_{\perp\alpha}}\hat{\mathbf{r}_{\perp\beta}}G_{\alpha\beta}(\mathbf{r},t), \tag{3.22}$$

where $\hat{\mathbf{r}}$ and $\hat{\mathbf{r}}_\perp$ are unit vectors, parallel and perpendicular to the distance vector $\mathbf{r}$ between two particles [96]. $G_\|(r)$ is positive when the particles move into the same direction and negative when they move in opposite directions. Because for $G_\perp$ the velocities of the particles are projected onto a vector perpendicular to the distance vector between the particles $G_\perp$ is a measure for rotation/vorticity.

First we check how the correlations look like when the results for the kinetic energy of the systems agree. As expected there are almost no correlations in the dilute system with small dissipation ($\nu = 0.05$, $r = 0.98$) as can be seen in Fig. 3.38.

The fact that the dense system with low dissipation ($\nu = 0.245$, $r = 0.98$) stays homogeneous in the course of the simulation is visible in the density correlation (Fig. 3.39) and the snapshot of the final configuration (Fig. 3.33). The higher density, however, leads to visible correlations in the velocity. Particles that are close to each other have the tendency to move into the same direction. This is a consequence of the fact that the relative motion leads to collisions and is therefore damped. HSMC and DSMC2 show larger values for $G_\|$ than ED but the artificial correlations are small enough to prevent significant consequences to the overall dissipation. For the systems with high dissipation the situation is more complex. Not only that there are significant differences between the simulation methods. We also have to judge the effect of the corrected distribution of the impact parameter. The results of the simulations with this distribution function are again labeled "corrected".

First we look at the dilute system (see Fig. 3.40). The clustering is of course clearly visible in the density correlations. All Monte Carlo Methods tend to overestimate the density correlation G_{nn}. Especially the DSMC methods results in too dense clusters. Changing the impact parameter distribution does not improve the results for G_{nn} in the case of DSMC. The limitation of the maximum density of the HSMC methods yields also better results for G_{nn}. This is improved to an almost perfect agreement with ED results if the impact parameter distribution is corrected. For HSMC this effect is smaller and even reduced further with the corrected impact parameter. Looking at the plots of $G_\|$ and $G_\perp$ it becomes obvious, that the unmodified HSMC only improves the results for the density correlations. However, combining the corrected impact parameter distribution with HSMC yields results that are very close to the ED results. The results of the two DSMC simulations show, that the corrected $p(b)$ alone does result in only a minor improvement of the Monte Carlo results.

The results of G_{nn} for the dense ($\nu = 0.245$) system with high dissipation ($r = 0.85$) show the limitation of the improvements. All Monte Carlo results deviate from the ED

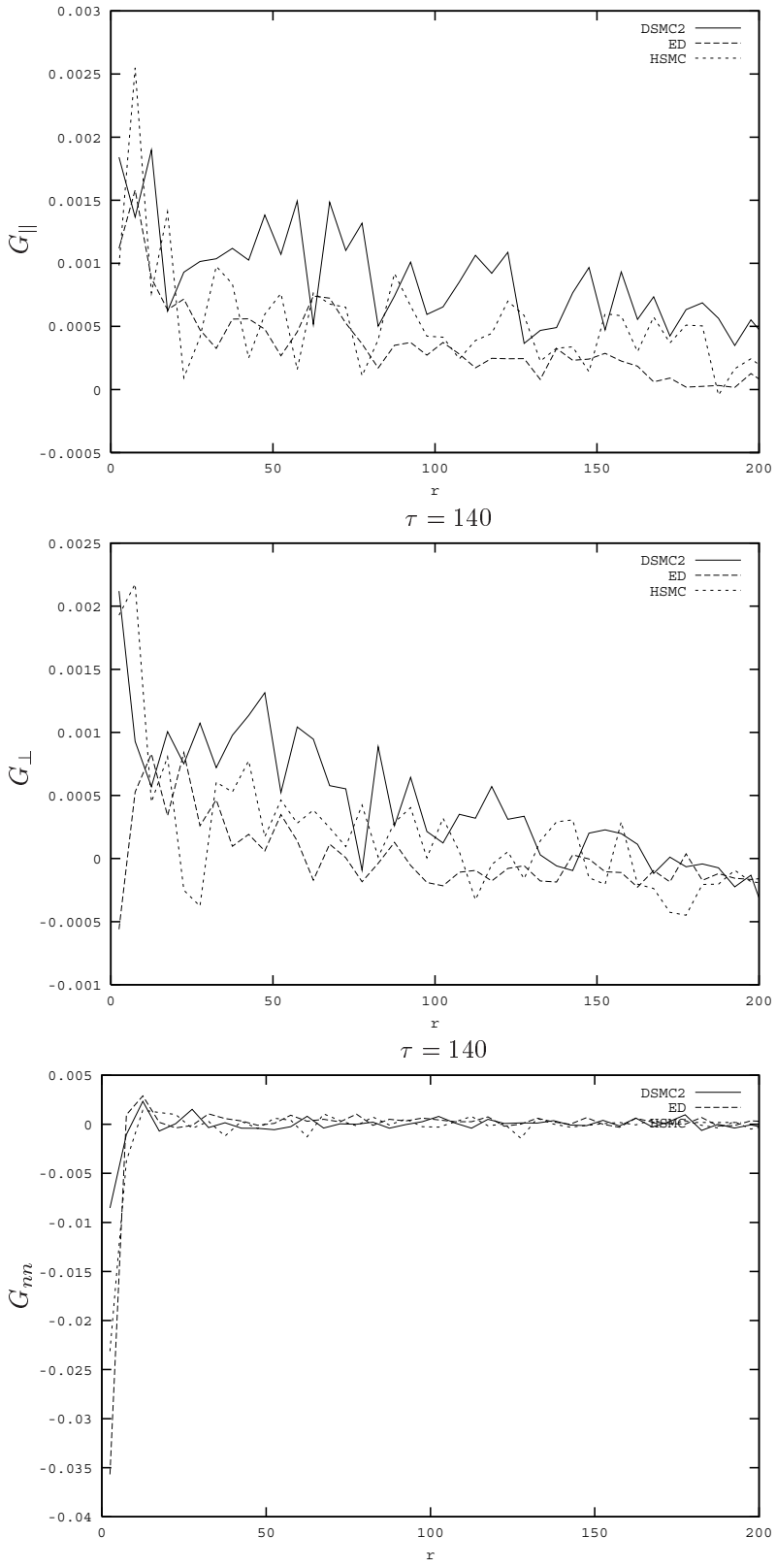


Figure 3.38.: Density correlation G_{nn} and velocity correlations $G_{\parallel}$, $G_{\perp}$ at $\tau = 140$ for a system with $N = 50000$ particles and $\nu = 0.05$, $r = 0.98$. Although there are no visible density correlations G_{nn}, there are weak velocity correlations due to the dissipation.

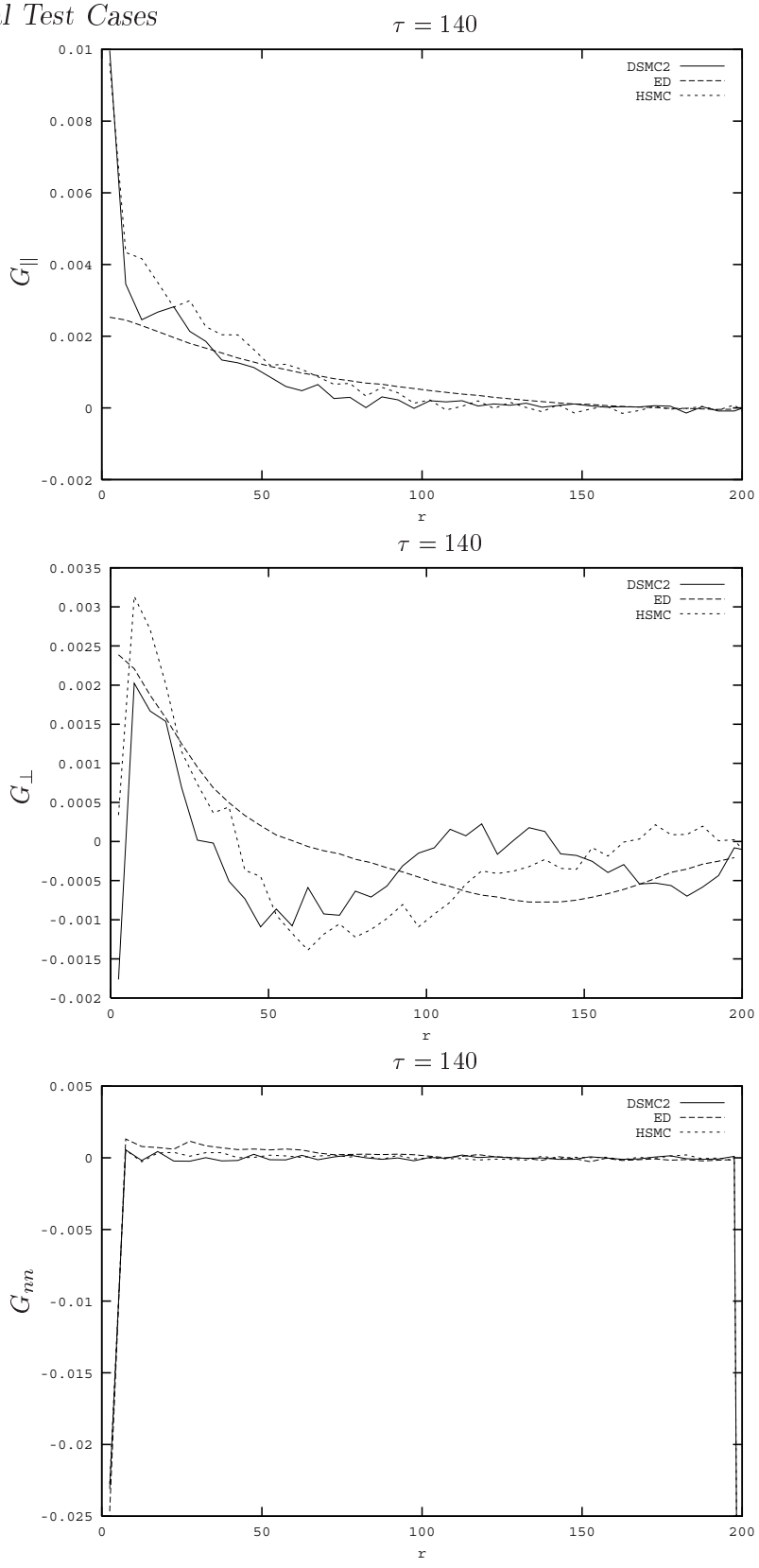


Figure 3.39.: Density correlation G_{nn} and velocity correlations $G_{\parallel}$, $G_{\perp}$ at $\tau = 140$ for a system with $N = 50000$ particles and $\nu = 0.245$, $r = 0.98$. DSMC and HSMC reproduce the velocity correlations shown by ED simulations, but there are small deviations.

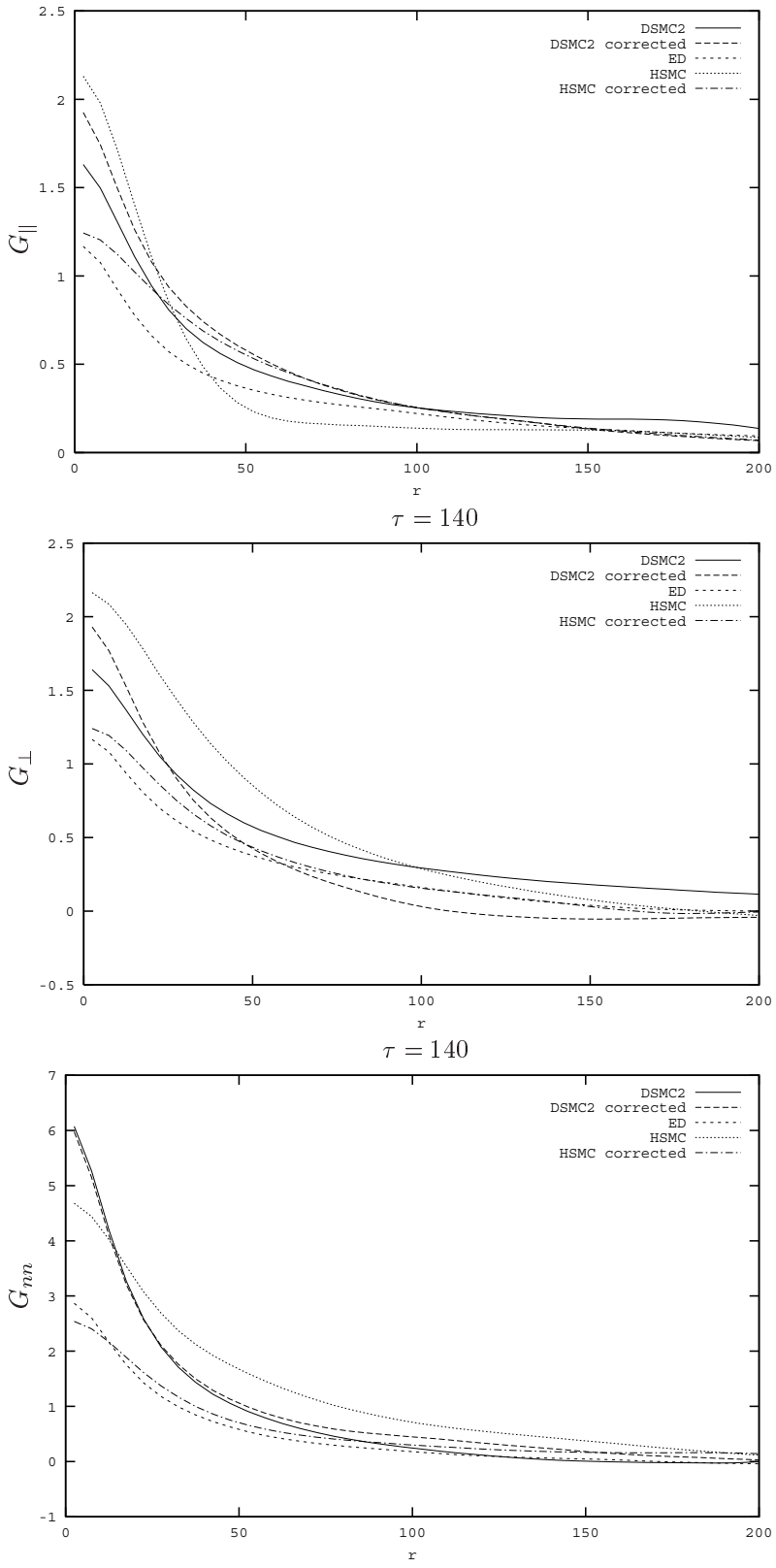


Figure 3.40.: Density correlation G_{nn} and velocity correlations $G_{\parallel}$, $G_{\perp}$ at $\tau = 140$ for a system with $N = 50000$ particles and $\nu = 0.05$, $r = 0.85$. There are strong deviations between DSMC/HSMC and the ED results. The HSMC with the corrected impact parameter distributions (labelled HSMC corrected) shows very good agreement with ED.

simulation. The deviation from ED is larger than the deviations within the various MC methods. The first obvious disagreement between ED and MC is the underestimation of both velocity correlations ($G_{\parallel}$ and $G_{\perp}$). As a consequence the dissipation is overestimated which results in higher values of G_{nn}, because the clustering is increased.

In the summary, HSMC together with the correction of the impact parameter distribution $p(b)$ delivered the best agreement with the results of the hard sphere molecular dynamic. The unmodified HSMC will also give better results for the density correlations than DSMC, but the velocity correlations are not correctly reproduced as soon as the system gets denser. Increasing the dissipation for the dense system shows, that the simple modification of $p(b)$ only results in a first approximation of the velocity correlations, which is no longer correct for stronger correlations.

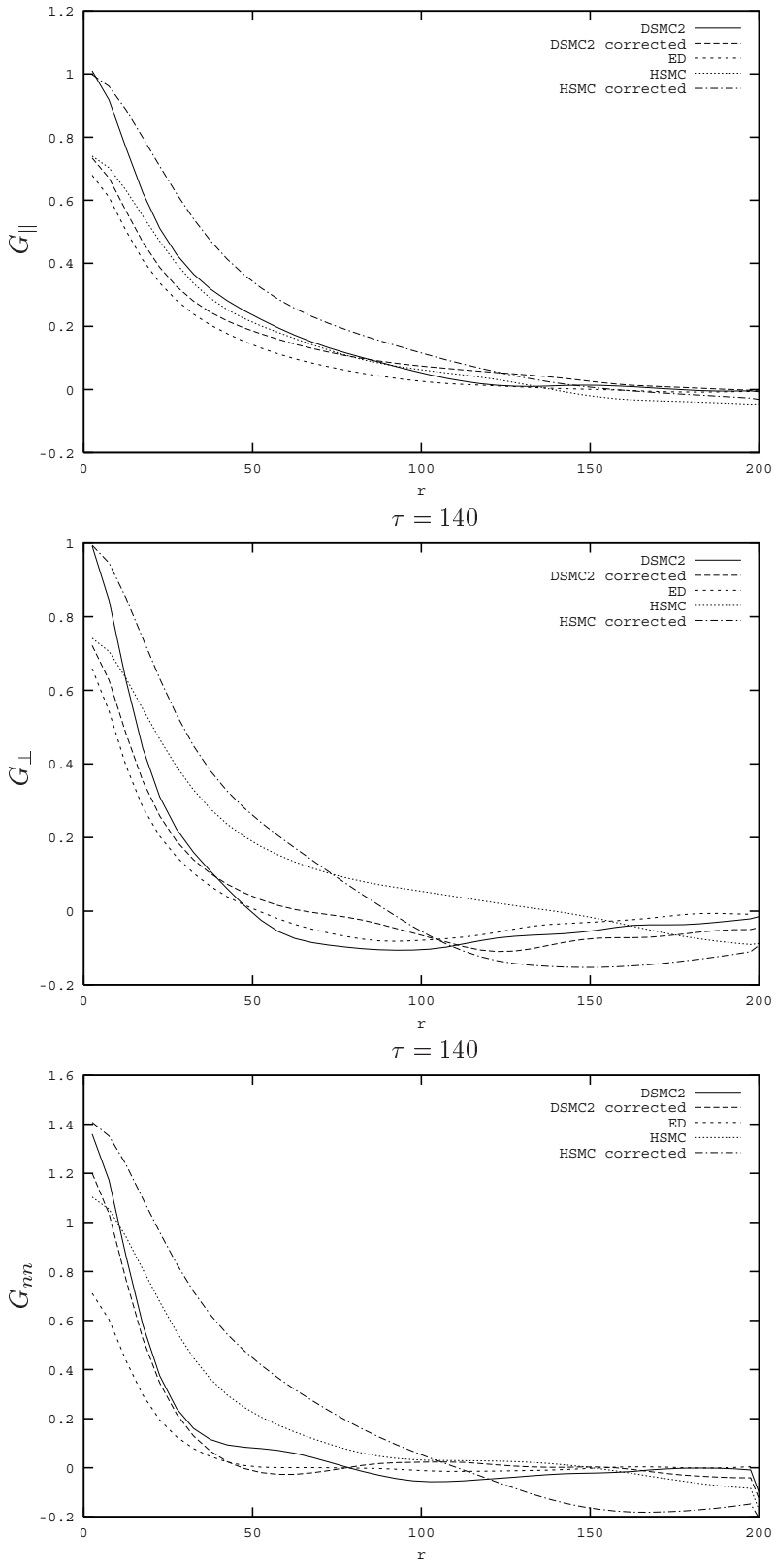


Figure 3.41.: Density correlation G_{nn} and velocity correlations $G_{\parallel}$, $G_{\perp}$ at $\tau = 140$ for a system with $N = 50000$ particles and $\nu = 0.245$, $r = 0.85$. The deviation between ED and the HSMC method with corrected impact parameter distribution (labelled HSMC corrected) show the limitation of the improvements.

3.7. Vibrated beds

In this chapter we focus on granular media in vibrated containers in two and three dimensions. If dissipation is not too strong and enough energy is fed into the system, e.g. via vibration of the container, the surface of the material may fluidize [17, 66] and the energy scales with the typical velocity of vibration rather than with the typical acceleration [13, 66, 113]. Simulations in 1D [17, 66, 75] were complemented by two dimensional simulations [63, 70] and experiments [13, 126]. In 1D and 2D the potential (or kinetic) energy E scales with the typical velocity V, i.e. $E \propto V^\alpha$, with $\alpha = 2$ in 1D and $\alpha \approx 1.4$ in 2D. The latter is also obtained from experiments [126], consistent with simulations, with respect to the fact that they lead also to an exponent $\alpha < 2$. Theoretical approaches usually lead to the exponent $\alpha = 2$ in 1D and 2D, see Ref. [61] and references therein.

In the following we introduce the numerical methods used and establish coincidence in 2D and 3D. We will discuss different boundary conditions and their effect on the exponent α in 2D and 3D. We are interested in the influence of the boundary because it always exists in the experiments and often determines the behavior.

3.7.1. Simulation Aspects

The container is made of a horizontal bottom, two infinitely high vertical walls, and it is open to the top. The position of the bottom is $z_0(t) = A\sin(2\pi ft)$ as a function of time t, with the amplitude A and the frequency f, so that its maximum velocity is $V = 2\pi Af$. Before we start the simulations, N particles with diameter d are filled into the container with random initial positions and velocities. We assure that the system reaches a steady state before performing averages.

Throughout this chapter three different methods are compared. ED (event-driven) labels the results of classical Hard Sphere Molecular Dynamics. DSMC stands for the unmodified Direct Simulation Monte Carlo Method. DSMC2 [90] contains two modifications: the number of collisions is modified due to the excluded volume of the particles, and the additional advection process of the CUBA method is applied [2, 30].

3.7.2. Results

Comparison of DSMC and ED in 2D

The height of the center of mass is used to measure the energy in the system. To evaluate the validity of the DSMC method for this application we first compare ED, DSMC and DSMC2 in two dimensional simulations. In figure 3.42 we plot the height of the center of mass H vs. velocity V for elastic walls and inelastic particles as well as the particle number density as a function of height for f=100Hz and V=0.11m/s. At low excitations of the bottom plate ($V < 0.6$m/s) the DSMC method is not longer valid. However, the improved DSMC2 method still gives good agreement with ED down to $V = 0.06$m/s.

In this case the density is about 84% of the density the system has at rest. The much better agreement of DSMC2 with ED can also be seen in the density distribution.

In figure 3.43 we plot the probability distribution for the horizontal (U_x) and vertical (U_z) velocities. Again we see agreement of DSCM2 and ED. The mean square velocity is larger for elastic compared to dissipative walls and the distribution of the vertical component is asymmetric with the maximum shifted towards negative velocity. The decay for positive velocities is slower, due to the dissipation in the system and a net energy flux from the bottom upwards.

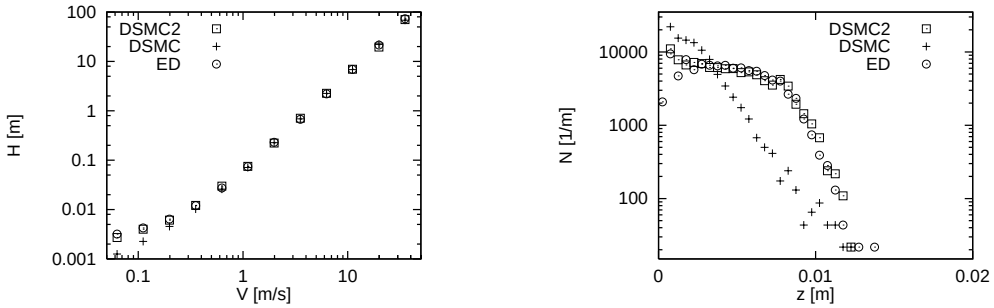


Figure 3.42.: Comparison of the three different methods: DSMC2, DSMC and ED. Left: height of the center of mass vs. V. Right: density distribution for $V = 0.11\mathrm{m/s}$.

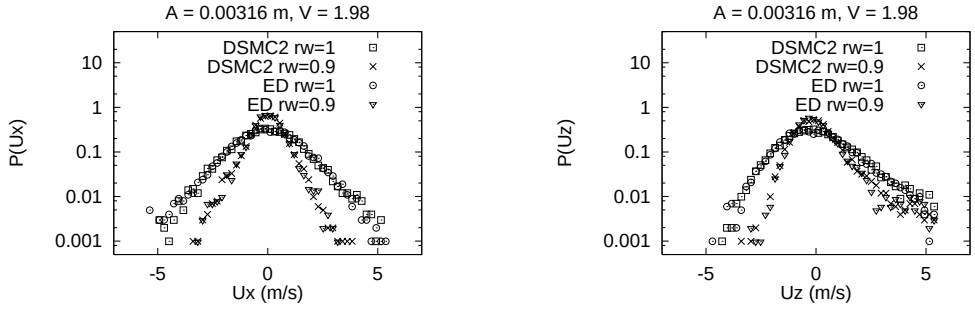


Figure 3.43.: Semilog plot of the probability distribution $P(U)$ against the horizontal velocity U_x (left) and the vertical velocity U_z (right).

Different boundary conditions in two dimensions

The scaling behavior was investigated like in Refs. [63, 70]. The results of DSMC2 are in reasonable agreement with the previous results of ED (see figure 3.44). This is true for the overall scaling and for the deviation from the theoretically expected power law $H \sim V^2$ at small V values. This proves that the reason for these phenomena is neither long range correlation nor a likewise multiple particle memory effect, since both are not explicitly included in DSMC2.

To understand the reason for the strange dependency of H on V with dissipative particles and walls we plot $H(V) - H(0)$, with the height of the center of mass at rest $H(0) = 2.492 \times 10^{-3}$m, for all possible combinations of r and r_w. The agreement between DSMC2 and ED for all boundary conditions shows that the particle particle and particle wall interaction is represented correctly. For elastic walls and dissipative particles we find $\alpha \approx 2$ above $V \approx 0.6$m/s. For inelastic walls we observe a smaller $\alpha \approx 1.5$ over two decades, whereas we see two different regimes in the case of elastic particles and inelastic walls. In the latter case we find $H \sim V$ for small V and $H \sim V^2$ for large V. For $V > 5$m/s the density is low enough that many collisions with the walls occur before the next particle collision. Therefore the horizontal velocity of a particle decays between two particle collisions if the walls are inelastic. Thus the dissipation of energy per unit time through the walls decreases due to decreasing collision frequency. Particle particle collisions are necessary to trigger dissipation and the system behaves similar to a system with dissipative particle collisions.

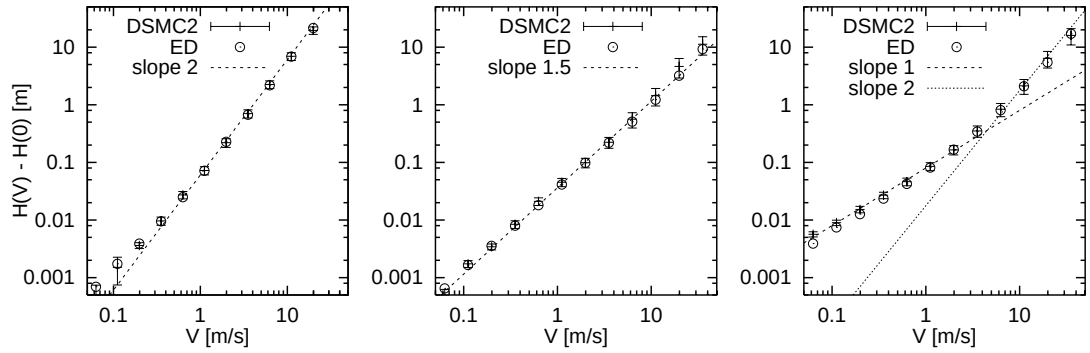


Figure 3.44.: Reduced height $H(V) - H(0)$ with different dissipation combinations. Left: $r_w = 1, r = 0.9$, middle: $r_w = 0.9, r = 0.9$, right: $r_w = 0.9, r = 1$. The error bars indicate the standard deviation.

Investigation in three dimensions

After the agreement between ED and DSMC2 is established in 2D, we can now investigate the behavior in 3D. In order to obtain densities comparable to 2D we increase the number of beads by a factor 10, because the system is 10 diameters wide and deep. We estimate the height of the center of mass at rest to $H_{3D}(0) = 2.2 \times 10^{-3}$m using an ED simulation with dissipative beads and a fixed bottom plate.

We do not find new qualitative aspects of the system's behavior when we make the transition from 2D to 3D irrespective of the particular boundary condition chosen. Thus effectively, we recover the physics of a 2D system, because of the equivalence of the two horizontal dimensions.

In this chapter the DSMC method was applied to dry granular media simulations. Reasonable quantitative agreement between the deterministic ED method and the partially stochastic DSMC2 algorithm is obtained. This proves that the assumptions made

for DSMC are correct in the parameter range discussed here and that the behavior of the system does not depend on possible correlations between collisions.

4. Computational Issues

4.1. Metacomputing

To increase computing power beyond today's limits several supercomputers can be connected to form one huge metacomputer. Not only the hardware of the participating computers is important but also the software and the data transfer must be optimized. A number of projects aim to couple computers in various ways in order to gain higher performance [15, 23, 24, 25, 59, 111]. During Supercomputing '98 a standard for coupling computers with an interoperable MPI (message passing interface) was proposed [18]. PACX-MPI[111] is a library that provides this interoperability and we used it during the last couple of years during the participation in the testbed described below [89, 101, 100]. The library is called PACX-MPI, because it extends MPI to couple parallel computers (PACX is an acronym for PArallel Computer eXtension).

In the frame of the G7 Global Information Society Initiative "Global Interoperability of Broadband Networks" the High Performance Computing Center Stuttgart (HLRS) and the Pittsburgh Supercomputing Center (PSC) have set up a transatlantic wide area application testbed. A dedicated ATM link was installed between the two sites[110]. This testbed allowed to gain experiences that led to substantial improvements in the PACX-MPI library and in the applications.

4.1.1. Description of `PACX-MPI`

PACX-MPI is an implementation of the message passing standard MPI which aims to support the coupling of different platforms and to bridge the gap of interoperability between MPI libraries of different vendors. Therefore PACX-MPI is optimally adapted to the two-level communication hierarchy of metacomputing.

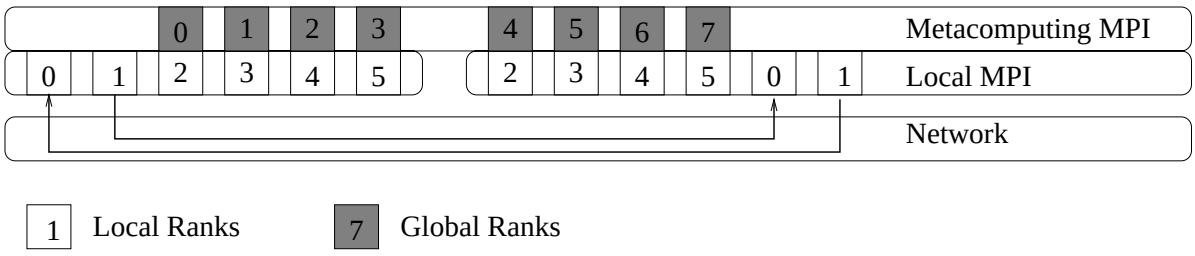


Figure 4.1.: Concept of the PACX-MPI metacomputing library.

Due to the existence of both communication levels internal and external operations are distinguished. Internal operations are operations which remain inside a single machine. They are executed by using the vendor-implemented MPI-library, because it is highly optimized.

External operations, e.g. point-to-point operations between two nodes on different machines, are handled using standard protocols, currently TCP/IP. In this sense PACX-MPI can be described as a tool to provide multi-protocol MPI for metacomputing.

PACX-MPI uses two specialized daemon nodes for the external communication on each machine. These two nodes are transparent for the application and are therefore not part of global communicators, like e.g. MPI_COMM_WORLD. The mapping of local MPI ranks to global ranks including the daemon nodes (nodes 0 and 1) is shown on Fig. 4.1.

The concept of PACX-MPI includes also data-compression for the external communication to improve throughput on low bandwidth connections and data-conversion to support heterogeneous clusters. These points together with the full concept of PACX-MPI are discussed extensively by Gabriel et al. [28].

Point-to-point communication in PACX-MPI

A point-to-point operation in this concept is realized as in Figure 4.2. The sending node has to check whether the receiver is on the same machine. If this is the case, it can make use of the fast communication subsystem of the machine. This means the library can call the MPI_Send command directly using the native MPI-library. If this is not the case, i.e. the receiving node is on another machine, it sends the message to a local daemon node. The daemon node transfers the message to the destination machine, where another daemon node receives the message and hands it out to the destination node. This way, all communication is bundled across only one connection link for each direction.

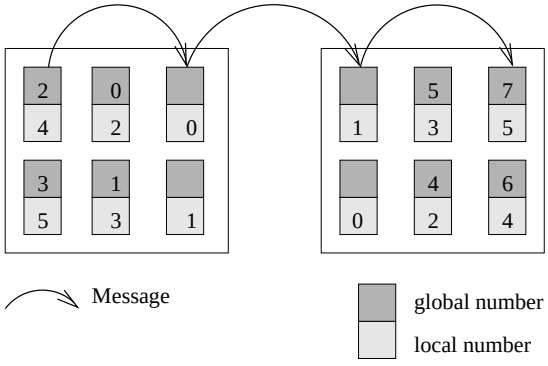


Figure 4.2.: Example for a point-to-point communication from global node 2 to global node 7

Collective communication in PACX-MPI

The use of the MPI API offers the possibility to optimize the algorithms used for collective operations for metacomputing. The algorithms applied in PACX-MPI try to minimize the use of the slow external network as much as possible [29]. On the local machine the implementation is again build on top of the vendor MPI collective routines.

To give an example we describe the algorithm of the MPI broadcast operation which is optimized to minimize the usage of the weak link between the machines. The algorithm consists of the following steps:

- Determine on each machine a so called local root node, which has to perform some additional work during the broadcast operation
- The root node of the broadcast operation sends a message to each local root node
- Each local root node distributes the data to the nodes on its machine, using the local broadcast operation of the vendor MPI library.

4.1.2. Description of the Application

The first application that was used in the metacomputing experiments was the DSMC code, that used as a starting point to develop the HSMC algorithm. Another applications that was adapted for metacomputing was a molecular dynamic program for short range interactions. The parallel paradigm applied here is domain decomposition (see Fig. 4.3) and message passing with MPI. Therefore every CPU is responsible for a part of the domain and has to exchange information about the border of its domain with its adjacent neighbors. Instead of exchanging them directly with all 8 or 26 neighbors in two or three dimensions respectively the Plimpton scheme [102] is applied here (see Fig. 4.4). Both applications have been used to simulate particle numbers that have been beyond the reach of a single computer [47].

4.1.3. Porting to a Metacomputer

In principle porting to a metacomputer is straightforward as long as one uses a library providing a single system image. PACX-MPI does this by replacing the standard MPI library. The only thing one has to do is to change the `include` path of your compiler to make it find the PACX-MPI header instead of the standard MPI header and link the PACX-MPI library in addition to the MPI library. Because PACX-MPI is still in development the set of supported MPI calls is however restricted to a subset of functions. If an unsupported function is used, one has to find a workaround by using other MPI calls. In the applications described here the MPI call `MPI_Cart_create` was used. Since PACX-MPI did not support this function, the provided functionality of generating a Cartesian grid with the information about connections between neighbors had to be implemented in the application itself. Other calls like `MPI_Ssend` were replaced by the standard `MPI_Send`. Details depend of course on the application. Future versions of

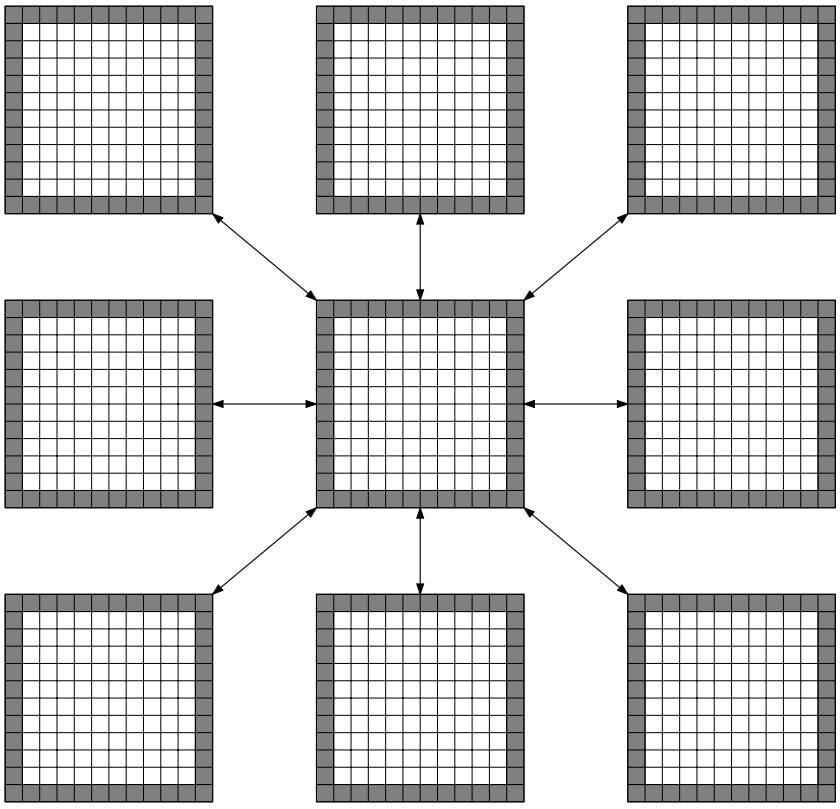


Figure 4.3.: Domain decomposition. The cells of the linked cell algorithm containing data that has to be shared with the neighbors (*shadow cells*) are in gray.

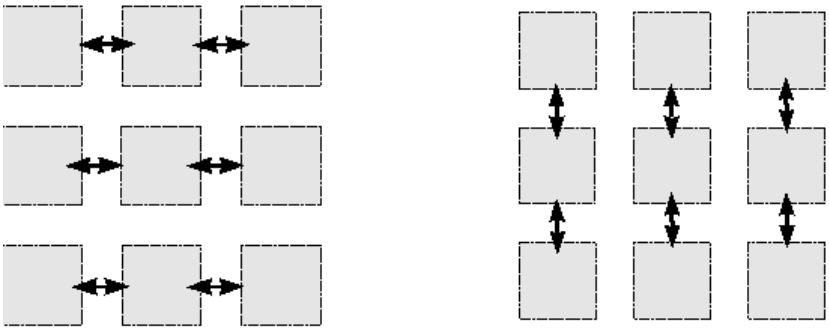


Figure 4.4.: Plimpton scheme to exchange *shadow cells*.

PACX-MPI will support more MPI calls, but even if all calls were implemented there might still occur problems. The simple reason is that on a metacomputer an application will run in a different way than on a ordinary parallel computer. Messages are sent different ways, latency and bandwidth vary across several orders of magnitudes within the metacomputer. This makes it different from almost all other parallel computers. Furthermore, most native MPI implementations are rather forgiving to the MPI user, i.e. it might be safe to reuse request handles without calls to `MPI_Wait` because the communication is always finished at the time of reuse. On a metacomputer it is much more likely that this does not hold. However, there should be no problem if the program is strictly conforming with the MPI standard.

4.1.4. Drawbacks of a metacomputer

Metacomputing introduces more stringent requirements on the communication pattern of the application, because latency and bandwidth are aggravated. An ordinary parallel computer is designed to deliver high performance for a wide range of communication patterns. Since this is not true for a metacomputer it is very important to know the exact details of the communication. The Plimpton scheme used in the MD application offers a very regular pattern that can also be controlled and adapted to metacomputing needs as described in the following.

Latency

Latency is defined to be the time a zero byte message needs to reach its destination. On the Cray/SGI T3E this time is around 18μs. On a metacomputer it depends on the interconnection between the computers and the protocol used. For PACX-MPI using TCP/IP via the external I/O interface on a T3E the latency grows to 4ms and coupling a T3E in Pittsburgh with its counterpart in Stuttgart adds another 70ms. This is a difference of more than three orders of magnitude. At a first glance this seems to be the major problem of metacomputing.

Library Viewpoint There is not much the library can do about latency. A great part (20ms) is due to the speed of light. Some more overhead is added by the routers used. Only 4ms are spent sending the messages to and from the PACX-MPI nodes and in the TCP/IP implementation of the hosts. This minor part could be circumvented by the use of a different protocol or other improvements in PACX-MPI.

More important than reducing the latency is to deliver support for asynchronous communication. PACX-MPI does this in several respects. First, the basic calls like `MPI_Isend` and `MPI_Irecv` are supported. Secondly calls to `MPI_Iprobe` that check for incoming messages are implemented with native MPI calls and thus do not have a large latency as it would be the case if a TCP/IP call had to be made. Furthermore large TCP windows are used to allow asynchronous communication between the participating computers.

Application Viewpoint To be able to run at a good level of performance on a metacomputer an application has to implement latency hiding. After initiating the communication it should compute parts of the results and afterwards wait for the completion of all communications that hopefully are finished at that time.

To achieve this, the communication was grouped in two sections. First the communication for the first dimension is initiated by calls to `MPI_Isend`. In a second stage this communication is completed and the calls for all remaining dimensions are performed, allowing a partial overlap of communication with computation. In this application the force calculation for particles that interact only with particles of the core domain is performed between the first and the second stage. If the calculation for one particle takes about 100μs one needs around 750 particles to hide the latency. As soon as latency hiding is implemented this is no major restriction anymore. If the calculation for a single particle is considerably faster, like in the DSMC calculations [88] one needs more particles to hide the latency. Fig. 4.5 shows a comparison of a simulation running on a metacomputer using PACX-MPI to a simulation running on the same number of nodes on a single machine using native MPI. For particle numbers above 15.000 per CPU there is no runtime difference between the two implementations.

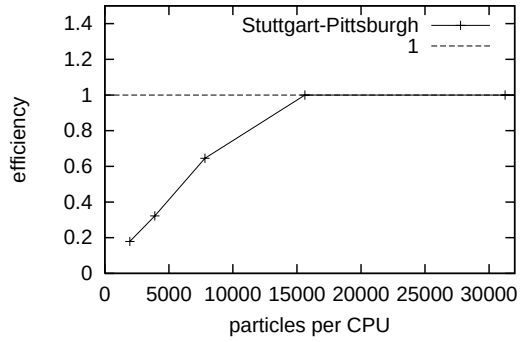


Figure 4.5.: Latency hiding for a DSMC application. The efficiency is defined as the performance of the metacomputing run with n1+n2 CPUs compared to the native MPI run with n1+n2 CPUs on a single host.

Bandwidth

Coupling two T3E means that the actually used metacomputer is cut in two large pieces. The bandwidth between the two parts of the system will depend on the connection. In our case the ATM connections had either 2MBit/s or 10 MBit/s bandwidth. The corresponding bi-section bandwidth of a T3E-1024 is around 36 GByte/s (64 connections with 2 times 300MB/s each). This is a factor of about 30,000. Using a 644 MBit connection would still result in a factor of 450.

Library Viewpoint The library cannot increase the bandwidth, but can use it more effectively. Therefore PACX-MPI offers the possibility to compress data before sending it across the network connection. This is an advantage if the PACX-MPI node can compress more data per second than the bandwidth of the connection. At the moment PACX-MPI uses the `lzo` library which compresses about 3MB/s on an T3E-900. The compression rate for the binary data we transferred is typically 0.6. Thus if the bandwidth of the connection is below 24MBit/s one can expect higher throughput by using compression. In addition the provided buffer space is used more efficiently. Tab. 4.1 shows the resulting performance improvement.

	time for simulation	inner force
with compression	251s	46s
no compression	357s	46s

Table 4.1.: Performance results with and without compression of the transferred data.

Application Viewpoint As a first approach latency hiding also helps to hide the increased communication duration due to a small bandwidth. Increasing the number of particles does however not only increase computing time but also the amount of data to be sent. For a latency of 74ms and a bandwidth of 5MBit/s in both directions even with particles consisting of only 24 bytes data there is a break even between latency and communication duration above 15000 particles, a value easily reached in real applications.

The ratio between communication time T_{comm} and computation time T_{comp} is

$$\frac{T_{comm}}{T_{comp}} = \underbrace{\left(T_l + \frac{4r_c M_p n L^{d-1}}{B}\right)}_{T_{comm}} \underbrace{\frac{1}{\left(\frac{L}{P^{1/3}} - 4r_c\right)^d n T_{single}}}_{T_{comp}} \tag{4.1}$$

where T_l is latency of the connection, B bandwidth, P number of CPUs, M_p memory per particle in bytes, T_{single} computation time per particle, r_c cut off length for the forces, L is the system length, and n the particle density. This formula holds for a square cube with periodic boundary conditions.

The particles of the MD application consisted of a scalar radius and the three vectors position, velocity and force with double precision summing up to 80 bytes in three dimensions. According to formula 4.1 2.5 million particles per CPU are necessary to completely hide latency with a bandwidth of 10MBit/s and P=1024, $T_{single} = 100\mu$s, $r_c = 2$, and $n = 0.5$. One time step thus would take 250s which would limit the number of possible time steps to almost useless values. In the following, some possible optimizations are described to reduce the amount of data that has to be transferred.

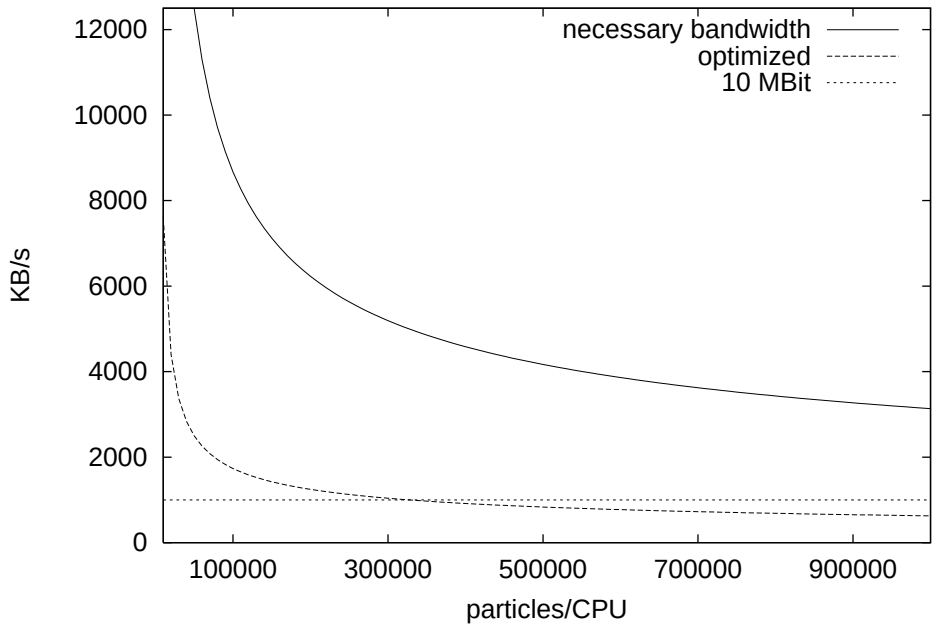


Figure 4.6.: Bandwidth demand of a three dimensional MD application for different number of particles per CPU according to Eq. 4.1. With P=1024, $T_{single} = 100\mu$s, $r_c = 2$, $n = 0.5$. The optimization techniques described significantly reduce the necessary.

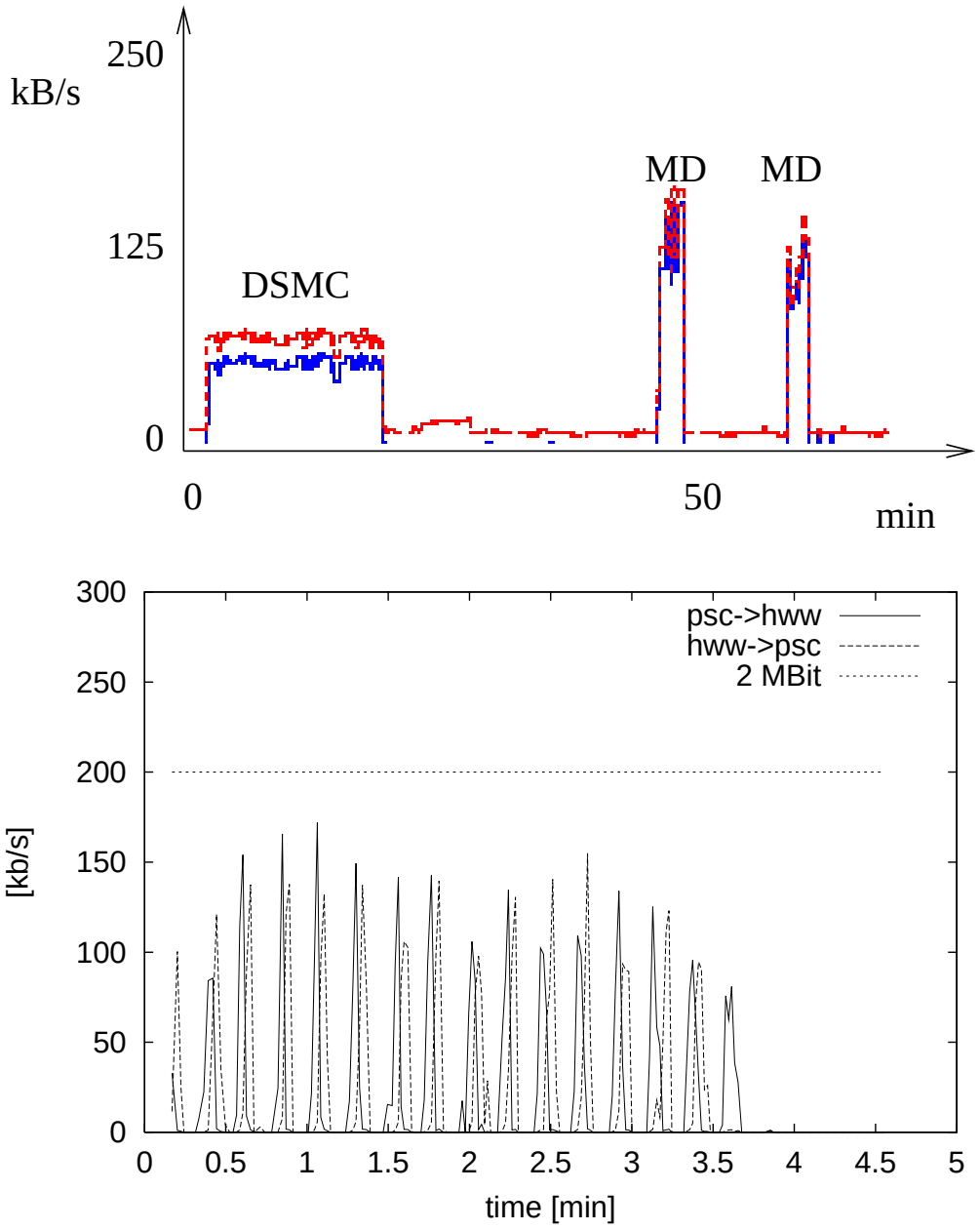


Figure 4.7.: Bandwidth measurement of different applications. On the top you can see the traffic off a DSMC and two MD runs. The sampling averaged over time periods larger than a time step. A saturated line would therefore indicate an application that waits for the communication. The measurement on the bottom shows an MD run with higher resolution. The high resolution run was with a 2MBit line. On this time scale the line gets saturated during communication.

Using floats Maybe the most obvious possibility to reduce the amount of communicated data is to use single instead of double precision. If the particles in the MD application consist only of floating point information the amount of data is reduced by 50 %. You can regard it as a form of non lossless compression in contrast to the lossless compression PACX-MPI provides. In contrast to the library, the application has the knowledge where loss is allowed. This approach however changes the numerics of the program and may not be feasible in all cases.

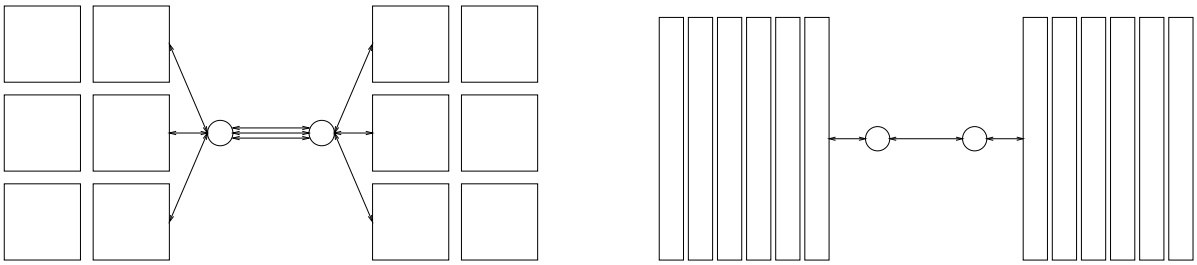


Figure 4.8.: Different domain decompositions and their effect on metacomputing. Circles denote the dedicated CPUs where PACX-MPI collects the messages before sending them across the link. For the 4x3 decomposition 3 packages have to be send across the link.

Using different domain decompositions Fig. 4.8 shows two possible domain decompositions for the same domain. While the amount of transferred data is almost unchanged it is send in a few big chunks instead of many small. Thus it helps PACX-MPI to use the TCP connection effectively. For the application however this means a less ideal domain decomposition, because more particles are in the shadow region now. Tab. 4.2 shows the execution time for different domain decompositions. The resulting additional work over-compensates the gain in this case.

decomposition	time for simulation	inner force
8x16	231s	47s
16x8	279s	50s
32x4	265	48s
128x1	308s	32s

Table 4.2.: Performance for different number of connections across the link. 8x16 is the best domain decomposition for the application. The 128x1 decomposition is the optimum case for the library because it has to handle only one large packet.

Node distribution The distribution of nodes between the participating computers is one of the most crucial points in metacomputing. A wrong assignment can spoil most of the performance. When running on two computers with one providing n_1 CPUs and the second n_2 PACX-MPI guarantees that the lowest n_1 ranks are on the first computer and the highest n_2 ranks on the second. The Cartesian domain decomposition is performed in a way to get the least communication between the two computers.

Distribution	time for simulation	inner force
256+240	251s	46s
248+248	353s	46s

Table 4.3.: Simulation time for a MD application with different node distributions.

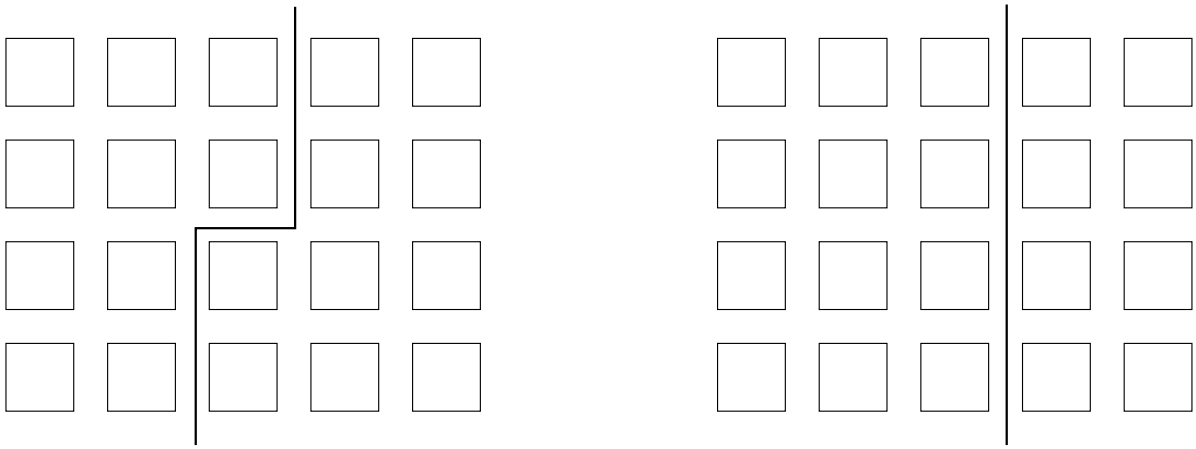


Figure 4.9.: Different node distribution for a Cartesian domain decomposition.

Table 4.3 and Fig. 4.9 show the impact of using a wrong node distribution. The distribution shown on the left side of Fig. 4.9 does not increase the traffic much, but prevents latency hiding because also the second stage of the communication is performed across the link.

Only send the core particle As described in section 4.1.2 the particles store the forces acting on them. Forces between real and shadow particles are also calculated. Therefore, the force does not need to be sent to a different CPU and can be set to zero when a particle is received. This reduces the amount of data from 7 to 5 floating point variables in 2D and from 10 to 7 in 3D.

Distinguish between real and shadow particles There are two reasons why a particle has to be send to a neighboring process: (i) it actually moved from one domain to another, (ii) it resides in a shadow region and is needed for the force calculation of a

different domain. In the latter case, it depends on the kind of force law which information of the particles is actually needed. In our case, we could drop the velocity of the shadow particles. This reduces the data for a shadow particle from 5 to 3 floating point numbers in 2D and from 7 to 4 in 3D. The total gain will depend on the simulation performed because the ratio between shadow and real particles depends on the average velocity of the particles. In general the number of shadow particles will dominate, thus the reduction is close to 0.6.

Simulate a different problem Equation 4.1 holds for cubic domains with periodic boundaries. The most effective reduction of bandwidth needs can be seen in applications like crack propagation, where the sample is elongated mainly in one dimension and it is non-periodic. A non-periodic system of size $L_x \times L_y \times L_z$ where $L_z > L_x = L_y$ needs only the bandwidth of a $2 \times \frac{L_z}{L_x}$ times smaller system.

4.1.5. Conclusion

Although a metacomputing library can provide a transparent usage of a metacomputer the differences concerning the performance of the connections between processing elements are clearly visible to the application.

Two important measures for this performance are latency and bandwidth. It turned out that even the huge latencies of transatlantic metacomputing in the range of 80ms can be handled by applications that can perform latency hiding. Bandwidth is a more restricting factor for applications like molecular dynamics. Several optimizations can improve the situation. On the library side compression of the transferred data is one possibility. However, additional improvements on the application side are necessary in order to make metacomputing efficient. The improvements presented in this work reduce the required bandwidth to 20% of the original value. This resulted in the qualitative change that the application can run as fast as on one huge supercomputer. It is important to note, that in contrast to the measures the library can perform, the application improvements are also valuable for high bandwidth connections – a field where improvements can be expected with the upcoming Gigabit networks.

4.2. Implementation

In chapter 2.3 the basic HSMC algorithm was described. This chapter contains general design concepts and some important details of the implementation. Without an implementation an algorithm is only a theoretical construct. An implementation is not only necessary to actually apply the algorithm to real world problems, but also to verify the correctness and completeness of the algorithm in an experimental way.

To achieve the maximum performance a parallel program was inevitable. The parallelization was also the most complex part of the implementation. Section 4.2.2 describes how this was done.

Another important decision during the implementation was the programming language. The availability of suitable compilers on the target platforms like the Cray T3E or other supercomputers limited the choice to the languages Fortran, C and C++. The final decision was to use C++. Besides of being a "better C" it offers some techniques of generic programming that are so far not available in Fortran. It will also be demonstrated how this resulted in a code re-usability that would have not been possible to this extend without these features. Last but not least it was a question of personal taste and familiarity with the language. Throughout this chapter code fragments will be given in C++.

Finally, performance is of utmost importance in the scientific community, where the calculations last thousands of CPU hours. It is therefore discussed in section 4.2.3 together with a performance comparison between C and C++.

4.2.1. Basic algorithm

The HSMC-algorithm integrates the time evolution of the system in discrete time steps dt. Every time step is split into the advection and collision step. Therefore the main loop of the program looks like this:

```
while(t<t_max){
   // advection step:
   do_advection(system,dt);

   // collision between particles:
   do_collisions(system,dt);

   // advance time
   t += dt;
}
```

For an efficient implementation the particles have to be stored in an appropriate way and at the same time access to all informations relevant for the algorithm should be provided. The chosen solution was to store all particles in a *container*, inspired by the container-concept of the `Standard Template Library` [95, 57] of C++. In addition to simple storage, the container also provides the partitioning into cells that is required by the DSMC and HSMC Algorithm.

Advection step In the case of DSMC the advection step was just the integration of the equation of motion. In general an analytical solution can be used, because the movement of one particle takes place completely independently from other particles. In the case of a constant gravitation $\mathbf{g}$ the new position $\mathbf{x}\_new$ and velocity $\mathbf{v}\_new$ at the time $t+dt$ is given by:

```
x_new = x_old + dt*v + 0.5*g*dt*dt
v_new = v_old+g*dt
```

This piece of code demonstrates nicely how in C++ the algorithm can be written in a natural and dimension independent way.

In HSMC the advection step is more complicated, because the "excluded volume" has to be considered.

The system is coarse grained to the level of the cells In the course of coarse graining the cells of the collision step are used as a space discretization. Every cell corresponds to a grid point of this discretization. The properties of each cell are determined by the particles in that cell. The mass is simply the total mass of the particles (Eq. (2.20)). Its velocity is calculated with Eq. (2.22) such that its momentum is the total momentum of the particles. Besides mass and velocity field additional fields store the volume that is available in the cell for incoming particles and the energy lost in the interaction between cells (see below).

The interaction between the cells is calculated As described in Sec. 2.3 the cells are treated like particles here. To allow a parallel update, the post collision velocities of the cells are stored in an auxiliary field. For the next step the energy loss of this step is stored in a second auxiliary field.

The energy dissipated in step two is redistributed as granular temperature The velocity of the cell is the velocity of the center of mass. The relative velocity to the center of mass is now rescaled according to Eq. (2.25).

Movement of the particles The particles are moved if they stay in the same cell or the new cell has enough unoccupied space. It should be noted that the available space is only calculated at the beginning of the advection step and not updated if a particle enters a cell. This allows for a parallel update of the particles position. Otherwise the result of the algorithm would depend on the sequence in which the particles are updated.

Collision step The implementation of the collision step is quite straightforward. After the container has sorted the particles into the cells that decompose the domain, all informations are available to calculate the number of collision pairs according to Eq. (2.34) and the procedure described in Sec. 2.3.

4.2.2. Parallelization

From the very beginning, a program suited for massive parallel computers was intended. The only suitable programming model that scales well to thousands of processors is message passing. However, the use of message passing and the corresponding MPI standard has its drawbacks. The message passing model differs significantly from the ordinary serial approach. Parallelizing the program after it has been implemented in a serial version has the disadvantage that the two versions soon become decoupled. Every time the serial version is changed this has to be incorporated in the parallel program. On the other hand you also don't want to only maintain the parallel implementation.

First, it is a major restriction if the program runs only on parallel platforms. Second, it takes in general more effort to modify the program in order to test new algorithms. The solutions to this dilemma is to hide the parallel implementation at a suitable level of abstraction.

First of all the most appropriate parallel programming model for the selected problem has to be chosen.

Task parallel Several tasks are distributed across the processing elements. They can act either on different data or the same. In the later case the tasks consist of different stages in a kind of pipelining process.

Data parallel Within this model the different processing elements perform the same operations for different data sets. Several approaches to select these data sets further distinguish the data parallel model.

Particle decomposition Every PE takes care of a fixed number of particles. The set of particles does not change during a simulation. Due to the particle interaction the required information has to be communicated between the PEs.

Domain decomposition To every PE a part of the system or domain is assigned. Every PE calculates the motion of the particles in its domain. For the interaction of the particles so called overlapping regions are established. Particles that are within cut-off range to a neighboring domain are not only stored a one PE, but also on the neighbor PE. The number of particles of each PE varies because the particles move from domain to domain. The consequence is a certain amount of load imbalance. As long as the system is homogeneous on the scale of PE domains this can be neglected.

For granular systems with short range interaction the domain decomposition offers the best solution. For most of the systems load imbalance was no problem.

One decision that has to be taken at different points throughout the implementation is *Replication* vs. *Communication*. E.g. when a force is calculated between two particles that belong to different PEs there are two possibilities. First, the force can be calculated on one PE and communicated to the second. Second, the force calculation is performed or *replicated* on both PEs. The approach taken in this implementation was to prefer *Replication*. The idea behind this decision is that communication is more expensive than calculation. While this is not necessarily true on platforms with extremely fast communication networks like the Cray T3E, it clearly is the preferred solution for all other situations, like e.g. in Metacomputing.

Speed up

One of the standard tests designed to measure the overhead of a parallel algorithm or implementation is the speed-up test. When T_s is the time to solve a given problem on

a single PE and $T(n)$ the time to solve the same problem on n PEs, the speed-up s is defined as

$$s = \frac{T_s}{T(n)}. \tag{4.2}$$

In this benchmark the same workload is distributed to more and more PEs. It is therefore a sensitive test for the overhead introduced by the parallel algorithm. Fig. 4.10 shows the result of a DSMC simulation compared to the speed-up of a molecular dynamic program. DSMC shows a much better speed-up due to its reduced communication overhead [46].

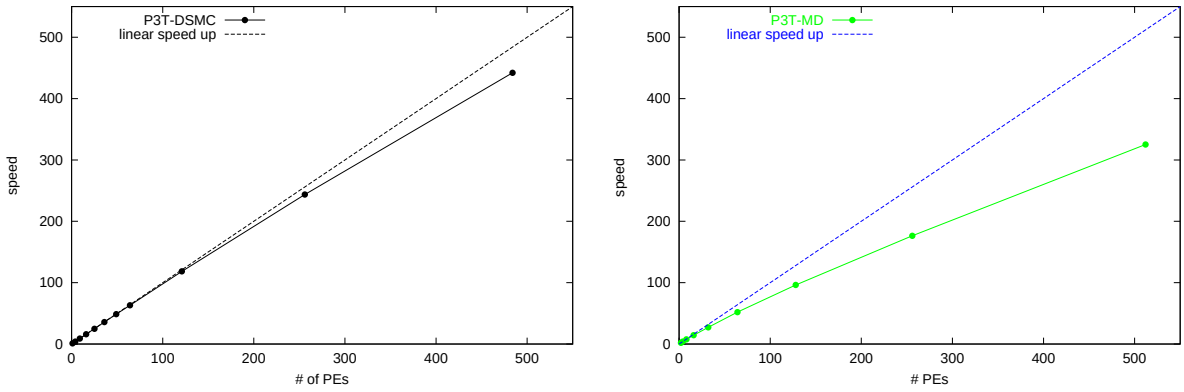


Figure 4.10.: Comparison of the speed-up of a DSMC program with a MD program.

4.2.3. Scientific Computing and C++

Advantages of Using C++

Instead of citing the standard advantages of C++, the focus of this section is on the experiences made in this work. At the beginning of this work it was not at all clear whether the use of C++ has more advantages than disadvantages. Now it is clear that C++ was the right choice. The reasons are manifold. First of all the language was designed to write libraries. With several colleagues it was thus possible to create a collection of useful algorithms and tools for the problems we were interested in [51]. This is of course possible in any language. However, what is enabled by good programming style and convention in languages like C or Fortran is enforced or standardized in C++. The class as the central concept forces the programmer to separate interface and implementation on one hand and on the other hand enables the user to employ the class without having to worry about the implementation and unforeseeable side-effects. E.g. the random number generators (RNG) used in HSMC are part of a class hierarchy, that despite the different internal algorithms to generate the random numbers all offer the same interface. In this way there are completely interchangeable which can be used to verify that the results are independent of the specific RNG used. Another advantage of C++ is the possible code

reuse due to the generic programming enabled by the template features. One example is the *container* that was developed to store the particles. The cells that are used within the DSMC or HSMC-Algorithm can easily be used in the linked cell algorithm of short range molecular dynamics. The *container* already takes care of the domain decomposition on distributed memory parallel computers. It is thus straightforward to write a parallel MD program [86]. The parallel particle container that was originally developed for HSMC was successfully used in several other projects [49, 50, 123]. Combining the parallel particle container with a parallel fluid algorithm, it was also possible to write a parallel program simulating particles in liquids [52, 124]. The coupling of this two codes was reduced to implement the interaction between the particles and the fluid. Since this interaction is local, this did not require any modification or even knowledge of the parallel code.

Performance

Comparison between C and C++ While recent benchmarks have shown that C++ can compete with C or Fortran [121] it is not clear whether this holds for a particular application. Since performance depends on the abstraction techniques used [37, 112] it is a non trivial task to find a suitable balance between certain techniques and performance. Unfortunately there was no comparable implementation of DSMC or HSMC available to me. Many of the techniques were also used in a MD program that I also wrote during my thesis. With this program a direct comparison with a similar program [116] written in C was possible.

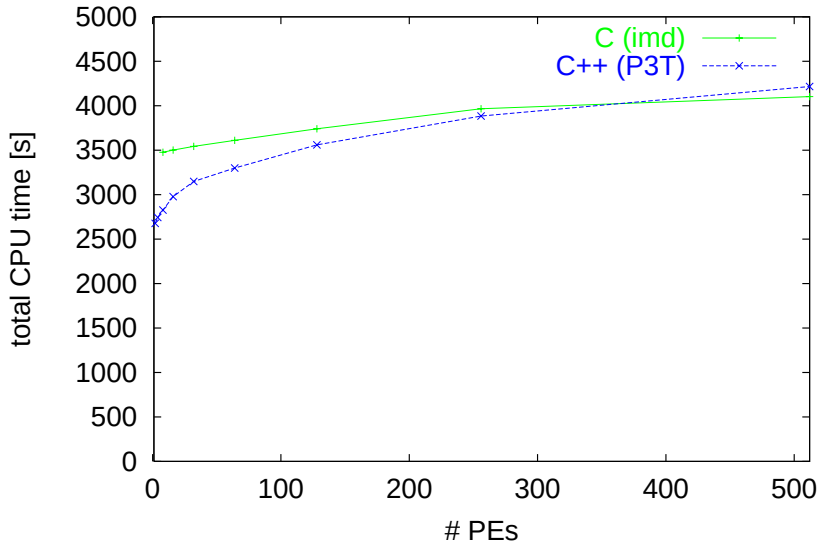


Figure 4.11.: Total runtime of two different Molecular Dynamics programs depending on the number of PEs.

The basic linked cell algorithm is the same. For a simulation of a Lennard-Jones fcc crystal with 442368 atoms the C++ version is between 3 percent slower and 20

percent faster. There are several reasons why the speed up of the C++ program is worse. First, latency hiding is applied and its advantage is lost rapidly with decreasing numbers of particles. Second, Newton's third law is not applied across PE boundaries. This increases the workload on a single PE and results in larger overall runtimes when the cost of the additional communication for exchanging forces is negligible. For large particle numbers the overlap of communication and computation will also overcome performance problems due to possible congestion of the T3Es network [91].

Performance of different compilers While C++ was designed to have only minimum runtime overhead, the actual abstraction penalty does not only depend on the abstraction level used, but also on the optimization technique of the compiler. Fig. 4.12 gives an impression of the differences we observed between different compilers. There is a factor of more than three between the best and worst optimizing compiler.

But using the best compiler is not enough. We observed a similar performance penalty when upgrading from KCC 3.2 to KCC 3.3. The reason was that with KCC 3.3 exception handling was turned on by default. Using a compiler switch to disable exception handling, we were back to the old performance. This cannot be intrinsic to exception handling because `cxx` does show good performance while still providing exception handling. Obviously some important optimizations are affected. This problem is more serious on architectures that depend on high level optimizations to achieve a godd performance [87]. Examples are vector supercomputers or the upcoming processors with explicit parallel instructions set.

Compiler	flags
SUN CC 4.2	-fast
DEC cxx 6.x	-O5 -tune host -ieee -assume noaccuracy_sensitive
IBM xlC 3.1.4	-O3
Cray CC 3.0.1.3	-O3
KAI KCC 3.2	+K3 -O3
GNU gcc 2.7.2	-O3

Table 4.4.: Optimization flags used for the different compilers.

Abstraction penalty and exception handling Exception handling (EH) is one feature of C++ that simplifies the aspects of programs dealing with error handling, especially regarding runtime errors that can occur during file I/O and similar situations. Many compilers offer the possibility to disable the support for exception handling. In the application section of this paper we have already seen that, even if exceptions are not used in the program itself, enabling EH support can inflict severe performance penalties. To see whether this holds for general applications we tried to reproduce this effect with the well known Stepanov Benchmarks. The Stepanov Benchmark is a collection of 13 kernels written in C++, each performing the same computation. Kernel zero is written at a low level, using no C++ data abstraction features. The other kernels use various

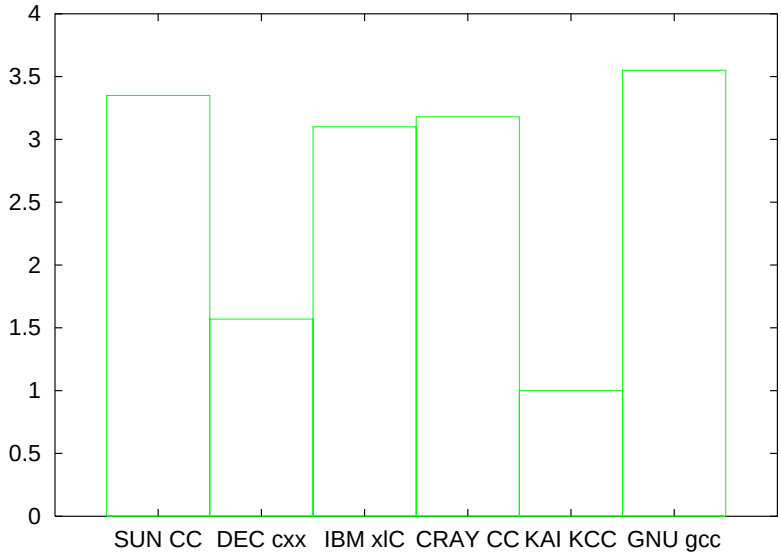


Figure 4.12.: Runtime of a MD program with different compilers relative to runtime using KAI KCC. The runtime with KAI KCC 3.2x is set to one, because it is available on all platform. The value for GNU gcc is the average from two platforms (Digital Unix (3.4) and SUN Solaris (3.7)).

C++ data abstraction features, in various combinations. Tab. 4.5 shows the results for the native CC compiler on the T3E and Origin (Cray C++ 3.2.0.1 and MIPSpro 7.2.1.2m) and a third party compiler from Kuck & Associates on the T3E (KCC 3.3d). Optimization was done with -O3 or +K3 -O3 in the case of KCC.

In the case of the native T3E compiler EH does not influence performance. The mean value of the abstraction penalty, 3.29 is however by a factor 2.5 slower than the best in this comparison. Kernel 1 reveals that the inlining of the T3E compiler is poor. The best optimizing compiler (KCC) is very sensitive to the influence of EH, some kernels are almost two times faster without EH. Like the native compiler on the T3E, the CC on the Origin shows no significant differences, but also offers less overall optimization.

Conclusions

I have shown that object-oriented concepts help to design flexible, portable and easy-to-use tools for important problem classes found frequently in science and engineering. One example is the parallel template particle container. First it was developed for the use in DSMC and HSMC. Since it was designed in a generic manner with the help of the C++ template mechanisms it was also applicable to fields like molecular dynamics and the simulation of particles in fluids.

Via comparison between a molecular dynamics program that made use of the parallel template particle container and a classical C program it was demonstrated that C++ does not necessarily show performance degradations compared to C.

	T3E CC		KCC		Origin CC	
	exceptions		exceptions		exceptions	
	no	yes	no	yes	no	yes
0	1.00	1.00	1.00	1.00	1.00	1.00
1	2.68	2.68	0.70	0.70	1.00	1.00
2	3.31	3.31	0.70	1.24	1.04	1.15
3	1.89	1.89	0.73	0.73	1.93	1.93
4	3.03	3.03	0.70	1.27	1.89	1.93
5	2.81	2.81	0.70	0.73	1.93	1.89
6	3.15	3.15	0.74	1.24	1.89	1.93
7	3.78	3.78	2.42	1.75	1.93	1.93
8	4.72	4.72	1.75	2.89	1.96	1.93
9	3.37	3.37	2.35	2.35	1.93	1.89
10	4.72	4.72	2.45	2.75	1.89	1.93
11	5.87	5.87	2.48	3.15	1.93	1.93
12	7.03	7.03	3.15	3.82	1.89	1.96
mean	3.29	3.29	1.28	1.55	1.65	1.67

Table 4.5.: Abstraction Penalty on CRAY/SGI T3E and SGI Origin (see text).

The Stepanov benchmarks show that an optimal performance is the result of a careful balance between necessary abstraction and desired performance.

5. Summary and Conclusion

The goal of this work was the design, implementation and analysis of a fast simulation method for granular materials. The starting point was the Direct Simulation Monte Carlo Method. Not because it was the best suitable method for granular materials, but because it is very fast. The short overview in chapter 2.5 also shows that the methods applied so far to simulate granular matter have their specific advantages and disadvantages. No single method can therefore considered to be the best.

Since DSMC was designed for the field of dilute gas flow, it has of course several limitations for dense flows. For dense systems several improvements like CBA, CUBA or ESMC exist (see chapter 2.4). The most serious disadvantage of DSMC for dissipative systems is the missing "excluded volume". In contrast to all other DSMC variations, I did not improve the collision step, but the advection step. Chap. 2.3 contains the arguments why it is not enough to change the advection step in the case of a dissipative system. In order to be more efficient than molecular dynamics, which naturally incorporates the excluded volume, the excluded volume is only enforced on a coarse grained level. Although there have been large modifications to the original DSMC algorithms, HSMC is identical to DSMC in the limit of dilute systems.

In order to test and verify the algorithm several test cases have been selected. Every test case covers specific properties of granular materials. As a whole they cover a wide spectrum of phenomena.

Homogeneous cooling (chap. 3.1) was simulated for comparison with the kinetic gas theory. HSMC reproduces the theoretical results for area fractions of up to 0.9. This demonstrates that the dissipation is still modeled correctly, even if a large number of moves of the particles is rejected and a more complex dissipation mechanism via "cell collisions" is used instead.

In the case of the clustering instability the results of HSMC have been compared with DSMC and ED. Since the local density increases with time, the suitability of HSMC for dense systems translates to longer time scales that are accessible with this method. As soon as the system is dominated by clusters, most of the energy is stored in slowly moving clusters with low granular temperature. The correct simulation of the energy thus also reflects the momentum conservation that has to be fulfilled. The dissipation in this system leads to a variation of energy across several orders of magnitude. Since there are no intrinsic time scales built into HSMC this is no major problem and only limited by the numerical representation of real numbers.

In a system that is sheared different velocity scales are present at the same time. In addition to the correct scaling of the viscosity with the shear rate as described by

Bagnolds law, two results should be noted. First, it is important that the underlying fixed cell structure of HSMC does not impose a fixed reference frame onto the system. Second, for efficiency reasons a novel approach to adjust the time step had to be developed, otherwise the fast movement of particles close to the walls would limit the time step to unreasonably small values.

Spontaneous formation of density waves in pipe flow is another property that is reproduced by HSMC. Experiments and theory agree that the surrounding fluid is an important factor for the density fluctuations. On the other hand LGA does not contain the fluid and shows a power law that is consistent with the experimental results. One major difference between LGA and HSMC is that the particle velocity is limited in LGA. This is similar to the limitation of the particle velocity due to a drag force. Since HSMC contains no such limitation, the validation of the correct power law is another, strong argument that the surrounding fluid is not required.

The simulation of heaps is the last test case. It was already demonstrated by ED simulations that static friction is not necessary to build realistic heaps, but steric effects are sufficient. In the same way HSMC can simulate heaps. HSMC reproduces not only the constant angle of repose, but also the small logarithmic deviations at the tail of the heap. Whether the heap is flat or steep at the tail depends on the boundary condition. It is flater than the angle of repose if the heap is on an infinite support and steeper if limited by an edge of a finite plate – an effect also visible in HSMC simulations. Quantitative comparisons are limited in these cases because the shape of particles has an influence on the macroscopic properties, and this information is not contained in HSMC. The missing steric effects on the particle scale are also the reasons for artifacts on the head of the heap, where the angle of repose is too steep, if the impact velocity of the particles is small.

The next chapter took a closer look at the limitations of HSMC by checking the correlations that exist in dissipative systems between different particles. The results show that higher order correlations between particles are the main reason for deviations between HSMC and ED simulations. It is also argued why these correlations cannot be introduced without sacrificing the efficiency of the method. The section about vibrated beds is an additional case study where the results of the stochastic DSMC are compared with results from hard sphere molecular dynamics. This shows that correlations between particles in a narrow two dimensional systems are not important for the correct scaling behavior.

Another important part besides the development and verification of the HSMC-algorithm was the design and implementation in C++. An object oriented approach that hides the difficulties of parallel programming and allows to reuse the software in other contexts without significant loss of performance was the goal. The suitability of the program for parallel computing was not only shown on MPPs, but also on coupled supercomputers, the so called metacomputing. Here perfect scale-up could be achieved by latency hiding. The reuse ability of the software was demonstrated by using the developed parallel template particle container for fields like particle in fluids and molecular dynamics. In the second field a comparison of a Molecular Dynamics program with a program written in C showed that the applied abstraction techniques did not reduce the

performance in comparison with classical procedural programming.

Altogether the development of HSMC resulted in an algorithm that reduces the limitation of DSMC, which served as a starting point. The resulting suitability to simulate granular matter was demonstrated in several different flows like homogeneous cooling, clustering, pipe and shear flow and heaps. These flows covered different aspects of granular matter and posed specific difficulties for the simulation method. The performance of the parallel program was tested on MPPs and in metacomputing testbeds. The comparison with a C program was not only another performance test but showed also the reuse ability of the software. Further improvements of the methods are possible to reduce the lattice artifacts, that occur in systems with high density and strong dissipation. Some possibilities were additional knowledge can be incorporated into HSMC have been discussed in chapter 2.3 and 3.5. The first example is the probability distribution function of the impact parameter $p(b)$, where the examination in chapter 3.6 have shown that a modified $p(b)$ can result in substantial improvements. A second example is the probability $p(\nu)$ of a particle to enter a cell with volume fraction ν, where so far a heavy-side function has been used (see chapter 2.3). A further possibility is to modify the lattice itself. Like in LGA simulations the appropriate orientation of the lattice relative to gravity or other preferred directions of the flow will reduce the lattice artifacts. The additional advantage of HSMC is, that the lattice is only used to enforce the excluded volume on a coarse grained level, i.e. the movement of the particles is not restricted to vertices of the lattice. It is therefore possible to rotate or shift [55] the lattice between time steps. This would *in the average* result in an isotropy and homogeneity of space, as long as there is no preferred position or orientation of the lattice.

Due to the high performance of the method it has huge advantages for systems where large particle numbers or simulation times are necessary, e.g. the three dimensional simulation of clustering. It can also be applied to examine whether correlations between particles or steric effects on the particle level are necessary to reproduce certain effects. Although DSMC can also be used for this, the limitation of DSMC to dilute flows is a large handicap for dissipative systems. Examples where the HSMC results can guide theoreticians are the scaling behavior of vibrated beds, the shape of a heap or density fluctuations in pipe flow.

A. Optimizations and Benchmarking

A.1. Optimizations for Cartesian Domain Decomposition

Domain Decomposition is a wide spread method to distribute the workload among a group of processors. In the case of a rectangular or cubic domain where every processor area has the same size it results in a Cartesian topology. The performance of exchanging data between neighbors in this topology is crucial for applications that apply this methology.

Benchmarks are widely used to measure performance. On one hand they should be specific enough to serve as a guideline for improvements, on the other hand they should reflect the demand of real applications. Most MPI benchmarks concentrate on point to point and global communications [108]. Some benchmarks also use kernels of applications or even complete applications [109, 48]. These benchmark results contain information about the performance of different communication patterns or parallel algorithms. This information gives guidance in the development of a high performance implementation of a parallel program. However, in the course of profiling and improving the parallel implementations of DSMC, HSMC and MD I encountered some problems that were not represented in standard benchmark results. Specifically this affected the performance of communication in a standard Cartesian topology. In the following the general problem will be described and an optimization technique will be provided. Performance results on a Cray T3E that demonstrate the effectiveness of this technique are also included.

A.1.1. PE mapping and MPI

MPI identifies a PE with its *rank*. In the first place there is no information about a topology associated with those ranks. On the T3E the PEs are numbered in a way that ranks with a small difference are likely to be close in the communication network. It is clear that it is impossible to provide a good solution for all situations.

MPI therefore offers to create special process topologies. One possibility is to create an n-dimensional Cartesian grid with a call to `MPI_Cart_create`. One option allows the ranks to be `reordered`, this gives the implementation the possibility to remap the PEs to get the best performance with the specific hardware. The PEs of the T3E are physically organized in a bidirectional three dimensional torus. This is well suited for the Cartesian grids found in applications with domain decomposition.

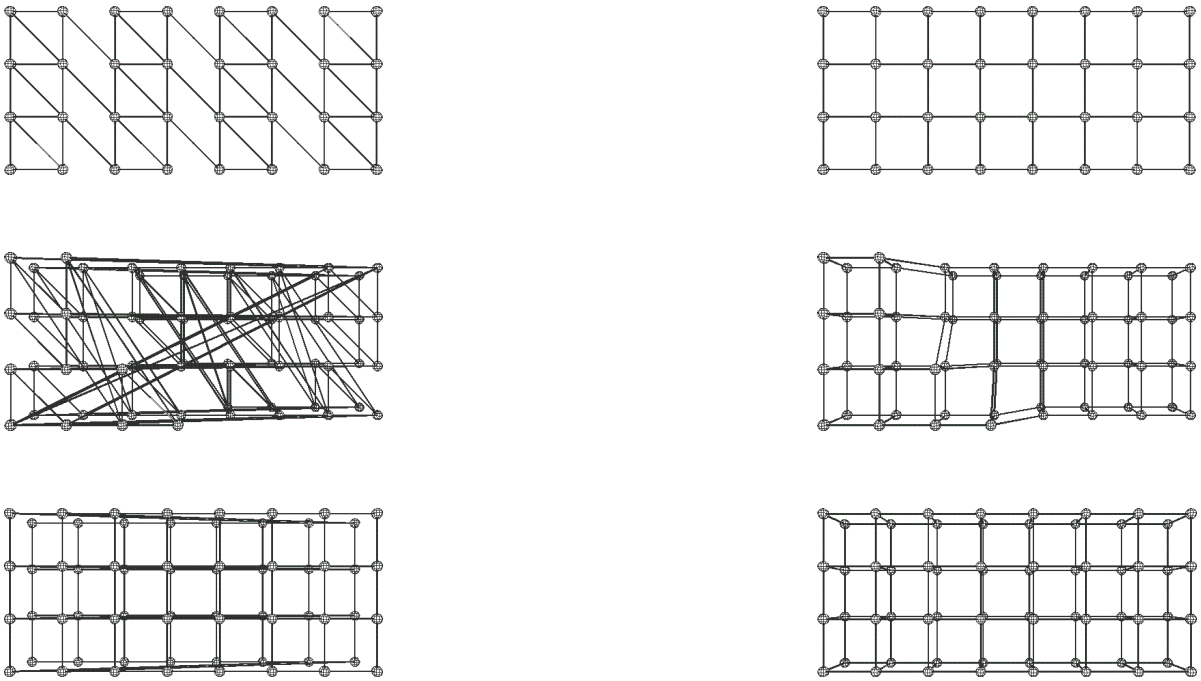


Figure A.1.: Different mappings of Cartesian coordinates onto PEs. Left: generated with `MPI_Cart_create`. Right: generated with `MPI_Cart_create` that performs reordering.

However, the MPI on the Cray does not perform any reordering of PEs. An optimized reordering of PEs can be based on the information about the physical layout if the PEs which is available from the *sysconf* call. The algorithm chosen for optimization is to sort the physical coordinates according to x,y and z-coordinates. There are 6 permutations of the (x,y,z) triplet. Each of them is tested with respect to average hop count and the optimum ordering is chosen. Results for different Cartesian communicators are shown in Fig. A.1.

The result of the optimization measured in reduction of average hop counts is given in Tab. A.1 for varying dimensions of the Cartesian communicator.

Grid	without reordering	with reordering
2x2x2	1.3	1.3
3x3x3	2.5	1.8
4x4x1	2	1
4x4x4	2.8	1.8
8x4x1	2.2	1
8x4x2	1.5	1

Table A.1.: Average number of hops a message has to travel.

A.1.2. Performance Measurements

Pair communication

For contention to occur several PE have to communicate simultaneously. A communication pattern where the PEs communicate pairwise with each other (see Fig. A.2) was chosen.

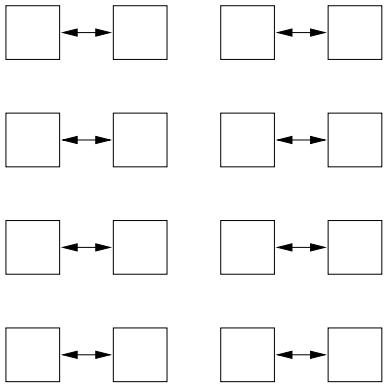


Figure A.2.: Topology of the benchmark problem

It is clear that this does not reflect any real application but the design goal was to have completely independent communications. The messages are sent with `MPI_Isend`

and received with `MPI_Irecv` (see Fig. A.3). Since the hardware of the T3E is capable of bidirectional communication the messages could be sent and received at the same time. A different possible optimization would have been to use `MPI_Sendrecv`, but the use of the immediate send and receive reflects the use in real application where some computations are done between the send/receive calls and the `MPI_Waitany`.

The topology is created with a call to `MPI_Cart_create` and `MPI_Cart_shift`. This gives the underlying MPI implementation the possibility to perform an optimized mapping onto the hardware.

```
MPI_Isend(sendfield,size,MPI_DOUBLE,partner,tag,comm,&request[0]);
MPI_Irecv(recvfield,size,MPI_DOUBLE,partner,tag,comm,&request[1]);
MPI_Waitall(2,request,status);
```

Figure A.3.: Code segment of the benchmark code.

The bandwidth is defined here as the amount of data that can be sent by one PE per second.

From the observed maximum bandwidth of about 200MB/s with 2 PEs (see Fig. A.4) we conclude that at least to some extent bidirectional communication is performed. One link of a T3E-900 has a bandwidth of about 300MB/s for MPI.

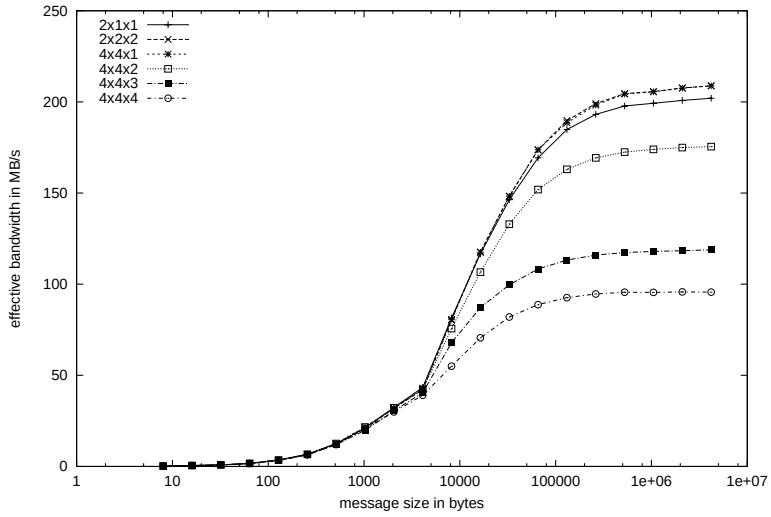


Figure A.4.: Bandwidth depending of message size and number of PEs. The topology is created with MPI_Cart_create.

The contention is obvious in Fig. A.4. The bandwidth for message sizes above 10.000 bytes drops with increasing number of PEs. For 64 PEs the bandwidth for large messages drops to less than half of the uncongested value.

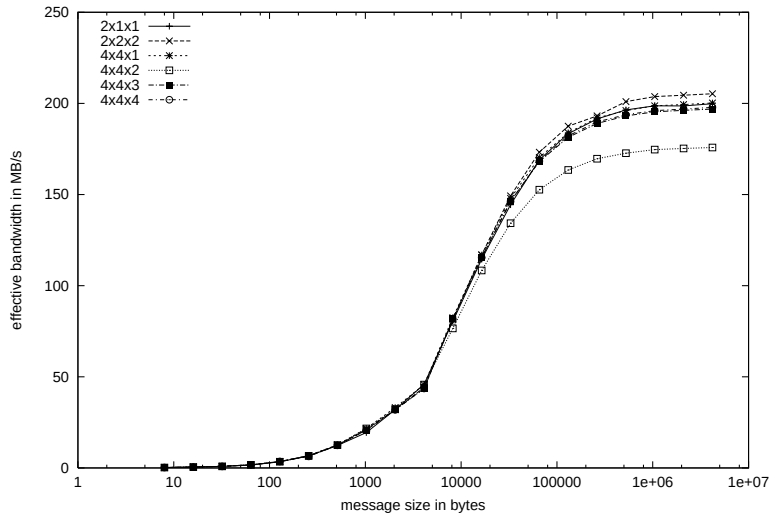


Figure A.5.: Bandwidth depending of message size and number of PEs. The topology is created with an optimized mapping.

Using the optimized mapping the congestion was reduced dramatically. Fig. A.5 shows that nearly for all cases the optimized mapping results in higher performance.

Extension to cyclic communication:

In order to reduce finite size effects, many simulation apply periodic boundary conditions. The periodicity is also reflected in the Cartesian topology of the processors. The communication network of the Cray T3E is a three dimensional torus and thus equivalent to a three dimensional periodic Cartesian grid.

To find out about the behavior of the toroidal communication network the example was extended. Now communication is done between all neighboring nodes in the x-direction. Each process has to talk to a left and a right neighbor. In principal all communication can be done within one single communication step. This should also hold for cyclic communication patterns assuming that all processes are equal with respect to network topology.

For the test example a 8x4x1 Cartesian communicator was created. First the influence of periodicity was measured. The tests with a periodic communicator include cyclic communication. For the non-periodic communicator no cyclic communication is done. A first guess would be that the additional communication should cost more. This would be true on machines with a standard network. However, the toroidal mesh of the T3E should show no difference. All tests were done using a fixed message size of 4MB. All measurements were done 100 times and the accumulated times are given. Without setting MPI_BUFFER_MAX one communication should take about 2.7 seconds.

First results showed a significant difference which was independent of the ordering method chosen. The overhead for cyclic communication was significant for both kinds

of communicators as can be seen in Tab. A.2.

	without reordering	with reordering
non-periodic	8.2 sec	5.9 sec
periodic	8.7 sec	6.8 sec

Table A.2.: Time for two sided exchange communication.

The striking point here is that even for a periodic communicator we would expect that all communication could be done within two communication cycles. This should take 5.4 seconds. For the Cray T3E it takes more than 3 communication cycles. Even for the optimized method there is some overhead.

To find out what could be done about that problem more communication patterns have been measured. The result was that once cyclic communication is involved the performance of the network decreases. Further investigation made us think that we should optimize our communication pattern to make it easier for the system to cope with a fully loaded network. So we explicitly decoupled the communication and did it in two steps. For this, every second processor row started with the communication to the right neighbor. Every other second row did the same for the left one. In the next step all processors switched to the other neighbor. This way we thought that within two communication steps all communication should be finished. The results are given in Tab. A.3.

	without reordering	with reordering
standard	8.7 sec	6.8 sec
new	9.2 sec	5.4 sec

Table A.3.: Time for two sided exchange communication using a two step communication scheme.

Although the decoupling of communication should make the Cray MPI version faster it actually slows it down even more. We suspect that by prescribing the communication pattern we loose the possibility to optimize the communication for the bad mapping of processes. For our own mapping strategy we find that we actually can achieve peak bandwidth for that communication pattern.

In this chapter we have seen that completely independent messages can interfere with each other: the communication gets slower because the network of the parallel computer gets congested. In the next section I will discuss the communication pattern found in the DSMC, MD and HSMC applications and show how the modified mapping can provide a solution to the contention problem.

Application benchmark results

The topology in this case is a three dimensional Cartesian grid. This corresponds to a standard domain decomposition that can be found in many applications. First the communication between the neighbors of one dimension is performed. After it is completed the communication for the next dimension is started. The topology is again created with a call to `MPI_Cart_create`.

Again, the bandwidth is defined as the amount of data one PE can send. Since there are two sends done simultaneously in different directions the theoretical peak bandwidth is 600MB/s for MPI.

The observed bandwidth is always below 200MB/s (see Fig. A.6). There seems to be just one send active at a time.

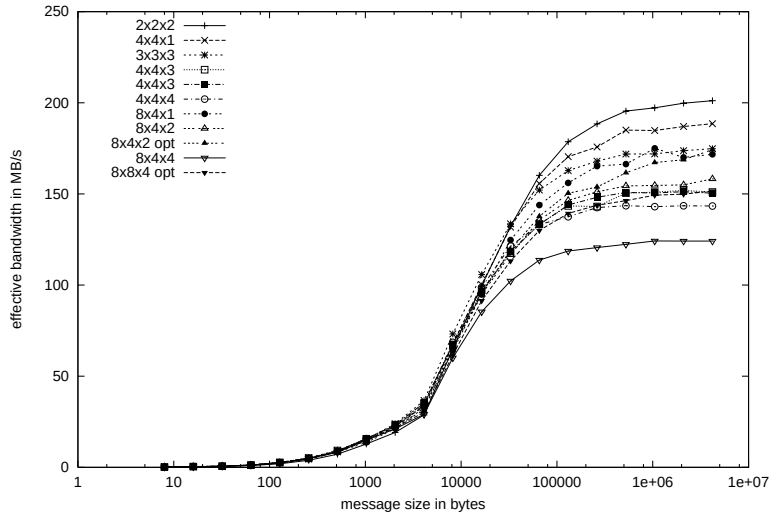


Figure A.6.: Bandwidth depending of message size without reordering of PEs.

Again congestion occurs. In this case the bandwidth decreased from 200MB/s to 124MB/s for 128 PEs (8x4x4). The decrease is less pronounced than in the benchmark case because the different directions are effected to different degrees. A poor performance of one direction is compensated by a good performance of another one. In principle there is no reason for a congestion, the layout of the T3E is a bidirectional three dimensional torus [114, 115]. Two neighbors in a Cartesian grid should have a direct connection.

However, the operating system does not know whether a request for 64 PEs results in a 4x4x4 or 8x8 grid. It guarantees just a connected area in physical PE space. This provides a first approach to local communication between PEs.

But even if the layout of the physical PEs corresponds to the requested grid contention occurs. `MPI_Cart_create` does not reorder the ranks to optimize the mapping.

Instead of calling the original `MPI_Cart_create` I used my own version that performs reordering as described above. The result is shown in Fig. A.7. The bandwidth drops only to 154MB/s. I checked the quality of a mapping by looking at the average number

of hops a message has to travel. The hop count of the grids created by the improved mapping was between 1 and 1.8 (see Table. A.1). Whenever the layout of the physical PEs corresponded to the requested grid the hop count was one. The performance is between 2 and 40 percent better than the unoptimized mapping for PE numbers larger than 8. Contention only occurs when the topology is strongly disturbed. In the case of 8x8x4 PEs the domain was practically separated into two partitions with 128 PEs. Small distortions are handled well as can be seen in Fig. A.1.

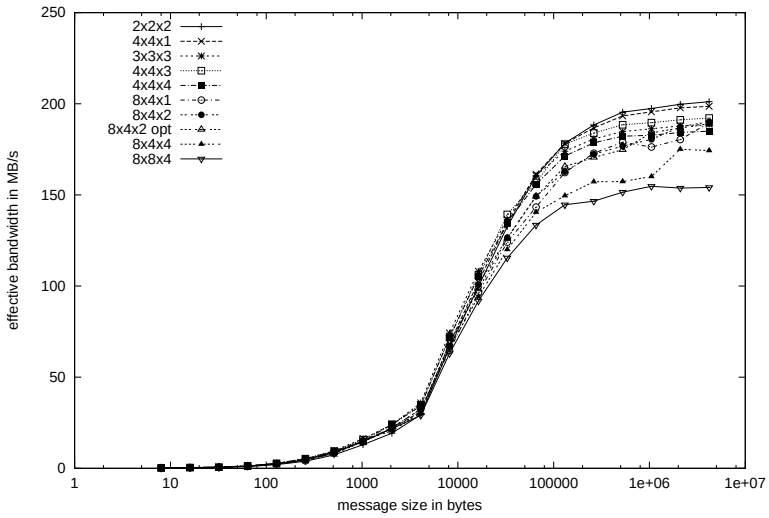


Figure A.7.: Bandwidth depending of message size with reordering of PEs.

I have shown that the mapping of communication topology onto the underlying hardware is important for benchmarks reflecting communication patterns found in real applications. This opens a wide field of optimization. The first finding was that the application partition of a T3E should be configured to be undisturbed by OS or CMD PEs in its area. This will not only provide better mapping for the standard programs but will also enable optimized mapping algorithms to find the optimal solution. Furthermore fragmentation does not only affect the jobs you start on a system, but also the performance of the already running jobs. An optimized batch system should take this into account.

Applications which suffer under contention and communicate in a Cartesian grid should use a version of `MPI_Cart_create` with reordering. The method presented here provides such an optimized mapping. It was easy to implement and the overhead of the optimization process is neglectable compared to simulation times of real applications.

A.2. Benchmarks

A.2.1. Memory usage

Measured on DEC alpha with g++:

Nr of particles	vsize	rsize
0	488k	488k
2	2776K	212K
100	2736K	212K
5000	3328K	786K
10000	3936K	1400K
100000	14M	12M

Memory usage: 2723k+N·0.12k vsize, 195k+N·0.12l rsize.

A.2.2. Computational speed

Computer	Compiler	Result
DEC alpha 255/233	g++	16662 Coll/s
SUN ultra1/170	g++	14121
IBM43P	g++	Compiler error
DEC alpha 255/233	cxx	11166 Coll/s
SUN ultra1/170	CC	10990
IBM43P	xlC	10600

List of symbols

symbol	description
A	area
α	angle of repose
b	impact parameter
BLGA	Boltzmann Lattice Gas Automaton
$\hat{C}$	collision operator
χ	Enskog factor
CD	Contact Dynamics
d	dimension
D	diameter
$\hat{D}$	advection operator
DSMC	Direct Simulation Monte Carlo
$\mathbf{e}$	unit vector
ϵ	inelasticity
ED	Event Driven
f	frequency
$f^{(N)}$	N particle distribution function
$\mathbf{F}$	Force
$\mathbf{F}_n$	normal Force
$\mathbf{F}_t$	tangential Force
Φ	flow
γ	flow or shear rate
G_{nn}	spatial density correlation
$G_{\alpha\beta}$	spatial velocity correlation
$G_{\parallel}$	parallel component of velocity correlation
$G_{\perp}$	perpendicular component of velocity correlation
h	height
H	Hamiltonian
HSMC	Hybrid Simulation Monte Carlo
k	wavenumber
$K(t)$	kinetic energy at time t
l	length
λ	mean free path
L	system length

symbol	description
LGA	Lattice Gas Automaton
n	particle density
N	total number of particles
ν	volume or area fraction
m	mass
M_c	total mass of particles inside a cell
MC	Monte Carlo
MD	Molecular Dynamics
μ	viscosity
μ_d	coefficient of dynamic friction
μ_s	coefficient of static friction
$\mathbf{p}$	momentum
p	wall roughness parameter
$p(b)$	probability distribution of impact parameter
$P(f)$	power spectrum
r	coefficient of restitution
R	Radius
ρ	density
$S(k)$	structure factor
σ	scattering cross section
τ	time step
Θ	Heavyside function
$\mathbf{v}$	velocity
V_c	Volume of a cell
$V_N(\mathbf{x}^N)$	potential Energy of a N-particle system
$\mathbf{x}$	position
Z	random number

List of Figures

List of Tables

Bibliography

[1] J.-L. Aider, N. Sommier, T. Raafat, and J.-P. Hulin. Experimental study of a granular flow in a vertical pipe. *Phys. Rev. E*, 59(1):778–786, Jan. 1999.

[2] F. Alexander, A. Garcia, and B. Alder. A consistent Boltzmann algorithm. *Physical Review Letters*, 74(26):5212–5215, 1995.

[3] F. Alexander, A. Garcia, and B. Alder. *Simulation of the consistent Boltzmann Equation for hard spheres and its extension to higher densities*, pages 82–90. Lecture notes in physics: 25 years of non-equilibrium statistical mechanics. Springer, 1995.

[4] F. J. Alexander, A. L. Garcia, and B. J. Alder. Direct simulation monte carlo for thin-film bearings. *Physics of Fluids*, 6(12):3854–3860, 1994.

[5] F. J. Alexander, A. L. Garcia, and B. J. Alder. The consistent Boltzmann algorithm for the van der Walls equation of state. *Physica A*, 240:196–201, 1997.

[6] M. P. Allen and D. J. Tildesley. *Computer Simulation of Liquids*. Clarendon Press, Oxford, 1987.

[7] J. Alonso and H. J. Herrmann. Shape of the tail of a two-dimensional sandpile. *Phys. Rev. Lett.*, 76(26):4911–4914, June 1996.

[8] G. W. Baxter and R. P. Behringer. Cellular automata models for granular flow. *Phys. Rev. A (Rapid Communication)*, 42(2):1017, 1990.

[9] G. W. Baxter and R. P. Behringer. Cellular automata models for the flow of granular materials. *Physica D*, 51:465, 1991.

[10] A. Bershadskii. Stochastic density waves of granular flows: strong-intermittent dissipation fields with self-organization. *Physica A*, 210:386–390, 1994.

[11] G. A. Bird. *Molecular Dynamics and the Direct Simulation of Gas Flow*. Oxford Science Publications, Oxford, 1994.

[12] T. Boutreux and P. de Gennes. Surface flows of granular mixtures: I. general principles and minimal model. *J. Phys. I France*, 6:1295–1304, Oct. 1996.

[13] C. E. Brennen, S. Ghosh, and C. Wassgren. Vertical oscillation of a bed of granular material. In C. Thornton, editor, *Powders & Grains 93*, Rotterdam, 1993. Balkema.

[14] J. J. Brey, M. J. Ruiz-Montero, and D. Cubero. Homogeneous cooling state of a low-density granular flow. *Physical Review E*, 54:3664–71, Oct. 1996.

[15] M. Brune, J. Gehring, and A. Reinefeld. Heterogenous message passing and a link to resource management. *Journal of Supercomputing*, 1:1–17, 1997.

[16] C. S. Campbell and C. E. Brennen. Computer simulation of granular shear flow. *J. Fluid Mech.*, 151:167–188, 1985.

[17] E. Clément, S. Luding, A. Blumen, J. Rajchenbach, and J. Duran. Fluidization, condensation and clusterization of a vibrating column of beads. *Int. J. of Mod. Phys. B*, 7(9 & 10):1807–1827, 1993.

[18] I. S. Committee. IMPI - interoperable message-passing interface. http://impi.nist.gov/.

[19] P. Cundall and O. Strack. A discrete numerical model for granular assemblies. *Géotechnique*, 29(1):47, 1979.

[20] P. A. Cundall. A computer model for simulating progressive, large-scale movements in blocky rock systems. In *Proc. Symp. Int. Rock Mech.*, volume 2, Nancy, 1971.

[21] J. Dufty, A. S. A, and J. Brey. Practical kinetic model for hard sphere dynamics. *Physical Review Letters*, 77(7):1270–1273, 1996.

[22] J. Duran, T. Mazozi, S. Luding, E. C. E, and J. Rajchenbach. Discontinuous decompaction of a falling sandpile. *Physical Review A*, 53(2):1923–1930, 1996.

[23] G. E. Fagg, J. J. Dongarra, and A. Geist. Heterogeneous MPI application interoperation and process management under PVMPI. In M. Bubak, J. Dongarra, and J. Wasniewski, editors, *Recent Advances in Parallel Virtual and Message Passing Interface*, pages 91–98. Springer, 1997.

[24] I. Foster, J. Geisler, W. Gropp, N. karonis, E. Lusk, G. Thiruvathukal, and S. Tuecke. Wide-area implementation of the message passing standard. *Parallel Computing*, 24, 1998.

[25] I. Foster and C. Kesselman. The globus project: A status report. In *Proc. IPPS/SPDP '98 Heterogeneous Computing Workshop*, pages 4–18, 1998.

[26] D. Frenkel and B. Smit. *Understanding molecular simulations: from algorithms to applications*. Academic Press, 1996.

[27] U. Frisch, B. Hasslacher, and Y. Pomeau. Lattice-gas automata for the Navier-Stokes equation. *Phys. Rev. Lett.*, 56(14):1505–8, 1986.

[28] E. Gabriel, M. Resch, T. Beisel, and R. Keller. Distributed computing in a heterogenous computing environment. In *Recent Advances in Parallel Virtual Machine and Message Passing Interface*, Lecture Notes in Computer Science. Springer, 1998.

[29] E. Gabriel, M. Resch, and R. Rühle. Implementing MPI with optimized algorithms for metacomputing. In *Message Passing Interface Developer's and Users Conference (MPIDC'99)*, pages 31–41, Atlanta, March 10-12 1999.

[30] A. L. Garcia, F. J. Alexander, and B. J. Alder. A particle method with adjustable transport properties - the generalized consistent Boltzmann algorithm. *Journal of Statistical Physics*, 89(1/2):403–409, 1997.

[31] I. Goldhirsch, M.-L. Tan, and G. Zanetti. A molecule dynamical study of granular fluids. I. The unforced granular gas in two dimensions. *Journal of Scientific Computing*, 8:1–40, March 1993.

[32] I. Goldhirsch and G. Zanetti. Clustering instability in dissipative gases. *Physical Review Letters*, 70:1619–22, March 1993.

[33] Y. Grasselli, H. J. Herrmann, G. Oron, and S. Zapperi. Effect of impact energy on the shape of granular heaps. cond-mat/9907364, 1999.

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

[35] K. P. Hadeler and C. Kuttler. Dynamcal models for granular matter. *Granular Matter*, 2:9–18, 1999.

[36] P. K. Haff. Grain flow as a fluid-mechanical phenomenon. *J. Fluid Mech.*, 134:401–430, 1983.

[37] S. Haney. Is C++ fast enough for scientific computing? *Computers in Physics*, 8(6):690–694, Nov/Dec 1994.

[38] J. P. Hansen and I. R. McDonald. *Theory of Simple Liquids.* Academic Press, London, 2nd edition, 1986.

[39] S. Harris. *An Introduction to the Theory of the Boltzmann Equation.* HOLT, RINEHART AND WINSTON, INC, New York, 1971.

[40] H. Hayakawa and K. Nakanishi. Universal behavior in granular flows and traffic flows. *Prog. Theor. Phys. Suppl.*, 130:57–75, 1998.

[41] J. Hemmingsson, H. J. Herrmann, and S. Roux. Vectorial cellular automaton for the stress in granular media. *Journal de Physique I*, 7(2):291–302, 1997.

[42] H. J. Herrmann. Spontaneous density fluctuations in granular flow and traffic. In J. Parisi, S. Muller, and W. Zimmermann, editors, *Nonlinear physics of complex systems. Current status and future trends*, pages 23–34. Springer-Verlag, Berlin, 1996.

[43] H. J. Herrmann, E. Flekkoy, K. Nagel, G. Peng, and G. Ristow. Density waves in dry granular flow. In D. Wolf, M. Schreckenberg, and A. Bachem, editors, *Workshop on Traffic and Granular Flow*, pages 239–250. World Scientific, 1996.

[44] H. J. Herrmann and S. Luding. Modeling granular media with the computer. *Continuum Mechanics and Thermodynamics*, 10:189–231, 1998.

[45] H. J. Herrmann and M. Müller. How much can we simplify a system of grains? In *Granular Gases*, Lecture Notes in Physics. Springer, 1999.

[46] H. J. Herrmann and M. Müller. Simulations of granular media. In N. Attig and R. Esser, editors, *Molecular Dynamics on Parallel Computers*. World Scientific, 1999.

[47] H. J. Herrmann and M. Müller. Simulations of granular materials on different scales. *Computer Physics Communications*, 127:120–125, 2000.

[48] R. Hockney and M. Berry. Report-1. Technical report, PARKBENCH Committee, 1994. http://www.netlib.org/parkbench/.

[49] K. Höfler. Räumliche Simulation von Zweiphasenflüssen. Master's thesis, Universität Stuttgart, 1997.

[50] K. Höfler, E. Kuusela, C. Manwart, R. Mück, and S. Schwarzer. Container size dependence of the velocity fluctuations in suspensions of monodisperse spheres. In E. Krause and W. Jäger, editors, *High Performance Computing in Science and Engineering*. Springer, Berlin, 2000.

[51] K. Höfler, M. Müller, and S. Schwarzer. Design and application of object oriented parallel data structures in particle and continuous systems. In E. Krause and W. Jäger, editors, *High Performance Computing in Science and Engineering*. Springer, Berlin, 1999.

[52] K. Höfler, M. Müller, S. Schwarzer, and B. Wachmann. Interacting particle-liquid systems. In E. Krause and W. Jäger, editors, *High Performance Computing in Science and Engineering*. Springer, Berlin, 1998. [Trans. HLR Stuttgart, 1998].

[53] S. Horikawa, T. Isoda, T. Nakayama, A. Nakahara, and M. Matsushita. Self-organized critical density waves of granular particles flowing through a pipe . *Physica A*, 233(3-4):699–708, Dec 1996.

[54] S. Horikawa, A. Nakahara, T. Nakayama, and M. Matsushita. Self-organzized critical desnity waves of granular material flowing through a pipe. *J. Phys. Soc. Jpn.*, 64(6):1870–1873, June 1995.

[55] T. Ihle and D. M. Kroll. Stochastic rotation dynamics: A galilean-invariant mesoscopic model for fluid flow. *Pysical Review E*, 63:20201–1 – 20201–4, 2001.

[56] J. T. Jenkins and M. W. Richman. Kinetic theory for plane flows of a dense gas of identical, rough, inelastic, circular disks. *Phys. of Fluids*, 28:3485, 1985.

[57] N. M. Josuttis. *The C++ Standard Library: A Tutorial and Reference.* Addison-Wesley, 1999.

[58] A. Karolyi and J. Kertesz. Hydrodynamic cellular automata for granular media. In R. Gruber and M. Tomassini, editors, *Proceedings of the 6th Joint EPS-APS International Conference on Physics Computing. PC'94. Physics Computing '94*, pages 675–681. Eur. Phys. Soc, 1994.

[59] T. Kimura and H. Takemiya. Local area metacomputing for multidisciplinary problems: A case study for fluid/structure coupled simulations. In *12th ACM International Conference on Supercomputing*, Melbourne, July 1998.

[60] M. Lätzel, S. Luding, and H. J. Herrmann. Macroscopic material properties from quasi-static, microscopic simulations of a two-dimensional shear-cell. *Granular Matter*, 2(3):123–135, 2000. cond-mat/0003180.

[61] J. Lee. Scaling behavior of granular particles in a vibrating box. *Physica A*, 219:305–326, 1995.

[62] B. D. Lubachevsky. How to simulate billards and similar systems. *J. of Comp. Phys.*, 94(2):255, 1991.

[63] S. Luding. Granular materials under vibration: Simulations of rotating spheres. *Phys. Rev. E*, 52(4):4442, 1995.

[64] S. Luding. Stress distribution in static two dimensional granular model media in the absence of friction. *Phys. Rev. E*, 55(4):4720–4729, 1997.

[65] S. Luding, E. Clément, A. Blumen, J. Rajchenbach, and J. Duran. Anomalous energy dissipation in molecular dynamics simulations of grains: The "detachment effect". *Phys. Rev. E*, 50:4113, 1994.

[66] S. Luding, E. Clément, A. Blumen, J. Rajchenbach, and J. Duran. Studies of columns of beads under external vibrations. *Phys. Rev. E*, 49(2):1634, 1994.

[67] S. Luding, E. Clément, A. Blumen, J. Rajchenbach, and J. Duran. Interaction laws and the detachment effect in granular media. In *Fractal Aspects of Materials*, volume 367, pages 495–500, Pittsburgh, Pennsylvania, 1995. Materials Research Society, Symposium Proceedings.

[68] S. Luding, E. Clement, J. Rajchenbach, and J. Duran. Simulations of pattern formation in vibrated granular media. *Europhysics Letters*, 36(4):247–252, 1996.

[69] S. Luding, J. Duran, E. Clement, and J. Rajchenbach. Simulations of dense granular flow - dynamic arches and spin organization. *Journal de Physique I*, 6(6):823–836, 1996.

[70] S. Luding, H. J. Herrmann, and A. Blumen. Simulations of two-dimensional arrays of beads under external vibrations: scaling behavior. *Physical Review E*, 50(4):3100–3108, 1994.

[71] S. Luding and S. McNamara. How to handle the inelastic collapse of a dissipative hard-sphere gas with the TC model. *Granular Matter*, 1(3):111, 1998. cond-mat/9810009.

[72] S. Luding, M. Müller, and S. McNamara. The validity of "molecular chaos" in granular flows. In *World Congress on Particle Technology*, Davis Building, 165-189 Railway Terrace, Rugby CV21 3HQ, UK, 1998. Institution of Chemical Engineers. CD: ISBN 0-85295-401-9.

[73] H.-G. Matuttis. Simulations of the pressure distribution under a two dimensional heap of polygonal particles. *Granular Matter*, 1(2):83–91, 1998.

[74] H.-G. Matuttis and A. Schinner. Influence of the geometry on the pressure distribution of granular heaps. *Granular Matter*, 1(4):195–201, 1999.

[75] R. Mazighi, B. Bernu, and F. Delyon. Steady state of a column of shaked inelastic spheres. *Phys. Rev. E*, 50:4551, 1994.

[76] S. McNamara and W. Young. Inelastic collapse in two dimensions. *Phys. Rev. E*, 50(1):R28–R31, July 1994.

[77] S. McNamara and W. Young. Dynamics of a freely evolving, two-dimensional granular medium. *Physical Review E [Statistical Physics, Plasmas, Fluids, and Related Interdisciplinary Topics]*, 53:5089–100, May 1996.

[78] S. McNamara and W. R. Young. Inelastic collapse and clumping in a one-dimensional granular medium. *Phys. Fluids A*, 4(3):496, 1992.

[79] N. Mitarai and H. Nakanishi. Stability analysis of optimal velocity model for traffic and granular flow under open boundary conditions. *J. Phys. Soc. Japan*, 68(8):2475–2478, August 1999.

[80] H. M.Jaeger, S. R. Nagel, and R. P. Behringer. The physics of granular materials. *Physics Today*, 49(4):32–38, 1996.

[81] J. Monaghan. Smoothed particle hydrodynamics. *Annu. Rev. Astron. Astrophys.*, 30:543–574, 1992.

[82] J. Montanero, A. Santos, and V. Garzo. Monte Carlo simulation of the Boltzmann equation for uniform shear flow. *Physics of Fluids*, 8(7):1981–1983, 1996.

[83] J. M. Montanero and A. Santos. Monte Carlo simulation method for the Enskog equation. *Physical Review E*, 54(1):438–444, 1996.

[84] O. Moriyama, N. Kuroiwa, and M. Matsushita. 4/3 law of granular particles flowing through a vertical pipe. *Phys. Rev. Lett.*, 80(13):2833–2836, March 1998.

[85] J. P. Morris, P. J. Fox, and Y. Zhu. Modeling low reynolds number incompressible flows using sph. *Journal of Computational Physics*, 136:214–226, 1997.

[86] M. Müller. Molecular Dynamics with C++. In F. Bassetti, K. Davis, and B. Mohr, editors, *Proceedings of the Workshop on Parallel Object Oriented Scientific Programming*. Technical Report FZJ-ZAM-IB-9906, Lisbon/Portugal, 1999.

[87] M. Müller. Abstraction benchmarks and performance of C++ applications. In *The Fourth International Conference on Supercomputing in Nuclear Applications*, TOKYO/JAPAN, 2000.

[88] M. Müller and H. J. Herrmann. DSMC - a stochastic algorithm for granular matter. In H. J. Herrmann, J.-P. Hovi, and S. Luding, editors, *Physics of dry granular media*, NATO ASI Series, Dordrecht, 1998. Kluwer Academic Publisher.

[89] M. Müller, U. Lang, and M. Resch. Libraries, applications and visualisation in metacomputing. In *Proceedings of ISThmus 2000*, pages 373–380, Poznan/Poland, April 2000.

[90] M. Müller, S. Luding, and H. J. Herrmann. Simulations of vibrated granular media in 2d and 3d. In D. E. Wolf and P. Grassberger, editors, *Friction, Arching and Contact Dynamics*. World Scientific, Singapore, 1997.

[91] M. Müller and M. Resch. PE mapping and the congestion problem on the T3E. In H. Lederer and F. Hertweck, editors, *Proceedings of the Fourth European Cray-SGI MPP Workshop*, pages 20–28. IPP, Garching, Germany, September 1998. see http://www.rzg.mpg.de/mpp-workshop/proceedings.html.

[92] A. Nakahara and T. Isoda. $1/f^{\alpha}$ density fluctuation at the slugging transition point of granular flows through a pipe . *Phys. Rev. E*, 55(4):4264–4273, April 1997.

[93] K. Nanbu. Direct simulation derived from the Boltzmann equation. *J. Phys. Soc. Japan*, 49:2042–2058, 1980.

[94] K. Nanbu. Direct simulation derived from the boltzmann equation. ii. rough sphere gases. *J. Phys. Soc. Japan*, 49:2050–2054, 1980.

[95] I. S. Organization. Programming languages - C++, iso/iec 14882. Technical report, ISO/ANSI, 1998.

[96] J. A. G. Orza, R. Brito, T. P. C. van Noije, and M. H. Ernst. Patterns and long range correlations in idealized granular flows. *Int. J. of Mod. Phys. C*, 8:953, 1997.

[97] G. Peng and H. J. Herrmann. Density waves of granular flow in a pipe using lattice-gas automata. *Phys. Rev. E*, 49(3):1796, 1994.

[98] G. Peng and H. J. Herrmann. Density waves and 1/f density fluctuations in granular flow. *Phys. Rev. E*, 51(3):1745–1756, March 1995.

[99] G. Peng and T. Ohta. Velocity and density profiles of granular flow in channels using lattice gas automaton. *Phys. Rev. E*, 55:6811, 1997.

[100] S. Pickles, J. Brooke, F. Costen, E. Gabriel, M. Müller, M. Resch, and S. Ord. Metacomputing across intercontinental networks. *Future Generation Computer Systems*, 17(8):911–918, June 2001.

[101] S. Pickles, F. Costen, J. Brooke, E. Gabriel, M. Müller, M. Resch, and S. Ord. The problems and the solutions of the metacomputing experiments in SC'99. In *Proceedings of HPCN 2000*, May 2000.

[102] S. Plimpton. Fast parallel algorithms for short-range molecular dynamics. *Journal of Computational Physics*, 117:1–19, 1995.

[103] T. Pöschel. Recurrent clogging and density waves in granular material flowing through a narrow pipe. *J. Phys. I France*, 4(4):499–506, 1994.

[104] T. Pöschel and H. J. Herrmann. Size segregation and convection. *Europhys. Lett.*, 29:123, 1995.

[105] H. Puhl. *Eine Studie zur Simulation granularer Medien.* PhD thesis, Universität Stuttgart, 1998.

[106] F. Radjai, M. Jean, J.-J. Moreau, and S. Roux. Force distributions in dense two-dimensional systems. *Phys. Rev. Lett.*, 77(2):274–277, July 1996.

[107] F. Radjai and D. E. Wolf. Features of static pressure in dense granular media. *Granular Matter*, 1(1):3–8, 1998.

[108] M. Resch, H. Berger, and T. Boenisch. A comparison of MPI performance on different MPPs. In M. Bubak, J. Dongarra, and J. Wasniewski, editors, *Recent Advances in Parallel Virtual Machine and Message Passing Interface*, Lecture Notes in Computer Science, pages 75–83, Heidelberg, 1997. Springer.

[109] M. Resch, T. Boenisch, and H. Berger. Performance of MPI on a cray T3E. Technical report, Third European CRAY-SGI MPP Workshop, Paris (France), Sept. 1997.

[110] M. Resch, D. Rantzau, and R. Stoy. Metacomputing experience in a transatlantic wide area application testbed. *Future Generation Computer Systems*, 15(5-6):807–816, 1999.

[111] M. M. Resch, T. Beisel, H. Berger, K. Bidmon, E. Gabriel, R. Keller, and D. Rantzau. Clustering T3Es for metacomputing applications. In *Cray User Group Conference*, Germany, 1998.

[112] A. D. Robison. C++ gets faster for scientific computing. *Computers in Physics*, 10:458–462, 1996.

[113] A. D. Rosato and Y. Lan. Discrete element modeling of vibrating granular beds. In C. Thornton, editor, *Powders & Grains 93*, page 241, Rotterdam, 1993. Balkema.

[114] S. L. Scott. Synchronisation and comunication in the T3E. In *Int. Conf. on Architectural Support for Programming Languages and Operating Systems proceedings*, volume 31, pages 26–36, October 1996.

[115] S. L. Scott and G. M. Thorson. Adaptive routing in a high performance 3d torus. In *HOT Interconnects*, volume IV. Stanford University, August 1996.

[116] J. Stadler, R. Mikulla, and H.-R. Trebin. IMD: A software package for molecular dynamics studies on parallel computers. *Int. J. Mod. Phys. C*, 8:1131–1140, 1997.

[117] P. L. Tallec and F. Mallinger. Coupling Boltzmann and navier stokes equations by half fluxes. *Journal of Computational Physics*, 136:51–67, 1997.

[118] M.-L. Tan and I. Goldhirsch. Intercluster interactions in rapid granular shear flows. *J. Fluid Mech.*, 9(4):856–869, 1997.

[119] M.-L. Tan and I. Goldhirsch. Rapid granular flows as mesoscopic systems. *Phys. Rev. Lett.*, 81(14):3022–3025, 1998.

[120] T. van Noije, M. Ernst, R. Brito, and J. Orza. Mesoscopic theory of granular fluids. *Phys. Rev. Lett.*, 79(3):411–414, July 1997.

[121] T. Veldhuizen. Scientific computing: C++ vs. Fortran. *Dr. Dobb's Journal*, November 1997.

[122] S. Vollmar and H. J. Herrmann. Settling of particles studied by a hydrodynamic cellular automaton for granular media. *Physica A*, 215:411–419, 1995.

[123] B. Wachmann and S. Schwarzer. Three-dimensional massively parallel computing of suspensions. *Int J. Mod Phys. C*, 1998.

[124] B. E. Wachmann. *Microsimulation of Two-Phase Systems*. PhD thesis, Stuttgart University, 1999.

[125] L. Wang and S. Jacques. Hybrid model of monte carlo simulation and diffusion theory for light reflectance by turbid media. *J. Optical Soc. Am. A*, 10:1746–1752, 1993.

[126] S. Warr, J. Huntley, and G. Jacques. Fluidization of a two-dimensional granular system: experimental study and scaling behavior. *Physical Review E*, 52(5):5583–5595, 1995.

[127] J. Wittmer, P. Claudin, M. Cates, and J. Bouchaud. An explanation for the central stress minimum in sand piles. *Nature*, 382(6589):336–338, 1996.

[128] D. Wolf, M. Schreckenberg, and A. Bachem, editors. *Workshop on Traffic and Granular Flow*. World Scientific, 1996.

Index

Acknowledgment

At the end of my thesis I would like to use the opportunity to thank all the people who made this thesis possible.

I would like to thank Stefan Luding, who at the beginning guided this work and provided most comparison data from ED. Stefan Luding, Thomas Ihle and Andrea Germer helped me with their proof-reading and comments. I am also indebted to Hans Herrmann for the opportunity to work at the ICA1. My gratitude also goes to Hans-Rainer Trebin and Ulrich Kneißl who find a date for my examination in their tight schedule and made it an enjoyable event.

My thank also goes to all the people who shared the burden of system administration with me, especially to Harald Puhl and Stefan Schwarzer.

I acknowledge the financial support by the Deutsche Forschungsgemeinschaft, Sonderforschungsbereich 382 and the access to the computer resources at the research center NIC in Jülich and the federal computer-center HLRS in Stuttgart. The metacomputing experiments would not have been possible without the support of Pittsburgh Supercomputing Center, Sandia National Laboratory and Tsukuba Advanced Computing Center.

Finally, I would like to express my deepest gratitude for the constant support, understanding and love that I received from my parents.

B. Deutsche Zusammenfassung

Die Größe granularer Teilchen reicht von Bruchteilen eines Millimeters in Staub bis hin zu Kilometern in planetaren Ringen. Systeme von industrieller Größenordnung enthalten Millionen von Teilchen, selbst kleinere experimentelle Anordnungen bestehen aus Tausenden von Teilchen. Aufgrund der Teilchengröße und ihrer Anzahl ist es prinzipiell möglich, die Systeme mit der klassischen statistischen Mechanik zu beschreiben.

B.1. Theorie und Modelle

B.1.1. Statistische Mechanik

Das Ziel der statistischen Mechanik ist, beobachtbare Eigenschaften eines makroskopischen Systems durch Berechnung von Zeit- oder Ensemblemittelwerten zu berechnen. Den zweiten Zugang bezeichnet man als Gibbs-Formalismus. Bei diesem Zugang wird die Verteilung eines Ensembles durch die Wahrscheinlichkeitsdichte $f^{(N)}(\mathbf{x}^N, \mathbf{v}^N)$ im Phasenraum beschrieben. Die Zeitentwicklung von f gehorcht der Liouville Gleichung:

$$\frac{\partial f^{(N)}}{\partial t} = \sum_{i=1}^{N} \left(\frac{\partial H_N}{\partial \mathbf{x}_i} \frac{\partial f^{(N)}}{\partial \mathbf{p}_i} - \frac{\partial H_N}{\partial \mathbf{p}_i} \frac{\partial f^{(N)}}{\partial \mathbf{x}_i} \right). \tag{B.1}$$

Die Funktion $f^{(N)}$ liefert dabei eine sehr detaillierte Beschreibung des Systems. Oft reicht es aus, nur eine Untermenge von n Teilchen zu betrachten. Die unnötige Information kann man durch Integration über die Koordinaten und Impulse der restlichen $N - n$ Teilchen eliminieren. Daraus resultiert die reduzierte Wahrscheinlichkeitsdichte $f^{(n)}$, die definiert ist als:

$$f^{(n)}(\mathbf{x}^n, \mathbf{p}^n, t) = \frac{N!}{(N-n)!} \int \int f^{(N)}(\mathbf{x}^N, \mathbf{p}^N, t) d\mathbf{x}^{(N-n)} d\mathbf{p}^{(N-n)}. \tag{B.2}$$

Wenn die Interaktion zwischen den Teilchen auf eine paarweise Wechselwirkung beschränkt ist, kann die resultierende Kraft $\mathbf{F}_i$, die auf ein Teilchen i wirkt, geschrieben werden als Summe einer externen Kraft $\mathbf{F}_{ij}$ und den Paarkräften $\mathbf{F}_{ij}$. Die Liouville Gleichung lautet damit

$$\frac{\partial f^{(N)}}{\partial t} + \frac{1}{m} \sum_{i=1}^{N} \mathbf{p}_i \frac{\partial f^{(N)}}{\partial \mathbf{x}_i} + \sum_{i=1}^{N} \mathbf{F}_i^{ext} \frac{\partial f^{(N)}}{\partial \mathbf{p}_i} = \sum_{i=1}^{N} \sum_{j=1}^{N} \mathbf{F}_{ij} \frac{\partial f^{(N)}}{\partial \mathbf{p}_i}. \tag{B.3}$$

Man integriert diese Gleichung über $(N-n)$ Positionen und Impulse und setzt Gleichung (2.5) ein. Zusätzlich nutzt man die Symmetrie von $f^{(N)}$ in Bezug auf Teilchenaustausch und die Tatsache, daß $f^{(N)}$ verschwindet, wenn entweder der Impuls unendlich ist, oder der Ort außerhalb des Systemvolumens ist. Sammelt man nun die Terme $f^{(n)}$ auf der linken Seite, so ergibt sich als Zusammenhang zwischen $f^{(n)}$ und $f^{(n+1)}$

$$\begin{aligned}&\left\{\frac{\partial}{\partial t}+\sum_{i=1}^{n}\left[\frac{1}{m}\mathbf{p}_i\frac{\partial}{\partial \mathbf{x}_i}+\left(\mathbf{F}_i^{ext}+\sum_{j=1}^{n}\mathbf{F}_{ij}\right)\frac{\partial}{\partial \mathbf{p}_i}\right]\right\} f^{(n)}(\mathbf{x}^n,\mathbf{p}^n,t)\\ &=-\sum_{i=1}^{n}\int\int \mathbf{F}_{i,n+1}\frac{\partial}{\partial \mathbf{p}_i}f^{(n+1)}(\mathbf{x}^{(n+1)},\mathbf{p}^{(n+1)},t)d\mathbf{x}_{n+1}d\mathbf{p}_{n+1}.\end{aligned} \tag{B.4}$$

Dieser Satz von Gleichungen ist als BBGKY Hierarchie bekannt. Setzt man $n=1$ so erhält man eine Gleichung für die Einteilchen-Wahrscheinlichkeitsdichte. Das Problem ist aber, daß man zur Berechnung von $f^{(1)}$ Kenntnis von $f^{(2)}$ benötigt. Mit Hilfe der sogenannten Annahme des molekularen Chaos $f^{(2)}(\mathbf{x}_1,\mathbf{p}_1,\mathbf{x}_2,\mathbf{p}_2)=f^{(1)}(\mathbf{x}_1,\mathbf{p}_1)f^{(1)}(\mathbf{x}_2,\mathbf{p}_2)$ erhält man eine geschlossene Gleichung, die sogenannte Vlasov Gleichung:

$$\left(\frac{\partial}{\partial t}+\frac{1}{m}\mathbf{p}_1\frac{\partial}{\partial \mathbf{x}_1}+\mathbf{F}_1^{ext}\frac{\partial}{\partial \mathbf{p}_1}\right)f^{(1)}(\mathbf{x}_1,\mathbf{p}_1,t)=-\bar{\mathbf{F}}(\mathbf{x},t), \tag{B.5}$$

wobei $\bar{\mathbf{F}}(\mathbf{x},t)=\int\int \mathbf{F}(\mathbf{x},\mathbf{x}_2)f^{(1)}(\mathbf{x}_2,\mathbf{p}_2,t)d\mathbf{x}_2d\mathbf{p}_2$ die mittlere Kraft der restlichen Teilchen auf ein Teilchen an Position $\mathbf{x}$ zur Zeit t ist.

B.1.2. Boltzmann Gleichung

Die Boltzmann Gleichung beschreibt das System mit der Einteilchenverteilungsfunktion f, wobei $f(\mathbf{x},\mathbf{v},t)$ die Wahrscheinlichkeit dafür ist, ein Teilchen zur Zeit t an der Position $\mathbf{x}$ mit der Geschwindigkeit $\mathbf{v}$ vorzufinden.

$$\frac{\partial f(\mathbf{x},\mathbf{v},t)}{\partial t}=-\underbrace{\mathbf{v}\nabla_{\mathbf{r}}f}_{\text{Fluß}}-\underbrace{\frac{\mathbf{F}}{m}\nabla_{\mathbf{v}}f}_{\text{Externe Kräfte}}+\underbrace{\left(\frac{\partial f}{\partial t}\right)_{\text{coll}}}_{\text{Kollisionen}} \tag{B.6}$$

mit

$$\left(\frac{\partial f}{\partial t}\right)_{\text{coll}}=\int d\mathbf{x}'\int d\mathbf{v}'\frac{\partial U(\mathbf{x},\mathbf{x}')}{\partial \mathbf{x}}\frac{\partial}{\partial \mathbf{v}}f_2(\mathbf{x},\mathbf{v},\mathbf{x}',\mathbf{v}') \tag{B.7}$$

Nimmt man molekulares Chaos an, d.h. daß die Teilchen unkorreliert sind, dann erhält man durch Einsetzen von $f_2=f_1f_1$ eine geschlossene Gleichung.

$$\left(\frac{\partial f(\mathbf{x},\mathbf{v}_1,t)}{\partial t}\right)_{\text{coll}}=\frac{\sigma_T}{2}\int\limits_{\hat{\mathbf{k}}\mathbf{v}_{12}>0}d\hat{\mathbf{k}}d\mathbf{v}_2(\hat{\mathbf{k}}\mathbf{v}_{12})\left(\underbrace{f(\mathbf{v}_1^*)f(\mathbf{v}_2^*)}_{\text{Gewinn}}-\underbrace{f(\mathbf{v}_1)f(\mathbf{v}_2)}_{\text{Verlust}}\right). \tag{B.8}$$

Im Gegensatz zur Vlasov Gleichung (B.5) wird dabei nicht eine mittlere Kraft auf ein Teilchen, sondern eine Wechselwirkung über Kollisionen angenommen.

B.1.3. HSMC Algorithmus

Der HSMC basiert wie DSMC auf der numerischen Lösung der Boltzmann Gleichung:

$$\begin{aligned}\frac{\partial f(\mathbf{x},\mathbf{v},t)}{\partial t} &= -\left(\mathbf{v}\nabla_{\mathbf{r}} + \frac{\mathbf{F}}{m}\nabla_{\mathbf{v}}\right) f + \left(\frac{\partial f}{\partial t}\right)_{\text{coll}} \\ &= -\hat{D}f + \hat{C}f.\end{aligned} \tag{B.9}$$

Die Zeitentwicklung enthält dabei die Beitrage $\hat{D}$ durch die Bewegung der Teilchen, und die durch Kollisionen verursachten Änderungen $\hat{C}$.

$$\begin{aligned}f(t+\Delta t) &\approx f(t) + \left(\frac{\partial f(t)}{\partial t}\right)\Delta t \\ &\approx \left(1 - \Delta t\hat{D} + \Delta t\hat{C}\right) f(t) \\ &\approx (1+\Delta t\hat{C})(1-\Delta t\hat{D}) f(t).\end{aligned} \tag{B.10}$$

In dieser Näherung ist die Zeitentwicklung aufgeteilt in einen Advektionsschritt $1-\Delta t\hat{D}$, der auf f wirkt und einen darauf folgenden Kollisionsoperator $1+\Delta t\hat{C}$. Der Kollisionsschritt wird dabei im HSMC-Algorithmus ausgeführt wie in der CBA-Methode (siehe Kapitel 2.4.1 und 2.3.2). Der einzige Unterschied ist die Unterdrückung des zusätzlichen Offsets in vollen Zellen, da die entsprechende Korrektur bei diesen Zellen im Advektionsschritt vorgenommen wird. Für die Berechnung der Wechselwirkung werden die Teilchen zuerst in Zellen der Länge L und Volumen $V_c = L^d$ einsortiert. Kollisionen finden dabei nur zwischen Teilchen in derselben Zelle statt. In jeder Zelle, werden dabei

$$M_c = \chi \frac{N_c(N_c-1)\sigma v_{max}\tau}{2V_c} \tag{B.11}$$

potentielle Kollisionspaare ausgewählt. Die Kollision zwischen den Teilchen i und j wird dabei ausgeführt falls $\frac{|\mathbf{v}_i-\mathbf{v}_j|}{v_{max}} < Z$ gilt. Z ist eine Zufallszahl aus dem gleichverteilten Intervall $[0;1]$. Diese Methode führt zu einer Kollisionswahrscheinlichkeit, die proportional zur Relativgeschwindigkeit der beiden Teilchen ist. Da für die Auswahl der Kollisionspartner die Orte der beiden Teilchen unberücksichtigt bleiben, muß zur Berechnung der Geschwindigkeiten nach dem Stoß ein Stoßparameter zufällig gewählt werden. An diesem Punkt wird angenommen, daß die Orte und Geschwindigkeiten der Stoßpartner unkorreliert sind, was zu einer Gleichverteilung des Stoßparameters führt.

Der wesentliche Teil des Algorithmus ist die Berechnung der Bewegung der Teilchen. Im Gegensatz zu allen DMSC-Varianten findet die Bewegung eines Einzelteilchens nicht unabhängig von den anderen Teilchen statt, sondern die Teilchen interagieren indirekt über die Zellen, die auch für die Kollisionen benutzt worden sind. Um die Zunahme der Dichte über einen maximalen Wert hinaus zu verhindern, werden Zellen, die eine festgelegte Dichte überschritten haben, als „voll“ markiert. Bewegungen von Teilchen in eine bereits volle Zelle werden dann verworfen. Dadurch wird der Volumenausschluß auf Ebene der Zellen garantiert. Ein weiterer Nachteil der DSMC-Varianten ist die fehlende

Wechselwirkung zwischen den Zellen. Ein Teilchen, das sich am Rand einer Zelle befindet ist quasi blind gegenüber Teilchen in den Nachbarzellen. Vor allem in dichten Systemen führt dies zu einer effektiv geringeren Wechselwirkung der Teilchen untereinander. Aus diesem Grund wurde in HSMC eine Wechselwirkung zwischen den Zellen eingeführt. Dazu werden die Teilchen innerhalb einer Zelle zu einem Makroteilchen zusammengefaßt, dessen Impuls und Masse dem Gesamtimpuls bzw. Masse der Einzelteilchen entsprechen. Jedes dieser Makroteilchen hat dabei $2d$ potentielle Stoßpartner, die den direkten Nachbarzellen entsprechen. Mit dem Stoßpartner, der die größte Relativgeschwindigkeit besitzt, wird dann eine Kollision durchgeführt. Die Kollision folgt dabei bezüglich Dissipation und Stoßparameter denselben Regeln wie der Zusammenstoß zweier Teilchen. Die hier dissipierte kinetische Energie geht allerdings nicht verloren, sondern führt zu einer Erhöhung der granularen Temperatur. Dazu werden die Relativgeschwindigkeiten der Teilchen reskaliert. Eine genaue Beschreibung dieses Vorgangs befindet sich in Kapitel 2.3.2. Im nächsten Abschnitt wird gezeigt, daß diese Vorgehensweise auch für dichte Systeme zur korrekten Gesamtdissipation führt.

B.2. Physikalische Testfälle

Um den Algorithmus zu testen und zu verifizieren wurden verschiedene Testfälle ausgewählt. Jeder Fall testet dabei spezifische Eigenschaften granularer Medien. In ihrer Gesamtheit überdecken sie ein breites Spektrum verschiedener Phänomene.

B.2.1. Abkühlung homogener Systeme

Dieser erste Testfall dient zur Verifikation, ob die Dissipation und damit eine der grundlegendsten Eigenschaften granularer Materialen korrekt wiedergegeben wird. Das System besteht aus N dissipativen Teilchen, und entwickelt sich ohne Einfluß äußerer Kräfte. Die kinetische Anfangsenergie des Systems wird durch inelastische Stöße zwischen den Teilchen dissipiert. Bei jeder Kollision geht dabei der Anteil $\epsilon = 1 - r^2$ der kinetischen Energie im Schwerpunktssystem der Teilchen verloren, r wird als normaler Restitutionskoeffizient bezeichnet.

Solange das System homogen bleibt[31, 32, 77], folgt die kinetische Energie $E(t)$ dem Abkühlgesetz von Haff[36]:

$$\frac{E(t)}{E_0} = \left(\frac{1}{1 + t/t_0}\right)^2 \quad . \tag{B.12}$$

Die Zeitskala $t_0 = \frac{\sqrt{\pi} D \chi(\nu)}{\sqrt{2}(1-r^2)\nu\bar{v}}$ ist dabei eine Funktion der mittleren Teilchengeschwindigkeit $\bar{v} = \sqrt{2E_0/Nm}$, dem Teilchendurchmesser D, dem Restitutionskoeeffizienten r, und der Volumenfraktion ν, mit $\chi(\nu) = (1-\nu)^2/(1-7\nu/16)$.

Fig. B.1 zeigt die Übereinstimmung zwischen kinetischer Theorie und der HSMC-Methode. Wichtig ist die Übereinstimmung vor allem im Bereich hoher Dichten, wo die Originalmethode stark verändert worden ist. Ein möglicher Artefakt der Methode wäre

eine zu geringe Dissipation, da die Wechselwirkung zwischen vollen Zellen energieerhaltend ist. Die Übereinstimmung mit den theoretischen Werten zeigt, daß die Umwandlung von kinetischer Energie der Zellen in granulare Temperatur zu einer insgesamt korrekten Dissipation führt.

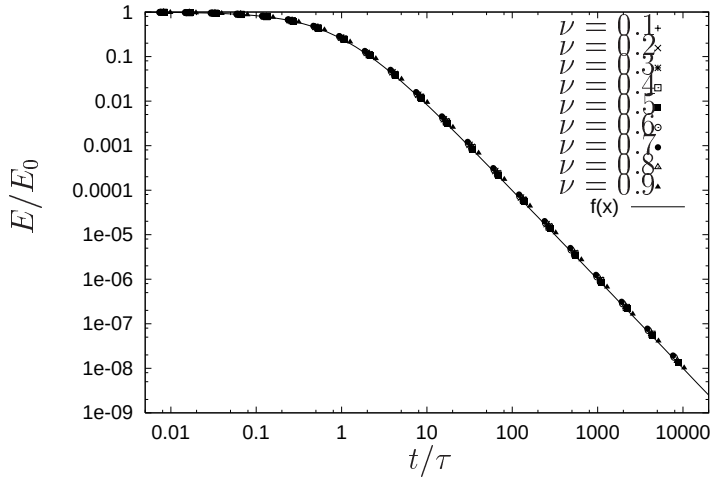


Abbildung B.1.: Abkühlung eines homogenen Systems granularen Gases für verschiedene Volumenfraktionen ν und konstantem Restitutionskoeffizienten $r = 0.99999$. f(x) bezieht sich auf Formel (B.12).

B.2.2. Cluster Instabilität

Ist ein anfänglich homogenes System granularen Gases ausreichend groß, kommt es zur spontanen Ausbildung von Clustern. Aufgrund der dissipativen Kollisionen kommt es in Gebieten, die aufgrund einer Fluktuation dichter sind, zu erhöhtem Energieverlust. Die reduzierte Temperatur resultiert in einem niedrigeren Druck, was weitere Teilchen in dieses Gebiet strömen läßt. Diese Tendenz existiert auch in kleinen Systemen, wird aber durch die Diffusion ausgeglichen. Erst wenn das System zu groß wird, ist die Diffusion zu langsam und das System wird durch die hydrodynamische Dichteinstabilität dominiert. Abb. B.2 zeigt, daß das System am Anfang wie ein homogenes System abkühlt. Mit dem Auftreten der ersten Cluster bei $t/t_0 \approx 2$ folgt die kinetische Energie einem anderen Verlauf. Erst bei $t/t_0 \approx 10$ weichen die Monte Carlo Verfahren von der harten Molekulardynamik ab. Der Grund für diese Abweichung sind die Korrelationen zwischen den Teilchen der dichten Cluster. Die Annahme einer konstanten Wahrscheinlichkeitsverteilung des Stoßparameters trifft nicht mehr zu, statt dessen finden mehr streifende Kollisionen statt. Dies führt zu einer stärkeren Dissipation in den Monte Carlo Methoden, da bei einer zentralen Kollision mehr Energie dissipiert wird. Das Auftreten der Instabilität und damit das qualitative Verhalten wird dadurch allerdings nur wenig beeinflußt, wie man an Abb. B.3 und den Strukturfaktoren dieser Systeme sehen kann.

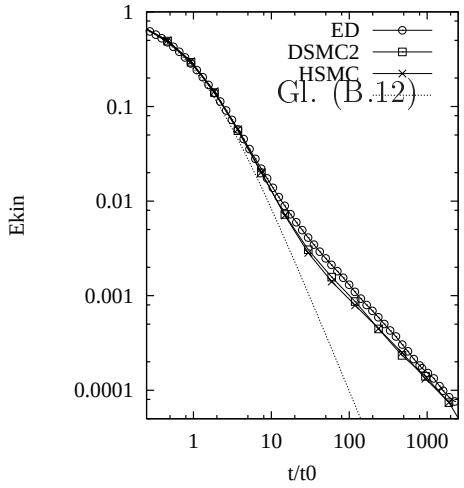


Abbildung B.2.: Zeitverlauf der normierten kinetischen Energie von ED, DSMC und HSMC Simulationen in 2D mit $N = 99856$, $\nu = 0.25$, und $r = 0.8$. Die gepunktete Linie stellt Gleichung B.12 dar.

B.2.3. Bagnold Scherfluß

Hierbei handelt es sich um ein klassisches Experiment zur Bestimmung der Viskosität μ. Die Kraft F, die benötigt wird, zwei Platten der Fläche A im Abstand l relativ zueinander mit der Geschwindigkeit v zu bewegen, folgt dabei dem Gesetz $F = \mu A \frac{v}{l}$. Da dem System durch die bei der Scherung verrichtete Arbeit Energie zugeführt wird, gibt es nur dann einen Gleichgewichtszustand, falls beispielsweise die Temperatur konstant gehalten wird. Bei granularen Systemen ist der Gleichgewichtszustand durch die Dissipation gewährleistet. Die Geschwindigkeit der Teilchen skaliert dabei mit der Geschwindigkeit der Wände. Dadurch ergibt sich das Bagnoldsche Gesetz $F \sim v^2$.

Mit HSMC konnte dieses Skalierungsverhalten über mehrere Größenordnungen nach-

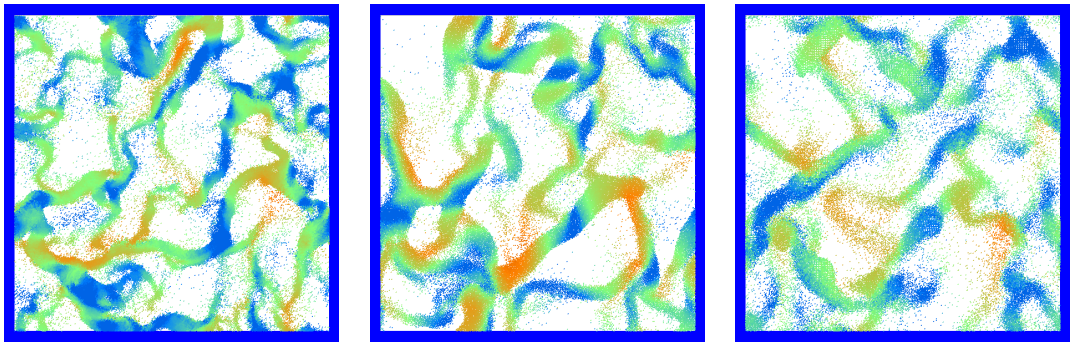


Abbildung B.3.: Abbildung eines geclusterten Systems von verschiedenen Simulationsmethoden: ED (links), DSMC2 (mitte), HSMC (rechts).

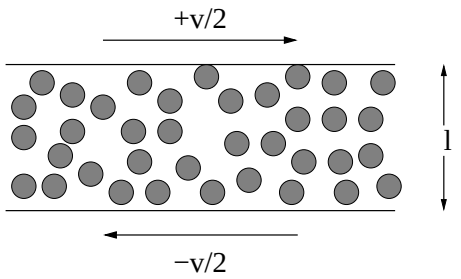


Abbildung B.4.: Aufbau für den Bagnold Scherfluß.

gewiesen werden. Aufgrund des linearen Geschwindigkeitsprofils senkrecht zu den Wänden, erstrecken sich die absoluten Geschwindigkeiten *eines* Systems ebenfalls über mehrere Größenordnungen. Simulationsmethoden, die nicht invariant unter Galileitransformationen sind, haben daher Schwierigkeiten diese Systeme zu simulieren. Durch das verwendete Gitter ist HSMC streng genommen nicht invariant unter Galileitransformationen. Das korrekte Skalierungsverhalten zeigt daher, daß es keine schwerwiegenden Artefakte durch das Gitter gibt.

B.2.4. Rohrfluß

Der vertikale Fluß von granularen Medien [1, 84, 99] durch ein Rohr war ein weiter Testfall. Im Gegensatz zu den bisher betrachteten Fällen wirkt hier die Gravitation als äußere Kraft auf die Teilchen. Da die Kollisionen zwischen den Teilchen und damit auch zwischen Teilchen und Wand bei HSMC rein stochastisch sind, stellt das korrekte Zusammenwirken von energiezuführender Gravitation und dissipativer Teilchenkollisionen eine neue Herausforderung für die Methode dar.

Von zentralem Interesse beim Rohrfluß sind die auftretenden spontanen Dichtewellen. HSMC reproduziert dabei sowohl diese Dichtewellen, als auch die Tatsache, daß deren spektrale Energiedichte einem Potenzgesetz $P(f) \sim f^{\alpha}$ folgt[53, 84, 98], wobei $\alpha \approx -\frac{4}{3}$ gilt. Die qualitative Abhängigkeit dieses Potenzgesetzes von der Stärke der Dissipation[98] wurde ebenfalls erhalten (siehe Fig. B.5). Die übereinstimmendes Resultate für den Exponenten durch die HSMC-Methode zeigen, dass ein umgebendes Fluid nicht notwendig ist, obwohl ein viskoser Reibungsterm Bestandteil der theoretischen Herleitung ist [84].

B.2.5. Statische Sandhaufen

Schüttet man Sand auf eine ebene Unterlage, dann bildet sich ein stabiler Haufen mit praktisch konstanter Neigung (siehe Abb. B.6). Da HSMC keine statische Reibung modelliert, ist die Frage, ob diese Methode überhaupt in der Lage ist, diesen statischen Endzustand korrekt zu simulieren. Bei der Verwendung von Molekulardynamik konnte gezeigt werden, dass Haufenbildung auch ohne statische Reibung möglich ist[64]. In

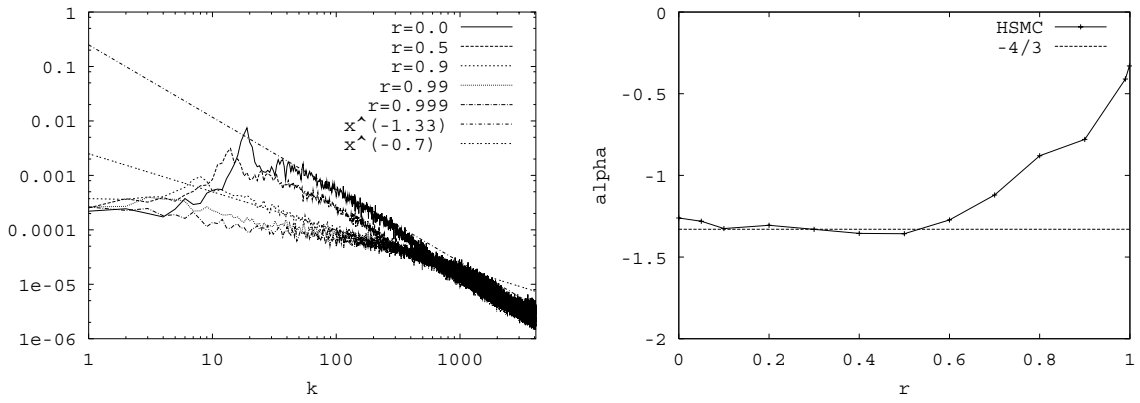


Abbildung B.5.: Spektrale Energiedichte der Dichtefluktuationen in einem Rohrfluß für verschiedene Restitutionskoeffizienten. Rechts ist der Exponent α des Potenzgesetzes in Abhängigkeit vom Restutitionskoeffizienten r aufgetragen. Deutlich ist zu erkennen, daß für $r \to 1$ der Exponent gegen eins strebt (weißes Rauschen).

diesem Fall wird der Haufen durch sterische Effekte aufgrund des Volumenausschlusses stabilisiert. Da HSMC diesen Volumenausschluß nur auf der Ebene der Zellen realisiert, ist zu prüfen, in welchem Maße Gitterartefakte auftreten.

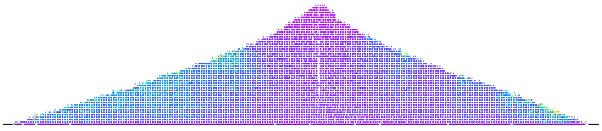


Abbildung B.6.: Sandhaufen aus 30500 Teilchen, der mit HSMC simuliert wurde.

Abb. B.6 zeigt, daß HSMC prinzipiell in der Lage ist, einen statischen Sandhaufen zu simulieren. Sichtbare Gitterartefakte gab es dabei nur bei extrem starker Dissipation ($r \approx 0$). Ein empfindlicherer Test ist, zu prüfen, ob gewisse Werte beim Neigungswinkel des Haufens bevorzugt werden, z.B. jene, bei denen die Steigung $m = \frac{\Delta y}{\Delta x}$ diophantischen Brüchen entspricht und damit optimal an das Gitter angepaßt ist. Dieser Effekt konnte nicht beobachtet werden (siehe Abb. B.7).

Betrachtet man die Oberfläche eines Sandhaufens genauer, so beobachtet man am unteren Ende des Haufens eine Abweichung der Oberfläche von der konstanten Steigung. Eine mögliches Modell zur Erklärung dieser Abweichung, ist die Vorstellung, daß der Haufen aus unvollständigen Lagen von Teilchen besteht. Die Oberfläche ist daher nicht eben, sondern besteht aus Stufen. Ist die Spitze des Haufens bei $x = 0$ und seine Höhe $h(0) = h_m$, so folgt eine ideal ebene Oberfläche dem Gesetz $h(x) = h_m - mx$. Unter der Annahme, daß die Dichte der Stufen in vertikaler Richtung konstant ist, und an jeder Stufe eine gewisse konstante Wahrscheinlichkeit r besteht, daß ein herunterrollendes

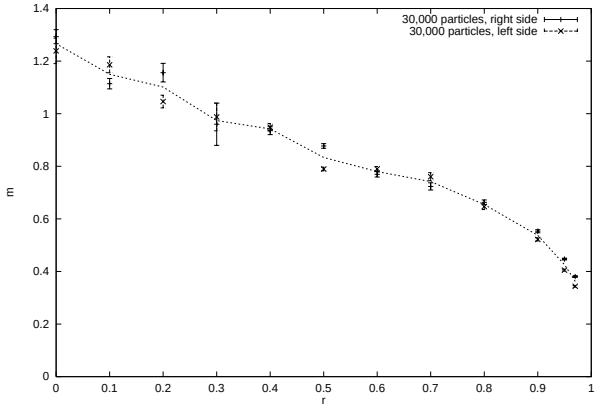


Abbildung B.7.: Böschungswinkel α als Funktion des Restitutionskoeffizienten (es gilt $m = \tan(\alpha)$).

Teilchen hängenbleibt, folgt für die Oberfläche des Haufens [7]:

$$x = \frac{h_m - h}{m} + \frac{l}{r} \ln(\frac{h_m}{h}). \tag{B.13}$$

Abb. B.8 zeigt, daß HSMC die Abweichung von der konstanten Steigung korrekt wiedergibt und mit Gleichung (B.13) gut übereinstimmt.

B.2.6. Korrelationen in dissipativen Systemen

Hier wurden die Dichte- und Geschwindigkeitskorrelationen untersucht, die in dissipativen Systemen existieren. Ein genaueres Verständnis ist hierbei wesentlich, um die Beschränkungen von HSMC zu verstehen. Es wurde gezeigt, wie durch den Einbau von Informationen, die durch andere Simulationsmethoden gewonnen wurde, der Anwendungsbereich von HSMC erweitert werden kann. Als Beispiel diente die Verteilung des Stoßparameters, dabei wurde die ursprünglich konstante Funktion durch Werte einer ED Simulation ersetzt. Die Methode bleibt allerdings durch die Existenz höherer Korrelationen in ihrem Anwendungsgebiet beschränkt.

B.3. Implementierung

Ein weiterer wesentlicher Teil der Arbeit war das Design und die Implementierung in C++. Das Ziel war, die Details und Schwierigkeiten der parallelen Implementierung zu verstecken, und gleichzeitig zu ermöglichen, die Software für andere Anwendungsgebiete wie z.B. der Molekulardynamik wiederverwenden zu können. Von entscheidender Bedeutung war dabei die Verwirklichung dieser Ziele ohne wesentliche Einbußen in der Performance.

Unter Verwendung der in C++ zu Verfügung stehenden Template-Mechanismen konnte das Programm allgemein genug geschrieben werden, um es nicht nur für die

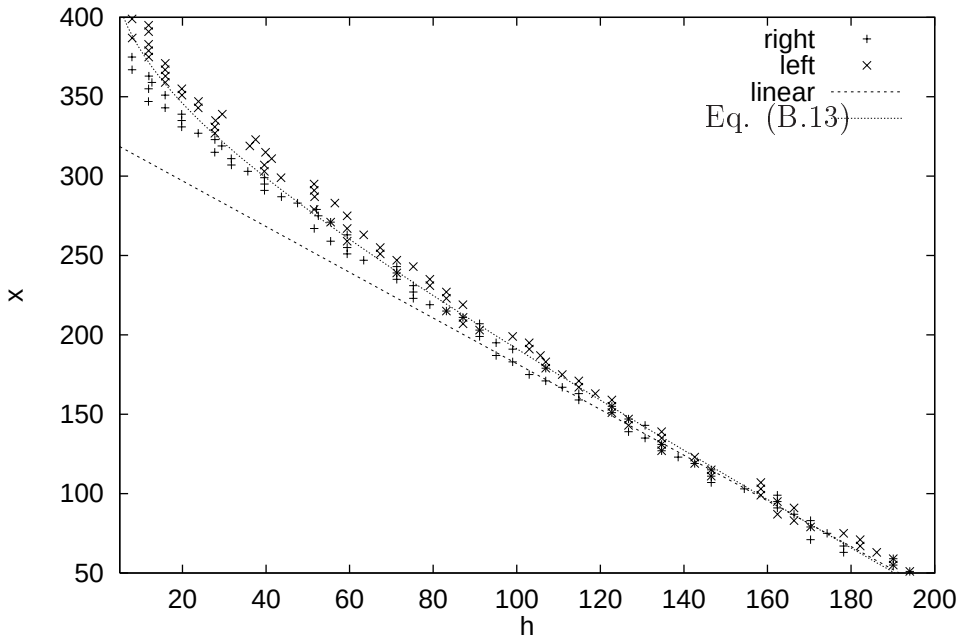


Abbildung B.8.: Vergleich der Form eines Sandhaufens auf einer Ebene mit einer analytischen Lösung nach Gleichung (B.13). $N = 30500$,$r = 0.9$

verschiedenen Varianten der Monte Carlo Simulationen (DSMC, CUBA, HSMC) zu verwenden, sondern auch für Molekulardynamiksimulationen. Der entwickelte parallele *Container*, der für die Speicherung und Verwaltung der Teilchen zuständig ist, versteckt dabei die Details der Parallelisierung mittels MPI. Damit wird es dem Anwender ermöglicht, sich auf die Modellierung der physikalischen Wechselwirkung zu konzentrieren und gleichzeitig einen leistungsfähigen Parallelrechner zu benutzen.

Die hohe Abstraktion, die bei diesem Zugang unvermeidlich ist, birgt natürlich die Gefahr von Performance-Einbußen. Aus diesem Grund wurde ein Vergleich mit einem optimierten MD Programms [116] in der Programmiersprache C durchgeführt. Dieser zeigte eine praktisch identische Geschwindigkeit der beiden Programme.

Inzwischen wird der entwickelte Code in mehreren Projekten erfolgreich eingesetzt[49, 50, 123].

B.4. Schlußfolgerung und Ausblick

Das Ziel dieser Arbeit war die Entwicklung, Implementierung und Analyse einer schnellen Simulationsmethode für Granulare Medien. Als Ausgangspunkt diente die Direct Simulation Monte Carlo Methode (DSMC). Der Grund war weniger deren Eignung für granulare Medien, sondern ihre Geschwindigkeit. Da DSMC für die Simulation dünner Gassysteme entwickelt worden ist, hat die Methode diverse Einschränkungen bei ihrer Anwendung auf Systeme hoher Dichte. Für diese Systeme existieren daher in der Li-

teratur verschiedene Verbesserungen wie CBA, CUBA oder ESMC (siehe Kapitel 2.4). Der größte Nachteil von DSMC und seinen Varianten ist der fehlende Volumenausschluß („excluded volume"). Im Gegensatz zu den anderen vorgestellten Variationen von DSMC wurde hier nicht der Kollisionsmechanismus zwischen den Teilchen modifiziert, sondern der Fortbewegungsalgorithmus. Um die höhere Geschwindigkeit von DSMC gegenüber Molekulardynamik zu erhalten, wird allerdings der Volumenausschluß nur auf einer größeren Längenskala garantiert. Bei HSMC ist dies die Zellstruktur, die auch für den Kollisionsmechanismus verwendet wird. Obwohl damit weitreichende Modifikationen am ursprünglichen DSMC Algorithmus vorgenommen worden sind, ist die HSMC Methode im Grenzfall dünner Systeme identisch zu DSMC. Die Untersuchung von HSMC durch Studien einer breiten Auswahl von granularen Systemen zeigte, dass die Methode ein breites Anwendungsspektrum besitzt. Im Vergleich zu anderen DSMC Varianten wurde der Einsatzbereich wesentlich erweitert. Weitere, denkbare Verbesserungen der Methode betreffen vor allem die Reduktion der Gitterartefakte, die bei Systemen mit hoher Dichte und Dissipation auftreten. Einzelne Möglichkeiten wurden in den Kapiteln 2.3 und 3.5 angesprochen. Ein Beispiel an dem gezeigt wurde, wie deutliche Verbesserungen erzielt werden können, ist die Wahrscheinlichkeitsverteilung des Stoßparameters $p(b)$. Ein weiteres Beispiel ist die Wahrscheinlichkeit $p(\nu)$ eines Teilchens in eine Zelle mit Volumenanteil ν einzudringen. Hier wurde bislang eine einfache Stufenfunktion verwendet. Eine weitere Möglichkeit ist die Modifikation des Gitters selbst. Wie in LGA Simulationen kann die geeignete Ausrichtung des Gitters relativ zu Vorzugsrichtungen wie der Gravitation erfolgen. Der zusätzliche Vorteil von HSMC ist, dass das Gitter nur benutzt wird, um den Volumenausschluß auf einer vergöberten Skala zu erzwingen, d.h. die Bewegung der Teilchen ist nicht auf die Eckpunkte des Gitters beschränkt. Es ist daher möglich, das Gitter zwischen den Zeitschritten zu rotieren oder zu verschieben. Im Mittel stellt dies die Isotropie und Homogenität des Raumes her, solange es keine bevorzugte Richtung oder Position gibt.

Aufgrund der hohen Geschwindigkeit bietet sich die Methode vor allem dort an, wo grosse Teilchenzahlen oder lange Simulationszeiten erforderlich sind, z.B. bei Studien zu Clustering in drei Dimensionen. Die Methode kann auch angewandt werden, um zu untersuchen, ob Korrelationen zwischen Teilchen oder sterische Effekte auf Teilchenskala notwendig sind, um gewisse Effekte widerzugeben. DSMC kann zwar ebenfalls zu diesem Zweck eingesetzt werden, die Beschränkung von DSMC auf dünne Systeme stellt allerdings für dissipative Medien eine große Einschränkung dar. Beispiele, bei denen HSMC Resultate wertvolle Einblicke für das theoretische Verständnis liefern können, sind das Skalierungsverhalten vibrierter Betten, die Form der Oberfläche eines Sandhaufens und Dichtefluktuationen beim Rohrfluß.

Lebenslauf

Persönliche Daten

Name:	Matthias Müller
geboren:	31.5.1969 in Sindelfingen
Eltern:	Manfred Müller
	Renate Müller, geb. Bellon

Schulausbildung

1975 – 1979	Justinus Kerner Grundschule Böblingen
1980 – 1988	Otto Hahn Gymnasium Böblingen
Mai 1988	Allgemeine Hochschulreife, Gesamtnote 1.3

Studium

Okt. 1989 – Jan. 1996	Physik-Studium, Universität Stuttgart
Jan. 1995 – Jan. 1996	Diplomarbeit am Institut für theoretische und angewandte Physik (Prof. Dr. H.-R. Trebin)
Jan. 1996	Diplom, mit Auszeichnung bestanden
Apr. 1996 – Mai 1999	Wissenschaftlicher Angestellter am Institut für Computeranwendungen (Prof. Dr. H. Herrmann)
seit Juni 1999	Wissenschaftlicher Angestellter am Höchstleistungsrechenzentrum Stuttgart

Eigene Veröffentlichungen

[1] H. J. Herrmann and M. Müller. How much can we simplify a system of grains? In *Granular Gases*, Lecture Notes in Physics. Springer, 1999.

[2] H. J. Herrmann and M. Müller. Simulations of granular media. In N. Attig and R. Esser, editors, *Molecular Dynamics on Parallel Computers*. World Scientific, 1999.

[3] H. J. Herrmann and M. Müller. Simulations of granular materials on different scales. *Computer Physics Communications*, 127:120–125, 2000.

[4] K. Höfler, M. Müller, and S. Schwarzer. Design and application of object oriented parallel data structures in particle and continuous systems. In E. Krause and W. Jäger, editors, *High Performance Computing in Science and Engineering*. Springer, Berlin, 1999.

[5] K. Höfler, M. Müller, S. Schwarzer, and B. Wachmann. Interacting particle-liquid systems. In E. Krause and W. Jäger, editors, *High Performance Computing in Science and Engineering*. Springer, Berlin, 1998. [Trans. HLR Stuttgart, 1998].

[6] S. Luding, M. Müller, and S. McNamara. The validity of "molecular chaos" in granular flows. In *World Congress on Particle Technology*, Davis Building, 165-189 Railway Terrace, Rugby CV21 3HQ, UK, 1998. Institution of Chemical Engineers. CD: ISBN 0-85295-401-9.

[7] M. Müller. Molecular Dynamics with C++. In F. Bassetti, K. Davis, and B. Mohr, editors, *Proceedings of the Workshop on Parallel Object Oriented Scientific Programming*. Technical Report FZJ-ZAM-IB-9906, Lisbon/Portugal, 1999.

[8] M. Müller. Abstraction benchmarks and performance of C++ applications. In *The Fourth International Conference on Supercomputing in Nuclear Applications*, TOKYO/JAPAN, 2000.

[9] M. Müller and H. J. Herrmann. DSMC - a stochastic algorithm for granular matter. In H. J. Herrmann, J.-P. Hovi, and S. Luding, editors, *Physics of dry granular media*, NATO ASI Series, Dordrecht, 1998. Kluwer Academic Publisher.

[10] M. Müller, U. Lang, and M. Resch. Libraries, applications and visualisation in metacomputing. In *Proceedings of ISThmus 2000*, pages 373–380, Poznan/Poland, April 2000.

[11] M. Müller, S. Luding, and H. J. Herrmann. Simulations of vibrated granular media in 2d and 3d. In D. E. Wolf and P. Grassberger, editors, *Friction, Arching and Contact Dynamics*. World Scientific, Singapore, 1997.

[12] M. Müller and M. Resch. PE mapping and the congestion problem on the T3E. In H. Lederer and F. Hertweck, editors, *Proceedings of the Fourth European Cray-SGI MPP Workshop*, pages 20–28. IPP, Garching, Germany, September 1998. see http://www.rzg.mpg.de/mpp-workshop/proceedings.html.

[13] S. Pickles, J. Brooke, F. Costen, E. Gabriel, M. Müller, M. Resch, and S. Ord. Metacomputing across intercontinental networks. To appear in Future Generation Computer Systems, 2001.

[14] S. Pickles, F. Costen, J. Brooke, E. Gabriel, M. Müller, M. Resch, and S. Ord. The problems and the solutions of the metacomputing experiments in SC'99. In *Proceedings of HPCN 2000*, May 2000.