Temperature Measurements in Optical Tweezer Experiments

Mathias Höld, BSc.

2016

1 Introduction

2 Motivation

3 The experiment

The starting point of this thesis is an experiment conducted by Gieseler et al [1]. It is an optical tweezer experiment, where the motion of a glass nanoparticle in a laser trap was used to investigate the fluctuation theorem[2].

3.1 Experimental setup

In the experiment, a silica nano particle with a radius of about 75 nm and mass of about 3×10^{-18}kg is trapped in a laser beam within a vacuum chamber. The trapping of the silica nano particle is achieved by a gradient force of the laser beam acting on the particle. The experimental setup is depicted in fig. 1.
The particle fluctuates within the trap in all three spatial directions. These fluctuations can be approximated such that they are decoupled, which means that they can be described by a one-dimensional Langevin equation:

$$\ddot{x} + \Gamma_0 \dot{x} + \Omega_0^2 x = \frac{1}{m} \left(F_{\text{fluct}} + F_{\text{ext}} \right) \tag{1}$$

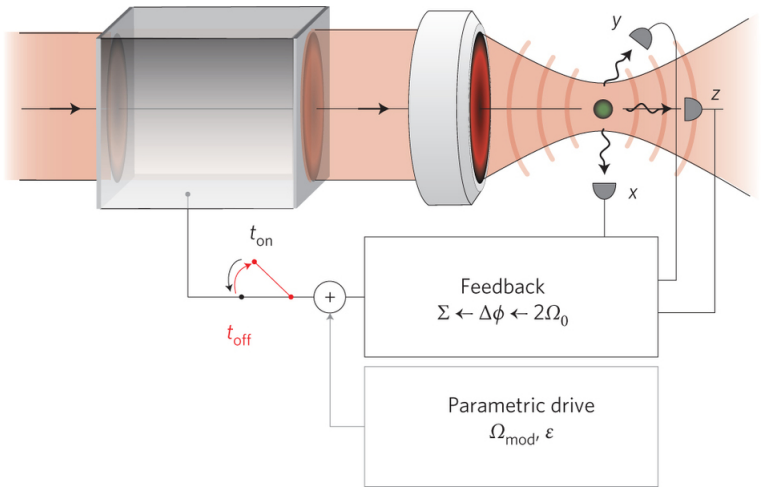


Figure 1: Experimental setup of the optical tweezer experiment. A silica nano particle is trapped in a laser beam via gradient force in a vacuum. The feedback is used to cool down the particle and create a non-equilibrium steady state. In the first part of the experiment, the feedback is turned off and the motion of the particle towards an equilibrated state is observed. In the second part of the experiment, the steady state of the particle is modified by a parametric drive. Both the parametric drive and the feedback are turned off and – as in the first part – the motion of the particle towards an equilibrated state is observerd. Figure taken from [1].

On the left hand side we have the position x (and its derivatives $\dot{x}$, the velocity and $\ddot{x}$, the acceleration), the friction coefficient Γ_0 and the angular frequency Ω_0 that describes the fluctuation along the chosen axis. On the right hand side, there are two forces. The first one is F_{fluct}, which describes a stochastic force caused by interactions with the gas in the vacuum chamber. This force is given by

$$F_{\text{fluct}} = \sqrt{2m\Gamma_0 k_B T_0}\ \xi\left(t\right) \tag{2}$$

where T_0 is the temperature of the heat bath (i.e., the surrounding gas in the vacuum chamber), k_B is the Boltzmann constant and $\xi(t)$ is white noise, which obeys the equations $\langle\xi(t)\rangle = 0$ and $\langle\xi(t)\xi(t')\rangle = \delta(t-t')$, which means that it is a random force. The term Γ_0 appears in the formula (2) due to the fluctuation-dissipation theorem, which links the damping rate to the stochastic force.

4 Simulation

The problem at hand can be studied on an atomic level with the use of computer simulation. There is a variety of methods for computer simulations that are widely used, one of which being molecular dynamics (MD) simulations.
The goal is to simulate the experiment depicted in Fig. 1 as accurately as possible. To achieve this, the setup has to be broken down into individual pieced that can be modeled by using existing methods. The nano particle will be approximated by a cube of particles, sitting on an FCC lattice that interact via a Lennard-Jones potential. The laser will be broken down into its two main purposes, as it acts as a trapping device as well as a heat source for the nano particle. Surrounding the nano particle will be a gas chamber, that acts as a thermostat in the simulation, that will be modeled as a box surrounding the nano particle. This box is filled with gas particles that interact with the nano particle via a soft-sphere potential and do not interact with one another.
The following section will give a brief overview of the concepts used to simulate every part of the experiment.

4.1 Molecular Dynamics

Molecular dynamics [3] simulations is a technique for simulating, as the name suggests, the dynamics of a classical many-body system. In this case, classical means, that the trajectories of the individual particles are calculated using classical mechanics rather then quantum mechanics. For relatively big atoms/molecules this is a very good approximation, whereas for systems consisting of hydrogen or helium the effects of quantum mechanics cannot be neglected and other methods have to be used.
The dynamics of the system are obtained by solving Newton's equations of motion for every particle.

4.2 The Nano Particle

The glass particle from the experiment will be modeled as a system of particles interacting via a Lennard-Jones pair potential,

$$U(r) = 4\varepsilon \left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6} \right] \tag{3}$$

where ε is the depth of the potential well (and thus its unit is energy) and σ is the distance at which the potential is zero. The form of the potential and the relation to the parameters is depicted in Fig. 2. Since ε and σ are crucial parameters for the simulation and do not change over time, it is practical to use them to define

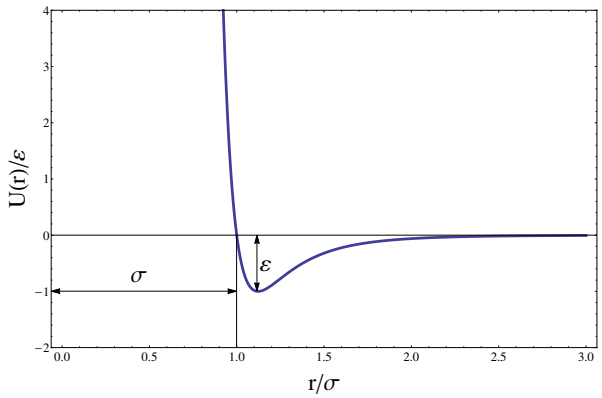


Figure 2: The Lennard-Jones 12-6 potential from (3). The x-axis is the particle distance divided by σ and the y-axis is the potential divided by the depth of the potential well.

the dimensions of the system. This means that the unit of distance is σ, the unit of energy is ε and the unit of mass is the mass of the simulated particle. The so called *reduced units* can be constructed from these three parameters and put into relation to the original units. Here are some examples:

- distance: $r^* = r/\sigma$
- potential energy: $U^* = U/\varepsilon$
- temperature: $T^* = k_B T/\varepsilon$
- time: $t^* = t\sqrt{\varepsilon/(m\sigma^2)}$
- pressure: $P^* = P\sigma^3/\varepsilon$
- density: $\rho^* = \rho\sigma^3$

One very popular choice for the simulated atoms is Argon because it is an inert gas and the atoms behave approximately like hard spheres which attract each other with weak van der Waals forces, which justifies the use of the Lennard-Jones potential. Argon has a mass of m $= 6.69 \times 10^{-26}$ kg, $\sigma = 3.4 \times 10^{-10}$m and $\varepsilon = 1.65 \times 10^{-21}$J.
With the above introduced reduced units, the Lennard-Jones potential can be written as

$$U(r^*) = 4\left[r^{*-12} - r^{*-6}\right]. \tag{4}$$

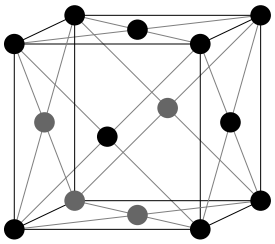
Since the reduced units will be used throughout the rest of this thesis, I will drop the asterisk henceforth.

From the Lennard-Jones potential the corresponding force can be calculated by taking the derivative with respect to the direction of interest:

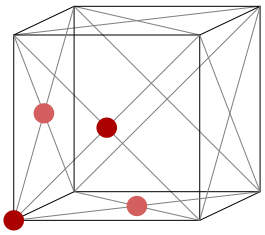
$$\begin{aligned} F_x &= -\frac{\partial}{\partial x} U(r) \\ &= -\frac{\partial}{\partial x} 4\left[r^{-12} - r^{-6}\right] \\ &= -4\left[(-12)r^{-13} - (-6)r^{-7}\right]\frac{\partial r}{\partial x} \\ &= 48\left[r^{-13} - 0.5\ r^{-7}\right]\frac{x}{r} \\ &= 48\left[r^{-14} - 0.5\ r^{-8}\right]x \end{aligned} \tag{5}$$

The force in the y and z direction can be calculated analogously.
The initial configuration of the particles is a face centered cubic (FCC) lattice. A schematic of the FCC lattice is depicted in fig. 3a. With the choice of FCC as initial configuration, there are optimal numbers for the numbers of the particles in the system. Since once FCC cell shares its atoms with its next neighbours, the number of atom per unit cell is 4 – 1/8 of a particle on eight corners and 1/2 of a particle on six faces. The whole system of atoms is then created by repeating this cell structure. One convenient way is to arrange the unit cells in a cubic system, so if there are M FCC unit cells on one edge, the whole system consists of M^3 cells. Since there are 4 particles per cell, there are ideal or so called *magic numbers* for atoms for which this setup works perfectly: $N = 4M^3 = 4, 32, 108, 256, 500, 864, \ldots$.

There are several ways to achieve this initial configuration and the one used in



(a) Schematic figure of a face centered cubic (FCC) lattice



(b) Setup for successively building a FCC lattice

this thesis [4] was to create a kind of unit cell consisting of four atoms, as shown

in fig. 3b, which can be described by a set of points

$$\begin{aligned} p_1 &= \{0, 0, 0\} \\ p_2 &= \{0.5 * a, 0.5 * a, 0\} \\ p_3 &= \{0.5 * a, 0, 0.5 * a\} \\ p_4 &= \{0, 0.5 * a, 0.5 * a\} \end{aligned}$$

From the particle number N and the number of FCC unit cells per edge M the *lattice constant* a can be calculated

$$a = \frac{L}{M} \tag{6}$$

where L is the side length of the cube that is the whole system and it is calculated via the density of the system

$$L = \sqrt[3]{\frac{N}{\rho}} \tag{7}$$

With the lattice constant and the 4 points of the FCC cell, all the particles can be put into place.

4.3 The Velocity-Verlet Algorithm

When we look at the system from a microscopic standpoint, we see that it follows some kind of path in the phase space as time progresses. Every point in this space corresponds to a set of positions and momenta and the connection between two points corresponds to the evolution of the system from one state to another. As mentioned above, this evolution (the dynamics of the system) is a crucial element to molecular dynamics. Since the equations of motion cannot be solved analytically in general, we need to approximate the solution.
The method used here is called finite difference approach. The trajectory of the system in the phase space is cut into finite pieces of length Δt and the equations of motion are solved for every segment separately (see fig. 4).
There are several ways to solve this kind of problem, but since we are interested in implementing it into a computer program the ideal solution should have some basic properties [5]:

- It should be fast and require little memory
- It should permit the use of a large time step Δt
- It should duplicate the classical trajectory as closely as possible

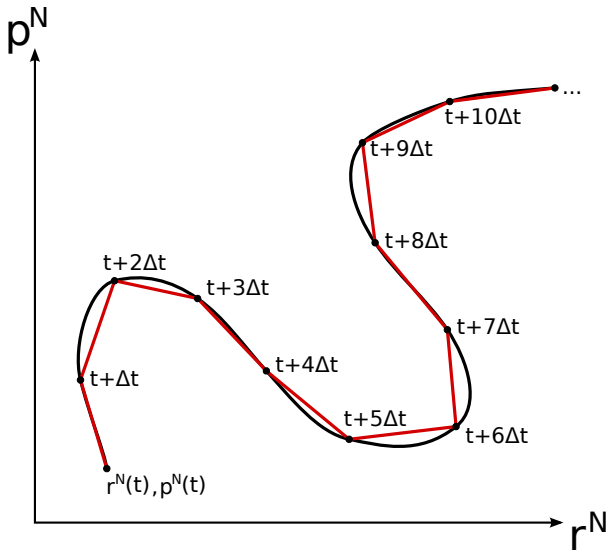


Figure 4: Simplified graphical schematic of the finite difference approach. The evolution of the system from a point $(r^N(t), p^N(t))$ in the phase space is approximated by slicing it up into pieces of length Δt. On every stop after the starting point $(t + \Delta t,\ t + 2\Delta t,\ t + 3\Delta t, \ldots)$ the equations of motion can be solved numerically.

- It should satisfy known conservation laws
- It should be simple and easy to program

One algorithm that has all of the above mentioned features is the one proposed by Verlet [6]. In his paper, Verlet starts by Taylor expanding the coordinate vector $\mathbf{r}_i$ for one particle after one time step Δt:

$$\mathbf{r}_i(t + \Delta t) = \mathbf{r}_i(t) + \dot{\mathbf{r}}_i \Delta t + \frac{1}{2}\ddot{\mathbf{r}}_i \Delta t^2 + \frac{1}{3!}\dddot{\mathbf{r}}_i \Delta t^3 + \mathcal{O}(\Delta t^4) \tag{8}$$

Since the first derivative of the coordinate is the velocity, $\dot{\mathbf{r}}_i(t) = \mathbf{v}_i(t)$, and the second derivative of the coordinate is the acceleration, $\ddot{\mathbf{r}}_i(t) = \mathbf{a}_i(t)$, using Newton's law $\mathbf{F}_i(t) = m_i \mathbf{a}_i(t)$ equation (8) can be written as:

$$\mathbf{r}_i(t + \Delta t) = \mathbf{r}_i(t) + \mathbf{v}_i(t)\Delta t + \frac{1}{2m}\mathbf{F}_i(t)\Delta t^2 + \frac{1}{3!}\dddot{\mathbf{r}}_i(t)\Delta t^3 + \mathcal{O}(\Delta t^4) \tag{9}$$

The same calculation can be carried out for one time step before t:

$$\mathbf{r}_i(t - \Delta t) = \mathbf{r}_i(t) - \mathbf{v}_i(t)\Delta t + \frac{1}{2m}\mathbf{F}_i(t)\Delta t^2 - \frac{1}{3!}\dddot{\mathbf{r}}_i(t)\Delta t^3 + \mathcal{O}(\Delta t^4) \tag{10}$$

The sum of those two equations yields

$$\mathbf{r}_i(t + \Delta t) + \mathbf{r}_i(t - \Delta t) = 2\mathbf{r}_i(t) + \frac{1}{2}\mathbf{F}_i(t)\Delta t^2 + \mathcal{O}(\Delta t^4) \tag{11}$$

and from this we get the final form for the new coordinates:

$$\mathbf{r}_i(t+\Delta t) = 2\mathbf{r}_i(t) + \mathbf{r}_i(t-\Delta t) + \frac{1}{2}\mathbf{F}_i(t)\Delta t^2 \tag{12}$$

This means that the new coordinates can be calculated using the current coordinates, the current forces and the coordinates from the past time step.
From equation (12) one thing becomes clear: the velocities are not necessary to calculate the new positions and they are not computed in the process, but for the calculation of i.e. the kinetic energy the velocities are needed. Thus, the velocities have to be calculated with a combination of the Taylor expansions of the coordinate vectors at $t+\Delta t$ and $t-\Delta t$:

$$\mathbf{r}_i(t+\Delta t) - \mathbf{r}_i(t-\Delta t) = 2\mathbf{v}_i(t)\Delta t + \mathcal{O}(\Delta t^3) \tag{13}$$

which can be rewritten as

$$\mathbf{v}_i(t) = \frac{\mathbf{r}_i(t+\Delta t) - \mathbf{r}_i(t-\Delta t)}{2\Delta t} + \mathcal{O}(\Delta t^2) \tag{14}$$

This approach has the advantage that it is fast, requires little memory and is reliable in the sense that there is no energy drift occuring during the simulation, which means that the energy is conserved. When we compare this to the list of desired properties this seems like a good algorithm.
This algorithm however has two significant disatvantages. The first one is the calculation of the velocities. As can be seen in equation (14), the accuracy of the calculation is only $\mathcal{O}(\Delta t^2)$ while the positions can be calculated with an error of order Δt^4. The other big disatvantage is the first step of the algorithm. Since the calculation of the new positions requires the current positions and the ones from one time step before, which technically do not exist.
The solutions to this problem is to include the stepwise calculation of the velocities [7]. For this we start again with the Taylor expansion of the coordinates mathbftor (9) and instead of Taylor expanding $\mathbf{r}_i(t-\Delta t)$, we write it as

$$\mathbf{r}_i(t) = \mathbf{r}_i(t+\Delta t) - \mathbf{v}_i(t+\Delta t)\Delta t + \frac{1}{2m}\mathbf{F}_i(t+\Delta t)\Delta t^2 \tag{15}$$

Using (9) in the abvove equation we get

$$\begin{aligned}\mathbf{r}_i(t) = \mathbf{r}_i(t) + \mathbf{v}_i(t)\Delta t + \frac{1}{2m}\mathbf{F}_i(t)\Delta t^2 \\ - \mathbf{v}_i(t+\Delta t)\Delta t + \frac{1}{2m}\mathbf{F}_i(t+\Delta t)\Delta t^2\end{aligned} \tag{16}$$

and thus

$$\mathbf{v}_i(t+\Delta t) = \mathbf{v}_i(t) + \frac{1}{2m}\left(\mathbf{F}_i(t) + \mathbf{F}_i(t+\Delta t)\right)\Delta t \tag{17}$$

With the addition of this equation this algorithm is called the Velocity-Verlet algorithm, which is summarized in the following two equations:

$$\begin{aligned} \mathbf{r}_i(t+\Delta t) &= \mathbf{r}_i(t) + \mathbf{v}_i(t)\Delta t + \frac{1}{2m}\mathbf{F}_i(t)\Delta t^2 \\ \mathbf{v}_i(t+\Delta t) &= \mathbf{v}_i(t) + \frac{1}{2m}\Big[\mathbf{F}_i(t) + \mathbf{F}_i(t+\Delta t)\Big]\Delta t \end{aligned} \tag{18}$$

This algorithm is self starting, uses a small amount of memory, conserves the energy (not exactly) and gives a very good approximation to the originial trajectory in the phase space. To program this algorithm the following steps are needed:

1. Calculate all the forces between the particles (for the first step only)
2. Calculate the new positions with current velocity and forces
3. Calculate first half of the new velocities with current forces
4. Calculate all the forces for the new position
5. Use new forces to calculate second half of new velocities

The first point only has to be carried out for the first step, because the forces have not been calculated at this point. As the forces are calculated for the new positions in step 4, they can be used for the next time step. For the first time step the velocities have to be chosen randomly and are calculated with this algorithm from that point forward.

4.4 The Laser Beam - Energy Influx

The nano particle is trapped in the laser beam. While the motion of the center of mass is localized, the individual atoms that make up the glass sphere absorb the energy from the laser which increases their velocity.
To simulate this kind of behaviour, thermostat algorithms [8] such as the Nosé-Hoover [9, 10] or the Andersen [11] algorithms are often used in simulation to change the temperature of the system in a controllable way.
The problem with these kind of algorithms is, that a target temperature has to be fixed which will be reached at some point. In order to simulate the influx of energy from the laser it would be better to have an algorithm that supplies the system with a certain amount of energy continuously. Fortunately, such an algorithm exists.
The algorithm is called Heat Exchange Algorithm (HEX) [12]. Its intended purpose is the use in non-equilibrium molecular dynamics (NEMD) to study transport

phenomena and determine transport coefficients. The algorithm works by introducing two regions in the system, one serving as a heat source and the other as a heat sink. A specific amount of heat is then exchanged between those two reservoirs. As it turns out however, this algorithm introduces an energy shift for longer simulation times. This led Wirnsberger et al. [13] to revisit the algorithm and identify the cause of this energy drift, to create a more suitable algorithm, which they called eHEX.
As in the HEX algorithm, regions are introduced to the system which act either as heat sinks or heat sources. These are labelled with Γ_k, where $k > 0$, and have corresponding amount of exchanged heat ΔQ_{Γ_k}. If ΔQ_{Γ_k} is negative, heat is subtracted from the system and vice versa. Regions that neither act as heat source or heat sink are labelled with Γ_0, which are also called Hamiltonian regions (see fig. 5). The centers of mass of the particles in simulation box, denoted by Ω, and the regions Γ_k are assumed to be moving with velocities v_Ω and v_{Γ_k} respectively.
The change of the energy in a region Γ_k is achieved by rescaling the velocities by a factor ξ_k and shifted by the velocity of the corresponding region:

$$\mathbf{v}_i \to \bar{\mathbf{v}}_i = \xi_k \mathbf{v}_i + (1 - \xi_k)\mathbf{v}_{\Gamma_k} \tag{19}$$

The bar over a quantity denotes the value after the exchange of heat.
The factor ξ_k is given by

$$\xi_k = \sqrt{1 + \frac{\Delta Q_{\Gamma_k}}{\mathcal{K}_{\Gamma_k}}} \tag{20}$$

where ΔQ_{Γ_k} is the exchanged heat in the region Γ_k and $\mathcal{K}_{\Gamma_k}$ is the non-translational kinetic energy of the region Γ_k and is given by

$$\mathcal{K}_{\Gamma_k} = \sum_{i \in \gamma_k} \frac{m_i v_i^2}{2} - \frac{m_{\Gamma_k} v_{\Gamma_k}^2}{2} \tag{21}$$

The sum is taken over all indices in γ_k which is the set of indices of particles in the region Γ_k.
For the final version of the eHEX algorithm there are three more quantities needed. The first one is the heat flux per time step, denoted by $\mathcal{F}_{\Gamma_k}$:

$$\mathcal{F}_{\Gamma_k} = \frac{\Delta Q_{\Gamma_k}}{\Delta t} \tag{22}$$

The second one is the thermostatting force $\boldsymbol{\eta}_i$, which is defined as

$$\boldsymbol{\eta}_i = \begin{cases} m_i \, \dfrac{\mathcal{F}_{\Gamma_{k(\mathbf{r}_i)}}}{2\mathcal{K}_{\Gamma_{k(\mathbf{r}_i)}}} \left(\mathbf{v}_i - \mathbf{v}_{\Gamma_{k(\mathbf{r}_i)}}\right) & \text{if } k(\mathbf{r}_i) > 0 \\ 0 & \text{otherwise} \end{cases} \tag{23}$$

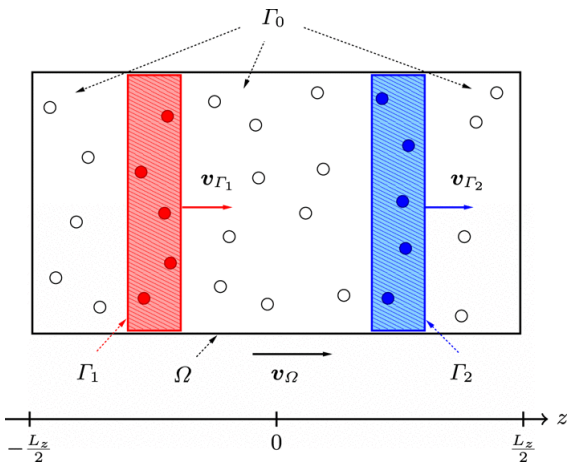


Figure 5: Setup for the use of the HEX and eHEX algorithms. The simulationbox Ω, which is moving with a velocity of v_Ω, contains a heat source (red), Γ_1, which is moving with velocity v_{Γ_1} and a heat sink (blue), Γ_2, which is moving with velocity v_{Γ_2}. The regions that are neither heat sources or heat sinks are denoted by Γ_0.

where $k(\mathbf{r}_i)$ is the index of the region in which particle i located, i.e. $k(\mathbf{r}_i) = 0$ means that the particle is in a Hamiltonian region and $k(\mathbf{r}_i) > 0$ denotes the heat sinks and sources.
The last quantity is the one that corrects the long term energy drift of the HEX algorithm, denoted by $\mathcal{E}r_{i,\alpha}$. The analysis and derivation of this term is given in [13].

$$\begin{aligned}\mathcal{E}r_{i,\alpha} = \frac{\eta_{i,\alpha}}{m_i \mathcal{K}_{\Gamma_{k(\mathbf{r}_i)}}} \left[\frac{\mathcal{F}_{\Gamma_{k(\mathbf{r}_i)}}}{48} + \frac{1}{6} \sum_{j \in \gamma_{k(\mathbf{r}_i)}} \mathbf{f}_j \cdot \left(\mathbf{v}_j - \mathbf{v}_{\Gamma_{k(\mathbf{r}_i)}} \right) \right] \\ - \frac{\mathcal{F}_{\Gamma_{k(\mathbf{r}_i)}}}{12 \mathcal{K}_{\Gamma_{k(\mathbf{r}_i)}}} \left[\frac{f_{i,\alpha}}{m_i} - \frac{1}{m_{\Gamma_{k(\mathbf{r}_i)}}} \sum_{j \in \gamma_{k(\mathbf{r}_i)}} f_{j,\alpha} \right]\end{aligned} \tag{24}$$

The $\mathbf{f}$ in the above equation denotes the force corresponding to the chosen intermolecular potential $U(\mathbf{r})$,

$$\mathbf{f} = -\nabla_{\mathbf{r}_i} U(\mathbf{r}_i) \tag{25}$$

With all the necessary quantities established, the updating sequence of the eHEX algorithm can be written down:

$$\bar{\mathbf{v}}_i^n = \xi^n_{k(\mathbf{r}_i)}\mathbf{v}_i^n + \left(1-\xi^n_{k(\mathbf{r}_i)}\right)\mathbf{v}^n_{\Gamma_{k(\mathbf{r}_i)}} \tag{26a}$$

$$\bar{\mathbf{v}}_i^{n+\frac{1}{2}} = \bar{\mathbf{v}}_i^n + \frac{\Delta t}{2m_i}\mathbf{f}_i^n \tag{26b}$$

$$\bar{\mathbf{r}}_i^{n+1} = \mathbf{r}_i^n + \Delta t\bar{\mathbf{v}}_i^{n+\frac{1}{2}} \tag{26c}$$

$$\mathbf{f}_i^{n+1} = -\nabla_{\mathbf{r}_i}U(\mathbf{r})|_{\mathbf{r}=\bar{\mathbf{r}}^{n+1}} \tag{26d}$$

$$\bar{\mathbf{v}}_i^{n+1} = \bar{\mathbf{v}}_i^{n+\frac{1}{2}} + \frac{\Delta t}{2m_i}\mathbf{f}_i^{n+1} \tag{26e}$$

$$\mathbf{v}_i^{n+1} = \bar{\xi}^{n+1}_{k(\bar{\mathbf{r}}_i)}\bar{\mathbf{v}}_i^{n+1} + \left(1-\bar{\xi}^{n+1}_{k(\bar{\mathbf{r}}_i)}\right)\bar{\mathbf{v}}^{n+1}_{\Gamma_{k(\bar{\mathbf{r}}_i)}} \tag{26f}$$

$$\mathbf{r}_i^{n+1} = \bar{\mathbf{r}}^{n+1} - \Delta t^3\mathcal{E}\bar{\mathbf{r}}_i^{n+1} \tag{26g}$$

This algorithm can be adapted for the problem of the levitating nano sphere in the laser beam by adjusting the setup and some of the parameters.
Firstly, the setup of the heat sinks and sources has to be changed. Since there is only energy pumped into the system from the outside, the region acting as a heat sink vanishes and the region acting as a heat source spans over the whole simulation box. This means that, using the notation of fig. 5, $\Omega = \Gamma_1$. Furthermore, neither center of mass of the particles the simulation box nor the heat source are moving, i.e. $\mathbf{v}_\Omega = \mathbf{v}_{\Gamma_1} = 0$. This affects the non-translational kinetic energy term $\mathcal{K}_{\Gamma_1}$, the thermostatting force $\boldsymbol{\eta}$ and the correction term $\mathcal{E}\mathbf{r}$. Since there is only one region acting as a heat source, the terms ΔQ, $\mathcal{K}$ and $\mathcal{F}$ don't need an index. The summation index in (21) and (24) can be changed to the number of particles in the system, N, since all the particles are within the heat exchanging region. The masses are set to 1, i.e. $m_i = 1$ with the chosen reduced units and with this the total mass of the region (in the term $1/m_{\Gamma_{k(\mathbf{r}_i)}}$ in (24)) is equal to the number of particles in the system. With these changes, the quantities can be written down

as:

$$\mathcal{K} = \sum_N \frac{v_i^2}{2} \tag{27}$$

$$\xi = \sqrt{1 + \frac{\Delta Q}{\mathcal{K}}} \tag{28}$$

$$\mathcal{F} = \frac{\Delta Q}{\Delta t} \tag{29}$$

$$\boldsymbol{\eta}_i = \frac{\mathcal{F}}{2\mathcal{K}}\, \mathbf{v}_i \tag{30}$$

$$\begin{aligned} \mathcal{E} r_{i,\alpha} &= \frac{\eta_{i,\alpha}}{\mathcal{K}} \left[\frac{\mathcal{F}}{48} + \frac{1}{6} \sum_N \mathbf{f}_j \cdot \mathbf{v}_j \right] \\ &- \frac{\mathcal{F}}{12\mathcal{K}} \left[f_{i,\alpha} - \frac{1}{N} \sum_N f_{j,\alpha} \right] \end{aligned} \tag{31}$$

This means that the laser is modeled to be pumping energy into the system, which increases the velocities of the particles in the system over time, while the center of mass motion is not affected by this.

4.5 The Laser Beam - Trapping

As mentioned above, the glass particle is trapped in the laser beam and the position of the particle is localized. This behaviour has to be modeled as well.
The approximation of the original paper [1], where the movements in the three spatial directions are decoupled will be used in the model as well. As to the model of the trapping force itself, the most straightforward approach is to use a harmonic oscillator potential. The force and the corresponding potential can be written as

$$\mathbf{F} = -k\Big[\mathbf{x} - \mathbf{x}_0\Big] \tag{32}$$

$$U = \frac{1}{2} k \Big[\mathbf{x} - \mathbf{x}_0\Big]^2 \tag{33}$$

where $\mathbf{x}_0$ is the position of minimal potential energy. Since the trap is acting on the whole system, the force is acting on the center of mass. To calculate the center of mass positions and velocities, the positions and velocities of all particles have to be summed up and divided by the particle number:

$$\mathbf{r}_{\mathrm{COM}} = \frac{1}{N} \sum_{i=1}^{N} \mathbf{r}_i \tag{34}$$

4.6 Surrounding Gas - Thermostat

Without any equilibrating mechanism the setup described by now would lead to the system heating up indefinetly, which is not desirable. Furthermore, the goal is to recreate the experiment as close as possible. Since the nano particle is trapped in a laser beam within a vacuum chamber, the surrounding gas has to be modeled as well. In the next step we will introduce the surrounding gas of the gas chamber that will absorb some of the energy in the system, leading to the final state.
Generally, pressure is introduced to the system by surrounding the object of interest (in this case the glass nano sphere) with a thermostatting pressure medium. There are two main requirements for the choice of such a pressure medium: the exerted pressure must be hydrostatic and the computation of the interaction between the pressure medium and the object of interest must not take up a lot of resources.
The model used in this thesis was developed by Grünwald and Dellago [14] and uses an ideal gas of non-interacting particles as thermostatting pressure medium. The particles of this pressure medium flow into the simulation from an outside volume, whose geometry is based on the form of the object of interest (this will be referred to as the minimal volume of cells) and leave the simulation box if their position reaches the boundary of this minimal volume of cells. This behaviour is very close to the real experiment which makes this thermostat an ideal candidate for the simulation of the experiment. The gas particles interact with the object via a soft-sphere potential of the form

$$U(r) = \varepsilon \left(\frac{\sigma}{r}\right)^{12} \tag{35}$$

where ε is the interaction strength and σ is the interaction range and r is the distance between the gas particle and the interacting particle. The gas particles do not interact with one another in this model.
In order to increase the efficiency of the computing process, σ should be chosen carefully. For larger σ, the number of interaction partners increases, which increases the number of force calculations which are a very time consuming part. For smaller σ the possibility for gas particles reaching the inside of the nano particle increases, which is not desirable. So σ should be chosen small enough to keep the force calculations at a minimum and large enough for the particle to stay on the outside of the crystal. In the computer simulations performed for this thesis, the interaction length is chosen to be $\sigma = 1$ with a cut-off radius of $r_c = 2.5$.
The algorithm can be performed by following these steps:

1. Randomly draw the number of particles that are created on a single side of

the minimal volume of cells, N_{fac}, from the distribution

$$\langle N_{\text{fac}} \rangle = \Delta t L^2 P \sqrt{\frac{1}{2\pi m k_B T}} \tag{36}$$

where Δt is the time step of the simulation, L is the side length of the cell in which the particle is created, P is the desired pressure, m is the mass of the gas particle, k_B is the Boltzmann constant (which will be set to 1 in reduced units) and T is the desired temperature. This chosen number of particles is then equally distributed over the face of the cell on which they are created. The velocities of the created particles are drawn from two different random number distributions. The component of the particle perpendicular to the surface is drawn from a Rayleigh distribution of the form

$$p(v_i) = \frac{m}{k_B T} v_i \; e^{-\frac{m v_i^2}{2 k_B T}} \tag{37}$$

The other components of the particles' velocity are drawn from a Maxwell-Boltzmann distribution.

2. Perform the first step of the velocity Verlet algorithm to propagate the particle positions by one time step.

3. Check if any gas particles have left the minimal volume of cells and remove those which have.

4. Check if the geometry of the crystal and with it the minimal volume of cells has changed. If it has, remove all gas particles in the cells that are no longer needed. Then insert new gas particles to the created cells with a number drawn from a Poisson distribution with mean value

$$\langle N_{\text{ins}} \rangle = \frac{P L^3}{k_B T} \tag{38}$$

If the length of the cell L is chosen to be equal to the cut-off radius r_c, this insertion should only be carried out with a probablity of

$$P_{\text{ins}} = e^{-\frac{U}{k_B T}} \tag{39}$$

(where U is the interaction energy between the gas particle and the crystal) because it is possible that the inserted particle is within the interaction range of the crystal.

5. Compute all forces.

6. Perform the second step of the velocity Verlet and propagate the particle velocities by one time step.

This algorithm is formulated in a general fashion, so that a wide range of crystals and geometries can be used. Although the shape of the nano particle changes over the course of the simulation, it is not necessary to change the geometry of the surrounding box. The algorithm will therefore not be carried out using cell-lists and a minimal volume of cells that is changing over the course of the simulation, but rather a fixed setup of the volume surrounding the nano particle.
Since the particle itself is modeled as a cube, a straighforward approach is surrounding it by a bigger cube. The distance between the cube face and the nano particle is chosen to be the side length of the nano particle, so the setup will properly scale for different numbers of particles. The schematic setup for this is depicted in Fig.6.

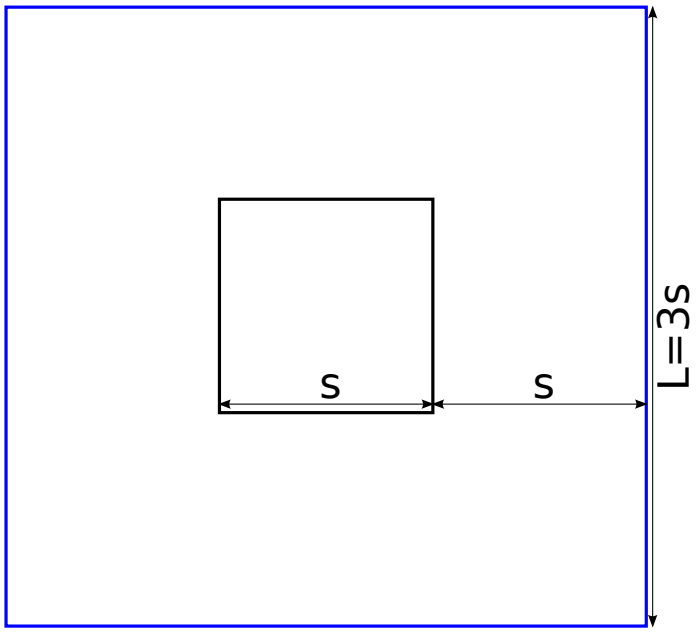


Figure 6: Schematic setup for the barostat. The outer volume (blue) is one side length s away from the nano particle (black) which means that the side length of the face of the surrounding box is $L = 3s$. The gas particles are created along the blue lines of the outer volume. Since the overall geometry of the nano particle does not change, the geometry of the outer volume will be constant throughout the simulation.

5 Results

Since the simulation consists of various parts that have to work together, we first need to check if every part itself works.

5.1 The Crystal

The first element of the simulation is the setup of the FCC lattice on which the particles are placed on. To check if this is done correctly, we can use a visualisation and rendering software such as VMD[15] or OVITO[16]. The rendered image of the nano particle can be seen in fig. 7. The chosen perspective shows the structure of the FCC lattice, where 3 layers are forming where the atoms take the position on the gap between the atoms from the layer below.

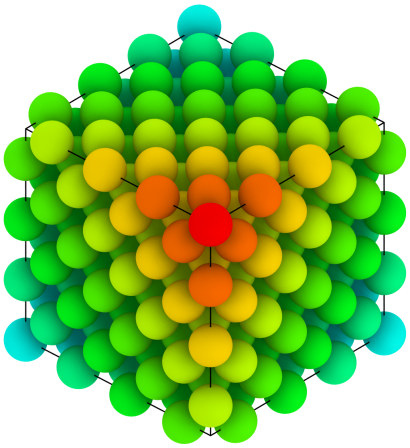


Figure 7: A rendered image of the FCC lattice. The chosen color-coding along the (1,1,1) Miller-index highlights the layers for this perspective. The perspective is chosen to see the layering of the FCC lattice with its 3 layer structure. The black box acts as a reference point to outline the geometry of the system.

5.2 Velocity Verlet

The velocity Verlet part of the simulation is taking place in a NVE ensemble from a thermodynamic point of view. This means that the particle number N, the volume V and the total energy E are constant. In contrast to the NVT ensemble used in Monte-Carlo simulations this means, that the velocities of the particle can not be chosen directly. The initial velocities have to be drawn from a random

number generator. This means that the velocities have to be adjusted after they have been drawn.
The first adjustment that needs to be made is making sure that the system is staying in place and not floating away. To do this, the velocity vectors of all the atoms in the system have to be added to get the resulting velocity vector for the whole system that is thought to be acting on the center of mass. This resulting vector is then divided by the number of particles in the system and the resulting vector is then subtracted from every velcocity vector of every atom in the system.

$$\mathbf{v}_{\text{tot}} = \sum_{i=1}^{N} \mathbf{v}_i \tag{40}$$

$$\mathbf{v}_{\text{cut}} = \frac{\mathbf{v}_{\text{tot}}}{N} \tag{41}$$

$$\mathbf{v}_i^{\text{new}} = \mathbf{v}_i - \mathbf{v}_{\text{cut}} \tag{42}$$

The velocities of the particle can roughly be chosen by adjusting the range of numbers the random number generator can choose from. However, the resulting velocities (and therefore the temperature of the system) are generally not precisely how they should be. If the system should have a specific starting temperature T, the velocities need to be scaled accordingly. This rescaling is done by summing the square of the velocities (this equals calculating the kinetic energy since the masses are set to 1) and rescale them by a factor λ:

$$E = \sum_{i=1}^{N} \frac{\mathbf{v}_i^2}{2} \tag{43}$$

$$\lambda = \sqrt{\frac{3(N-1)T}{2E}} \tag{44}$$

$$\mathbf{v}_i^{\text{new}} = \lambda \mathbf{v}_i \tag{45}$$

This process has to be done a couple of times, until the system equilibrates to the desired temperature. The problem with this procedure is the fact that it contradicts the constant energy property of the ensemble. If we look at the energy values in fig. 8 we can see the irregularities (jumps) before the system equilibrates to a constant value. This equilibration process divides the simulation in two phases: the equilibration phase and the measurement phase. In the equilibration phase, the system has time to reach a configuration that is (as the name suggests) in an equilibrated state. During this phase, there are usually no measurements (such as energy) performed, since the values from this phase are distorting the mean value of the simulation. That is why the configurations from the equilibration phase are discarded and the measurement phase takes place, where the values do get calculated and are used to determine the mean values.

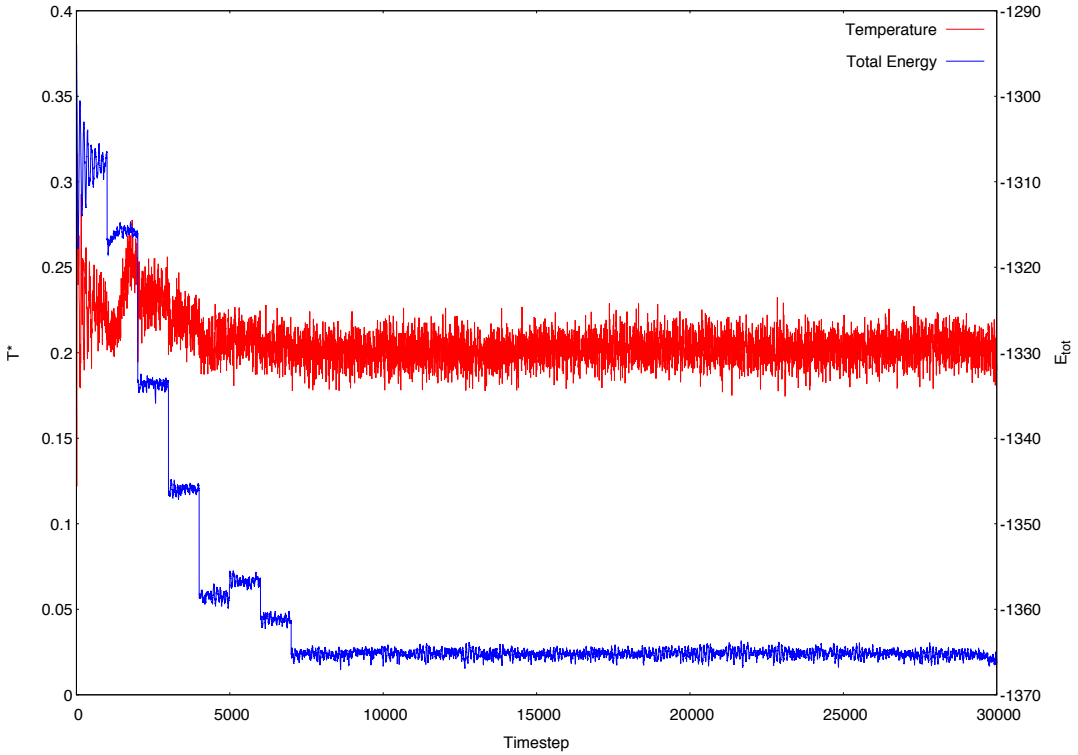


Figure 8: Temperature and energy in the equilibration phase of the velocity Verlet algorithm. The velocity rescaling (see eq. (43)-(45)) is taking place every 1000th step until the 9000th, which can be seen in the energy curve whre the jumps are happening. The desired temperature in this case is $T^* = 0.2$, which is reached after about 7000 steps.

5.3 eHEX

The next checkpoint is the eHEX algorithm. Since it contains several steps and variables that have to be calculated, this algorithm is very prone to errors. Perhaps the most crucial of those variables in the algorithm is the amount of heat injected into the system, ΔQ. The effect of the value of this variable on the system has therefore to be checked to make sure the system is not overheating.
The investigation of the effects on the system can be done in two ways: comparing the temperature and the energy of the system during the application of the algorithm (similar to the way it's been done with the velocity Verlet) and comparing the effect of different values of ΔQ.
Since the eHEX algorithm injects heat into the system, one would expect the en-

ergy and the temperature to rise during the application of the algorithm. In fig. 9 we can see that exactly this is happening. The graph shows a velocity Verlet phase (that follows an equilibration phase which is not recorded) of constant temperature and energy and an eHEX phase, where the temperature and the enrgy are constantly rising. The comparison of different values of ΔQ can be seen in fig. 10.

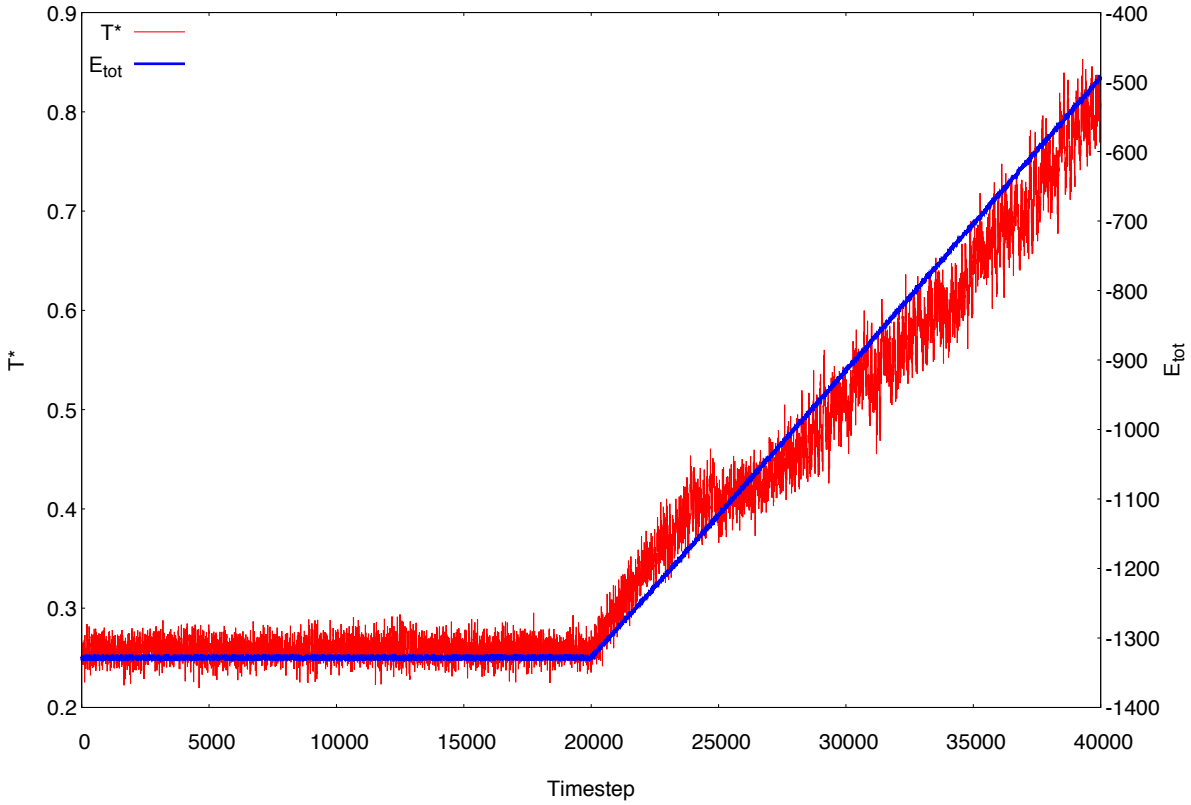


Figure 9: Comparison of temperature (red) and energy (blue) during the application of the eHEX algorithm with $\Delta Q = 0.04$ (in reduced units). The first 20000 timesteps, the energy and temperature are constant during the application of the velocity Verlet algorithm and start to rise, once the eHEX algortithm takes over. The values for the temperature can be seen on the ordinate on the left hand side and the values for the energy on the right hand side.

As the correlation between the temperature and the energy has been shown in fig. 9, the plotting of the temperature data should suffice.

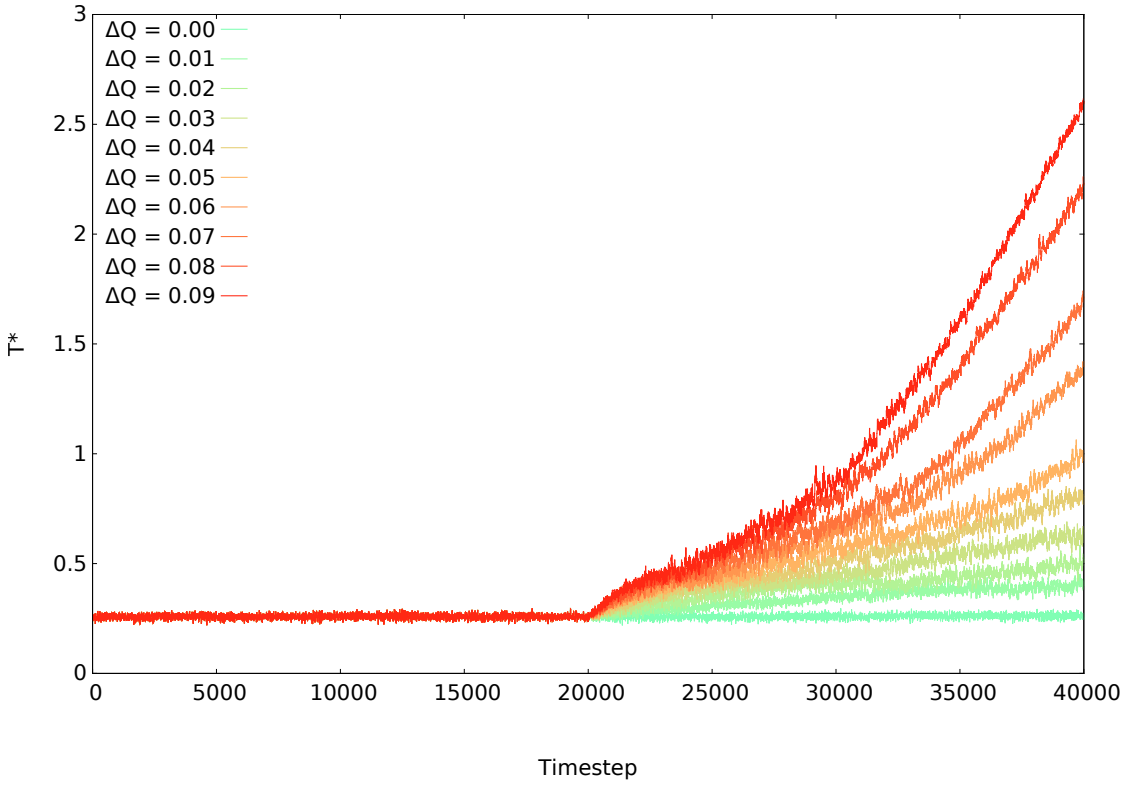


Figure 10: Comparison of different values for ΔQ, ranging from 0 to 0.09. As the value $\Delta Q = 0$ corresponds to no heat being injected into the system, it is not surprising that the temperature remains constant. The graph shows that an increase in ΔQ leads to an increase in the rate of warming up, which means that for bigger ΔQ, the system is heating up more quickly.

5.4 Thermostat

The barostat, as described previously, keeps the system from overheating. The barostat algorithm is applied at the same time as the eHEX algorithm, after the velocity Verlet has equilibrated the system.
While the eHEX algorithm is pumping energy into the nano particle while the surrounding gas particles absorb excessive energy. This leads to the development of a new, higher steady temperature of the nano particle, which can be seen in Fig. 11. As we are interested in the temperatures of the particles before and after the interaction with the nano particle, we need some way to differentiate those two states. This can be done when looking at the total energy of the gas particles.
The gas particles are created on the surface of the simulation box, which is further

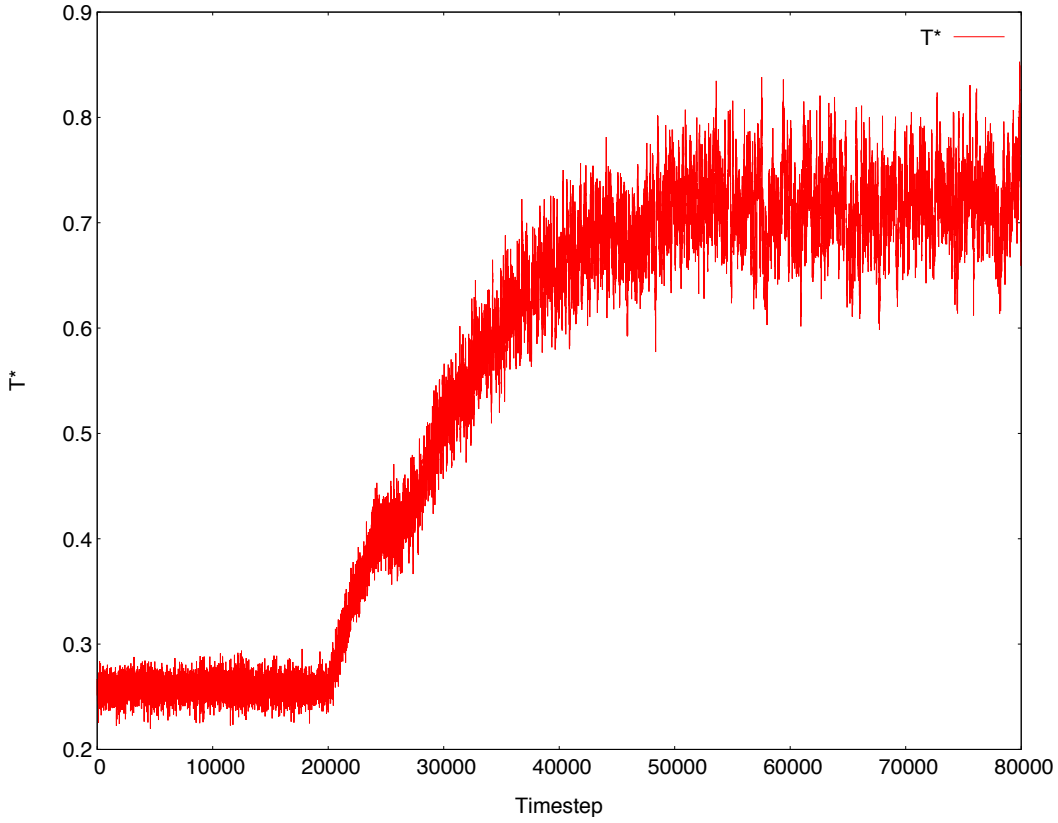


Figure 11: Temperature of the system during the application of the eHEX and the barostat algorithm. During the first 20000 steps the system is in equilibrium while the velocity Verlet algorithm is applied. After that, the eHEX and the barostat algorithms are turned on. This leads to an increase in temperature until a new steady temperature is reached.

away than the interaction range between the gas particles and the nano particle. Since the gas particles do not interact with each other, the total energy of the gas particles does not change until they reach the interaction range of the nano particle. There they undergo a change of energy due to the interaction with the nano particle and then make their way away from the nano particle. After the interaction, when they reach a point outside of the interaction range, the enrgy stays constant again until the particles reach the limits of the simulation box.
Therefore, the life span of a single gas particle can be partioned into 3 stages: incoming, interacting, outgoing. In a typical scene of interaction all three of these stages are represented by different gas particles, as can be seen in fig. 12.

Since the gas particles are interacting with the nano particle, the temperatures

and energies are values of interest. The energies are recorded using histograms and the temperature is calculated instantaneously every 100th timestep.
The calculation of the energies of the incoming particles has to be done with care. The problem are particles that enter the simulation box, but fly by the nano particle at a distance that is greater than the interaction range. Since the change of the energy is the interesting part, only the energy particles of that *do* interact with the nano particle shall be calculated. This can be achieved by counting particles that enter the stage of *interacing*. One such histogram can be seen in Fig. 13.

6 Conclusion

References

[1] Jan Gieseler, Romain Quidant, Christoph Dellago, and Lukas Novotny. Dynamic relaxation of a levitated nanoparticle from a non-equilibrium steady state. *Nat Nano*, 9(5):358–364, May 2014.

[2] Gavin E. Crooks. Entropy production fluctuation theorem and the nonequilibrium work relation for free energy differences. *Phys. Rev. E*, 60:2721–2726, Sep 1999.

[3] D. Frenkel and B. Smit. *Understanding Molecular Simulation: From Algorithms to Applications*. Computational science series. Elsevier Science, 2001.

[4] Idea taken from the Molecular Dynamics part of this lecture: `http://www.physics.buffalo.edu/phy411-506/`.

[5] M. P. Allen and D. J. Tildesley. *Computer Simulation of Liquids*. Clarendon Press, New York, NY, USA, 1989.

[6] Loup Verlet. Computer "experiments" on classical fluids. i. thermodynamical properties of lennard-jones molecules. *Phys. Rev.*, 159:98–103, Jul 1967.

[7] William C. Swope, Hans C. Andersen, Peter H. Berens, and Kent R. Wilson. A computer simulation method for the calculation of equilibrium constants for the formation of physical clusters of molecules: Application to small water clusters. *The Journal of Chemical Physics*, 76(1):637–649, 1982.

[8] Philippe H. Hünenberger. *Thermostat Algorithms for Molecular Dynamics Simulations*, pages 105–149. Springer Berlin Heidelberg, Berlin, Heidelberg, 2005.

[9] Shuichi Nosé. A unified formulation of the constant temperature molecular dynamics methods. *The Journal of Chemical Physics*, 81(1):511–519, 1984.

[10] William G. Hoover. Canonical dynamics: Equilibrium phase-space distributions. *Phys. Rev. A*, 31:1695–1697, Mar 1985.

[11] Hans C. Andersen. Molecular dynamics simulations at constant pressure and/or temperature. *The Journal of Chemical Physics*, 72(4):2384–2393, 1980.

[12] B. Hafskjold, T. Ikeshoji, and S. Kjelstrup Ratkje. On the molecular mechanism of thermal diffusion in liquids. *Molecular Physics*, 80:1389–1412, December 1993.

[13] P. Wirnsberger, D. Frenkel, and C. Dellago. An enhanced version of the heat exchange algorithm with excellent energy conservation properties. *The Journal of Chemical Physics*, 143(12), 2015.

[14] M. Grünwald and C. Dellago. Ideal gas pressure bath: a method for applying hydrostatic pressure in the computer simulation of nanoparticles. *Molecular Physics*, 104(22-24):3709–3715, 2006.

[15] William Humphrey, Andrew Dalke, and Klaus Schulten. VMD – Visual Molecular Dynamics. *Journal of Molecular Graphics*, 14:33–38, 1996.

[16] Alexander Stukowski. Visualization and analysis of atomistic simulation data with ovito–the open visualization tool. *Modelling and Simulation in Materials Science and Engineering*, 18(1):015012, 2010.

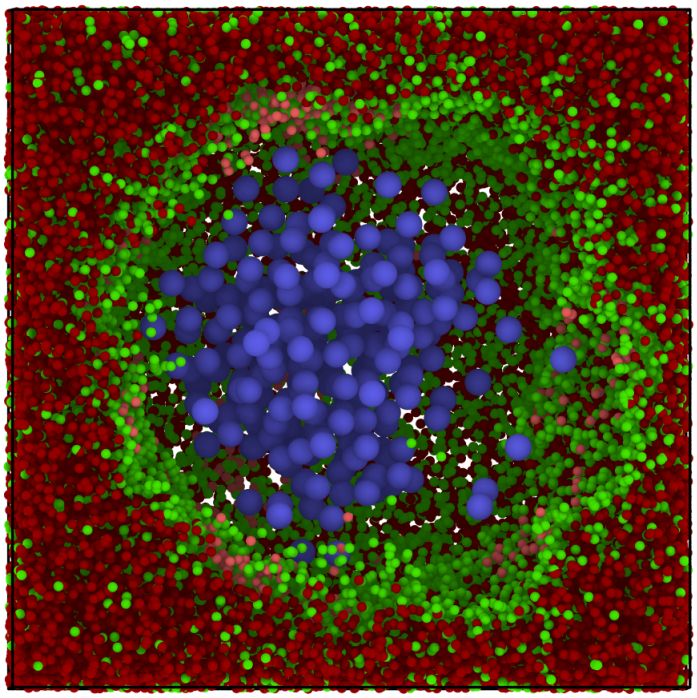


Figure 12: Rendering of the application of the barostat algorithm. The red particles are incoming gas particles, that have not yet interacted, pink particles are currently interacting and green particles are leaving the simulation box and moving away from the nano particle (blue). To render this image, a cut has been made to make a view into the simulation box possible.

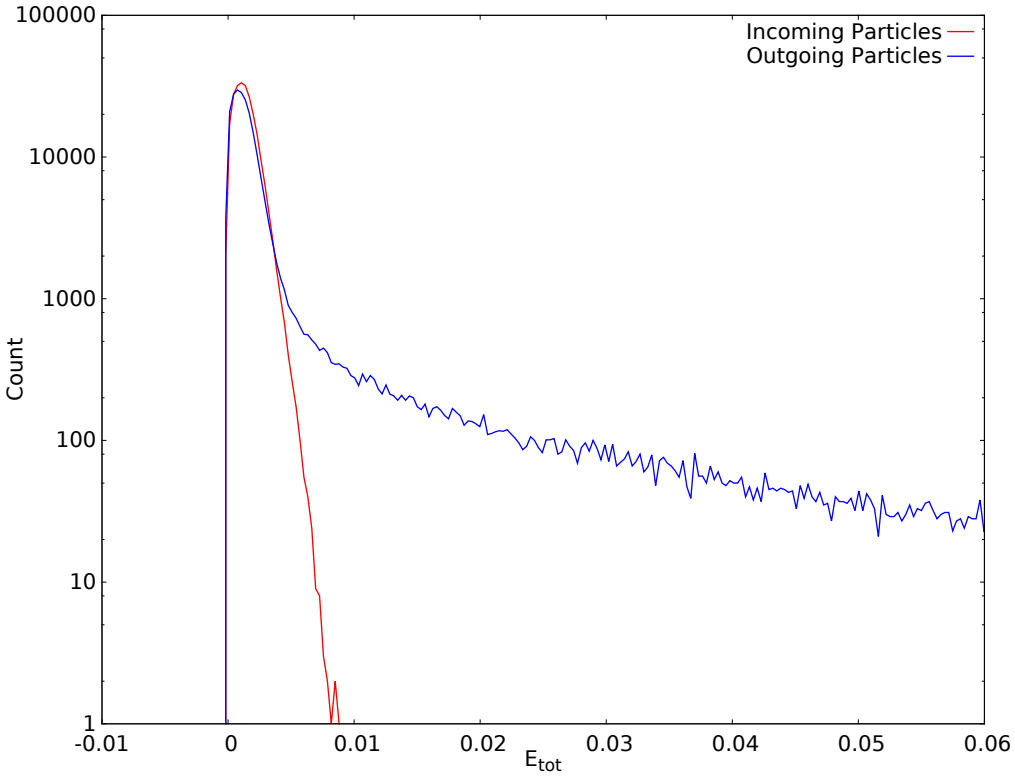


Figure 13: Histogram of total energies for incoming (red) and outgoing (blue) particles. The parameters for this histogram are $\Delta Q^* = 0.04$, $T^*_{\text{ambient}} = 0.08$, $P^* = 0.8$. What can be seen here is that the outgoing particles have a higher count of particles with more total energy.