CLOSE ENOUGH FOR GAS

by

Anders Hafreager

THESIS

for the degree of

MASTER OF SCIENCE

(MASTER I FYSIKK)

Faculty of Mathematics and Natural Sciences
University of Oslo

October 2013

Det matematisk- naturvitenskapelige fakultet
Universitetet i Oslo

Abstract

This is an abstract text.

To someone

This is a dedication to my cat.

Acknowledgements

I acknowledge my acknowledgements.

Contents

List of Figures

List of Tables

Part I

Introduction

Chapter 1

Introduction

1.1 Introduction

1.1.1 Motivation

1.1.2 The structure of this thesis

Chapter 2

Background

2.1 History

2.1.1 Gas dynamics

2.1.2 The breakdown of contiinum

2.1.3 Porous media

2.1.4 Nanoporous media

2.2 Applications

2.2.1 Shale gas extraction

2.2.2 Electronic devices

2.3 Multi scale physics

2.3.1 Our world

Throughout the history, from the ancient greeks up until today, humans have tried to understand the rules of our universe and how they affect what we see

and experience every day. Today, we know that everything is controlled by the rules of quantum mechanics, where some of its effects are visible for the naked eye. An example is seen every summer during hot days while driving a car on a straight road; mirage. The warm road reflects light as if it's covered in water, so you can see the sky and oncoming cars. This phenomena is explained by quantum electrodynamics and the fact that there is a temperature gradient in the air pointing from the asphalt and upwards. Light moves slower through dense air, and the air is less dense at high temperatures, so the light wants to travel closer to the asphalt. This means that the actual path the light takes is not a straight line, but is bent, see figure **??**. The light is taking a shortcut. Mirage can be explained, or should I say modelled, by simpler ideas, but the actual reason arises from quantum mechanics.

2.3.2 Simulation of the universe

Say we want to use our computers and simulate a universe very similar to our own. We will use all our knowledge about physics, so the fundamental theory we plug in is quantum mechanics. We need to account for relativistic effects, so we have to unify quantum theory with general theory of relativity first. A universe, even a tiny one, contains a lot of information, so the amount of required computer memory is enormous. In order to solve the fundamental equations of unified relativistic quantum mechanics with todays techniques, we will probably need a very small timestep. Larger time scales, like the ones needed to compute cosmological properties of the universe, are therefore out of reach even with decades of Moore's law.

2.3.3 An ideal world

2.3.4 The weakest link

Part II

Molecular Dynamics

Chapter 3

Introduction

Molecular Dynamics is an numerical model that describes the behaviour of liquids, gases and solids with the finest scale of any classical model. We study the dynamics of single atoms and how they interact with each other forming molecules and larger objects. The idea is simple, and has been used since the time of Sir Isaac Newton in the 17th century when he discovered his laws of motion. With the knowledge of the relevant forces, we can solve Newton's equations and calculate the dynamics of the atoms. In this chapter, we will discuss how to implement an efficient Molecular Dynamics program with advanced force fields allowing us to do calculations on systems with silica (SiO2) and water (H2O). This will be used to study water flow in nanoporous silica, which is of great importance for geological, biological and technological applications.

3.1 The model

The state of a molecular dynamics system is fully described by seven variables per atom, three positions, three velocities plus the atom type. These phase variables are evolved through the laws of motion. The atomic forces are calculated as the gradient of some potential that can differ quite a lot depending on the requirements of the model. System statistics are sampled as ensable averages through ergodicity over large times.

3.1.1 Force calculation - potentials

In general, the potential energy is given as a function summing over the configurations given by the atom positions

$$U(\mathbf{r}) = \sum_{i<j} U_2(r_{ij}) + \sum_{i<j<k} U_r(\mathbf{r}_i, \mathbf{r}_j, \mathbf{r}_k) + ..., \tag{3.1}$$

where $\mathbf{r}$ is the phase space point, U_n is a function of the positions of n atoms, r_{ij} is the relative distance between atom i and j, $\mathbf{r}_i$ is the position of atom i. Advanced potentials, such as ReaxFF, have 5-atom contributions, or more, to the energy. The reason for this is simple; when three atoms are close to each other, their electrons are positioned differently as if there were only two atoms. These effects play a large role in forming molecules, such as water. In this thesis, we will only use potentials using two- and three-particle terms. Numerically, force calculations are the most expensive part of the whole program, so for simple systems, or for educationally purposes, we might not need the most advanced ones. The simplest, yet remarkable for many purposes, potential students often meet is the Lennard-Jones potential.

Lennard-Jones potential

We often see this potential referred to as the 6-12 potential, which is a simple function that contains the main properties of atomic forces, the short-ranged Pauli repulsion and the long-ranged van der Waals force. The potential is between pairs of atoms only

$$U_{LJ}(r) = 4\epsilon \left[\left(\frac{\sigma}{r_{ij}} \right)^{12} - \left(\frac{\sigma}{r_{ij}} \right)^{6} \right], \tag{3.2}$$

where r is the relative distance between the two atoms, ϵ and σ are coupling constants giving the depth of the potential well and the distance where the potential is zero. We can extend this potential to behave differently for several atom types, by allowing the coupling constants to be dependent of the two interacting atoms. The Lennard-Jones potential is then written as

$$U_{LJ}(r) = 4\epsilon_{AB} \left[\left(\frac{\sigma_{AB}}{r_{ij}} \right)^{12} - \left(\frac{\sigma_{AB}}{r_{ij}} \right)^{6} \right], \tag{3.3}$$

where the coupling constants are specified for interacting atom pairs of type A and B. If the system has only two different atoms, it is usually called

a *binary Lennard-Jones fluid*. The force is given as the gradient of this potential yielding

$$F_{LJ}(\mathbf{r}) = -\nabla U_{LJ}(r_{ij}) = 24\epsilon_{AB}\left[\left(\frac{\sigma_{AB}}{r_{ij}}\right)^{13} - \left(\frac{\sigma_{AB}}{r_{ij}}\right)^{7}\right]\mathbf{u}_r,$$

where $\mathbf{u}_r$ is the unit vector pointing from atom i to atom j.

Silica-water potential

3.1.2 Time integration

As mentioned in the introduction, the system is evolved following Newton's equations of motion. These equations are integrated by some integration scheme.

3.1.3 Time integration

3.2 Physical properties

Molecular dynamics is used to evolve the system in time, and the phase space variables can then be used to sample statistical properties of the system. Properties of interest are kinetic and potential energy, temperature, pressure, diffusion constant and different correlation functions (such as the pair correlation function, cage-correlation function, static and the dynamic structure factor). In this section, we will define these properties and discuss how to measure them.

3.2.1 Kinetic and potential energy

We measure the kinetic energy directly through its definition for point particle objects

$$E_k = \sum_i \frac{1}{2} m_i v_i^2, \tag{3.4}$$

and the potential energy through (3.2).

3.3 c++-code

3.4 F77-code

Part III

Direct Simulation Monte Carlo

Chapter 4

Kinetic theory

4.1 Origin

The Boltzmann Equation

H-theorem

4.2 The Chapman-Enskog method

4.3 Useful things

4.3.1 Equipartition theorem

4.3.2 Maxwell speed distribution

Chapter 5

Direct Simulation Monte Carlo

Even though Molecular Dynamics is a very precise model that can give deep, detailed insight about advanced systems, it is computationally extremely expensive. As mentioned in section TODO, the size of a typical Molecular Dynamics system is of order a few nanometers. In nanoporous materials such as shales, the gas is dilute, so the said system might only have a few hundred free gas molecules. In this section, we will discuss

5.0.3 The model

5.1 Surface interactions

The effects of surface interactions become significant as the pore sizes decrease. For very small pores, the number of atoms near the surface is comparable with total number of atoms. In MD, these effects are automatically taken care of through the atomic forces, but in DSMC, we need a surface interaction model. We investigate four different models in this section.

5.1.1 Specular wall

The specular wall behaves just like a classical mirror; the colliding objects are reflected so that the normal component of the velocity is reversed whereas the tangential components remain unchanged.

5.1.2 Thermal wall

If we instead think of the wall as an object with a given temperature T_w, we can imagine that the molecules go into the wall, collide with the wall atoms as a random walk, and return with no correlation with the incoming velocity. We can then choose a new, random velocity vector from a distribution so that the gas temperature converges to the wall temperature. We can find this by looking at the velocity distribution of particles going through an imaginary wall during a time Δt.

An imaginary wall

We think of an area A randomly placed inside an isotropic gas, and look at the number of particles with velocities in the range $(v, v+dv)$ with direction $(\theta, \theta+d\theta)$ and $(\phi, \phi+d\phi)$. Since the gas is isotropic, the number of particles having velocities within these ranges is

$$\frac{N}{V} f(v)dv\frac{d\Omega}{4\pi} = nf(v)dv\frac{d\Omega}{4\pi},$$

where $d\Omega = \sin\theta d\theta d\phi$. During a time Δt small enough that there are no collisions between particles, the volume of the prism is $v\Delta t A$, and the number of collision per time per area is

$$\frac{n}{4\pi} v f(v)dv \sin\theta d\theta d\phi.$$

If we only look at the particles passing through from one side of the wall, we integrate over the angles and get the number density

$$\begin{aligned} d\nu(v) &= \frac{n}{4\pi} v f(v)dv \int_0^{2\pi}\int_0^{\pi/2} d\theta d\phi \sin\theta \\ &= \frac{n}{4} v f(v)dv. \end{aligned}$$

The total number of particles that passes through the wall is

$$\nu = \int_0^\infty \frac{n}{4} v f(v)dv = \frac{n}{4}\langle v\rangle.$$

We get the normalized distribution by looking at

$$f_e(v) = \frac{d\nu}{\nu} = \frac{v}{\langle v\rangle} f(v)dv = \frac{1}{2}\left(\frac{m}{kT}\right)^2 v^3 e^{-mv^2/2kT} dv$$

which is very similar to the Maxwell distribution but with slightly faster particles.

5.1.3 Maxwell scattering

$$R(\mathbf{v}' \to \mathbf{v}; x) = (1 - \sigma_v)\delta(\mathbf{v}' - \mathbf{v} + 2\mathbf{n}v_n) + \frac{2\sigma_v\beta_w^4}{\pi}\exp(-\beta_w^2 v^2) \tag{5.1}$$

5.1.4 The Cercignani-Lampis model

Maxwell's scattering kernel was widely used until the 1960s, before more detailed models were developed. As of today, the most popular one is the Cercignani-Lampis model **??**. The kernel is given as

$$\begin{aligned} R(\mathbf{v}' \to \mathbf{v}; x) = {} & \frac{2\sigma_n\sigma_t(2-\sigma_t)\beta_w^4}{\pi} \\ & \times \exp\Big(-\beta_w^2 \frac{v_n^2 + (1-\sigma_n)(v_n')^2}{\sigma_n} - \beta_w^2 \frac{(v_t - (1-\sigma_t)v_t')^2}{\sigma_t(2-\sigma_t)} \Big) \\ & \times I_0\Big(\beta_w^2 \frac{2\sqrt{1-\sigma_t}v_n v_n'}{\sigma_n}\Big), \end{aligned} \tag{5.2}$$

where v_n and v_t again are the normal and tangential components of the velocities and I_0 is the zeroth-order modified Bessel function of the first kind.

5.1.5 The random surface model

5.2 Complex geometries

All the surface interaction models from the previous section have the surface normal and tangent vectors as input parameters. These vectors are easy to determine if the system consists of two parallel plates in the xy-plane, or any other mathematically well described geometry. These systems are interesting as validation test cases, but most real world materials have a more complex geometry without any simple mathematical description. A very much used representation of such geometries is a triangle mesh in which the surface consists of many connected triangles. The triangles have a well defined normal vector and tangent plane which is easy to calculate. In this thesis, I have chosen another approach by representing the system as a large, binary three-dimensional matrix. In this section, we will discuss how to create

such a matrix, how to identifying the surface points and how to calculate the mentioned vectors.

5.2.1 Binary representation

With this method, any system geometry is fully described by a three dimensional matrix with dimensions $m \times n \times l$. Each matrix element represents a voxel in the physical space, and can take values 0 or 1. A value of one means that the voxel is filled, whereas zero means empty. No particles can be inside a filled voxel, so this is how we do surface collision detection.

5.2.2 Collision detection

We define a collision as whether or not a particle has moved into a wall during the timestep Δt. This has to be checked for every particle each timestep, and in the case of a collision, we need to calculate the resultant velocity. The collision detection algorithm is best illustrated by a code example:

```
bool did_collide(double *position) {
        int voxel_index_i = position[0] / system_length[0] *
            num_voxels[0];
        int voxel_index_j = position[1] / system_length[1] *
            num_voxels[1];
        int voxel_index_k = position[2] / system_length[2] *
            num_voxels[2];

        // The world matrix is a binary matrix
        return world_matrix[voxel_index_i, voxel_index_j,
            voxel_index_k];
}
```

This is just a quick memory lookup. The really expensive part of the full collision algorithm is finding exactly which voxel is the first surface voxel the particle hits. We need to precalculate all the surface voxels so they are marked in the matrix during runtime.

5.2.3 Identifying the surface voxels

Given the binary matrix, we have identified every solid part of the system. The voxels inside a wall that are not part of the surface all have neighbouring voxels that are also marked as walls. We *define* the surface as the filled voxels that have less than 26 neighbouring filled voxels. The algorithm could be implemented like this (one would also have to take care of the periodic boundary conditions, but that is not important to illustrate the idea):

```
bool is_surface(short ***world_matrix, int voxel_index_i, int
    voxel_index_j, int voxel_index_k) {
        for(int i=-1;i<1;i++) {
        for(int j=-1;j<1;j++) {
                        for(int k=-1;k<1;k++) {
                                if(i == j == k == 0) continue; //
                                    Skip self
                if(world_matrix[voxel_index_i + i][voxel_index_j
                    + j][voxel_index_k + k] == 0) {
                        // This neighbour is empty
                        return true;
                }
            }
        }
    }

    return false;
}
This has to be done for every voxels in the system, but only once
    per system.
```

5.2.4 Calculating normal and tangent vectors

5.3 Pressure driven flows

5.3.1 Gravity

5.3.2 Real pressure

5.4 Parallelization

5.5 Code

5.6 Code validation

Every time a physicist implement a model, it is important to verify that the implementation is correct. New models that describe unknown areas of physics (such as new length scales) might be difficult to confirm if there are no experiments or comparable models available.

5.6.1 Velocity distribution

The simplest

Specular wall

Thermal wall

Maxwell kernel

Cercignani Lampis kernel

Random surface

Part IV

Visualization