Computer Physics Communications 185 (2014) 3156–3161

Contents lists available at ScienceDirect

Computer Physics Communications

journal homepage: www.elsevier.com/locate/cpc

Simulation of a hard-spherocylinder liquid crystal with the *pe*

Ellen Fischermeier [a,*], Dominik Bartuschat [b], Tobias Preclik [b], Matthieu Marechal [a], Klaus Mecke [a]

[a] Institut für Theoretische Physik I, Friedrich-Alexander Universität Erlangen-Nürnberg, Staudtstraße 7, D-91058 Erlangen, Germany

[b] Lehrstuhl für Systemsimulation, Friedrich-Alexander Universität Erlangen-Nürnberg, Cauerstraße 11, D-91058 Erlangen, Germany

ARTICLE INFO

Article history:
Received 11 February 2014
Received in revised form
14 July 2014
Accepted 19 August 2014
Available online 27 August 2014

Keywords:
Parallel framework
Rigid body dynamics
Hard rod
Liquid crystal
Frank elastic constants

ABSTRACT

The *pe* physics engine is validated through the simulation of a liquid crystal model system consisting of hard spherocylinders. For this purpose we evaluate several characteristic parameters of this system, namely the nematic order parameter, the pressure, and the Frank elastic constants. We compare these to the values reported in literature and find a very good agreement, which demonstrates that the *pe* physics engine can accurately treat such densely packed particle systems. Simultaneously we are able to examine the influence of finite size effects, especially on the evaluation of the Frank elastic constants, as we are far less restricted in system size than earlier simulations.

1. Introduction

The *pe* physics engine [1,2] is a software framework for simulating large-scale multi-body systems in soft and hard contact. The simulated bodies have spatial extension and their core shapes are assumed to be rigid. The software supports various numerical methods to resolve the multi-contact problems such as discrete element methods (DEM) [3] for soft contacts and the fast frictional dynamics (FFD) [4] and non-smooth contact dynamics (NSCD) [5] methods for hard contacts. The physics engine was successfully used to simulate large-scale granular flows [6] on its own but also coupled to the waLBerla lattice Boltzmann framework [7] for the simulation of particulate flows [8] and self propelled swimmers [9]. The software framework was designed from the ground up targeting distributed-memory architectures and has been shown to scale well up to the full size of today's largest supercomputers [6,10,9]. In particular, the FFD method qualifies well for the parallelization since the contact problems are treated in a localized manner keeping the communication amount to a minimum. The FFD method is originally based on Moreau's midpoint rule, an established time-stepping scheme for simulating mechanical systems in frictional contact [11]. It uses a time-integrator equivalent to a Leapfrog Verlet integration if no contacts are present. The friction model is derived from the principle of maximal dissipation and captures essential frictional properties [4]. However, the contact problem simplification stemming from the localization and the special type of friction model require a thorough validation of the numerical method. This paper performs such a validation on the basis of a hard-spherocylinder system as a model for liquid crystals, where the implementation of the FFD method within the *pe* framework replaces conventional molecular dynamics techniques.

Studies of hard-spherocylinder systems date back to the '70s and the work of Few and Rigby [12] and Vieillard-Baron [13]. Spherocylinders consist of a cylinder of length L and diameter D with hemispherical end caps. For $L/D = 0$ this model system corresponds to the hard-sphere model whereas for $L/D \to \infty$ the system consists of infinitely thin needles. Systems of rigid straight rods have also been realized experimentally for example in the form of tobacco mosaic viruses [14] and colloidal silica rods [15].

All phase transitions of such a system are purely driven by entropy as the hard particles interact exclusively via excluded volume. Therefore, the temperature k_BT acts only as an energy scale. Two types of entropy compete here: The orientational entropy drives the system towards an isotropic phase where both center of masses and orientations are uniformly distributed (see Fig. 1a,b). The second type of entropy, translational entropy, favors a uniform orientation of the particles which minimizes the excluded volume. An example of such an ordered phase is the nematic phase which is characterized by alignment of the particles with respect to each other, giving rise to an anisotropic angular

* Corresponding author. Tel.: +49 91318528450.
E-mail address: ellen.fischermeier@fau.de (E. Fischermeier).

http://dx.doi.org/10.1016/j.cpc.2014.08.014
0010-4655/

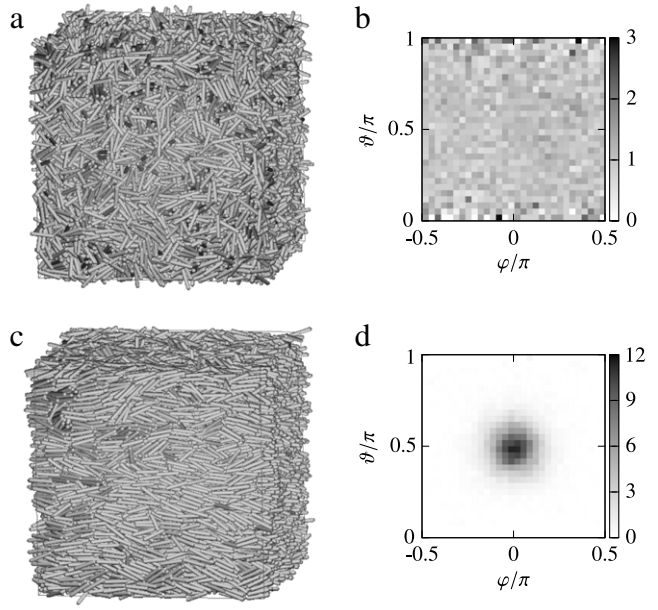

Fig. 1. Snapshots of the $L/D = 5$ hard-spherocylinder system in (a) an isotropic ($\rho^* = 0.4$) and (c) a nematic ($\rho^* = 0.5$) state with the corresponding angular distribution profiles ((b) isotropic, (d) nematic) normalized to the ideal isotropic distribution profile. (For definition of dimensionless density ρ^* see Section 2.2.) The increased noise around $\vartheta = 0$ and $\vartheta = \pi$ in (b) is due to the reduced statistics in this regime as the size of the solid angle bins behaves like $\sin(\vartheta)\mathrm{d}\vartheta\mathrm{d}\varphi$.

distribution profile while keeping a homogeneous distribution of the center of masses (see Fig. 1c,d). This combination of liquid-like translational and crystal-like orientational properties which extends to many observable physical properties is also the origin of the name liquid crystal.

For $L/D \to \infty$ in the Onsager limit [16] the hard-spherocylinder system exhibits a phase transition from isotropic to nematic at vanishing volume packing fraction (order D/L). For systems with a finite aspect ratio L/D and full translational and rotational freedom, Frenkel and co-workers were the first to report such a phase transition with molecular dynamics (MD) and Monte Carlo (MC) simulations [17,18]. They also discovered a crystalline solid phase and a smectic A phase where the system exhibits layering of the particles, thus having a one-dimensional positional ordering in addition to the orientational ordering. In their later work [19,20] they demonstrated that the hard-spherocylinder system indeed exhibits a rich phase diagram which depends on the aspect ratio of the particles as well as on the packing fraction, as confirmed by the studies of McGrother et al. [21].

This well understood system provides an ideal test case for our simulation framework, the *pe* physics engine, in the context of micro-scale multi-particle systems. The massive parallelism of the *pe* software framework enables us to study far larger systems than those considered in the publications mentioned above. Those were restricted to a few hundred particles with only a couple of exceptions which included up to a few thousand particles. We can easily simulate systems with 20 times more particles without reaching the limits of computational power. This gives us the possibility to improve the confidence level of the results for the observables studied in such a hard-spherocylinder system. We focus on the $L/D = 5$ system at a single packing fraction in the nematic phase where the particles are rather densely packed. As it is known that equilibration and numerical treatment become difficult in this regime, these are ideal conditions for testing the power of our framework.

The paper is structured as follows: Section 2 describes the *pe* rigid body physics engine in detail and depicts both, the simulation setup, and the specifics of the model system we employ. In Section 3, we elaborate on the variables we evaluate in this work, which are the order parameter, the pressure, and the Frank elastic constants. We further give a short overview over their theoretical description. The results of this work are presented in Section 4 and some conclusions are drawn in Section 5.

2. Numerical methods and simulation setup

2.1. The rigid body physics engine *pe*

The *pe* physics engine is a powerful tool for large-scale rigid body dynamics simulations [1,2]. The algorithm we chose to employ in this work is based on the FFD solver first proposed by Kaufman et al. [4,22], where we set all friction coefficients to zero to make our results comparable to previous studies. It is parallelized with MPI [23] following a domain decomposition approach: Each process is responsible for the time-integration of the bodies whose center of mass is located in its associated computational subdomain. In addition to these local bodies processes obtain shadow copies of bodies intersecting their subdomain. Algorithm 1 lists pseudo-code for the time step procedure.

Algorithm 1 Rigid Body Time Step of Size dt [1, 2]

```
// 1. MPI message exchange: Reduce forces and torques

// First half time step
for each local rigid body B_i do
   advance position and orientation for dt/2
   advance translational and angular velocity for dt/2
end for

// 2. MPI message exchange: Synchronize rigid bodies

// Collision detection
for each local rigid body and shadow copy B_i do
   detect all contacts C(B_i)
   for all contacts k ∈ C(B_i) do
      determine acting constraints
   end for
end for

// 3. MPI message exchange: Gather contact constraints

// Collision resolution and second half time step
for each local rigid body B_i do
   if B_i has constraints then
      find post-collision translational and angular velocity
   else
      advance translational and angular velocity for dt/2
   end if
   advance position and orientation for dt/2
end for

// 4. MPI message exchange: Synchronize rigid bodies
```

The time step starts by sending all forces and torques applied to shadow copies to the respective owner process of the body. The owner process can then reduce them and proceed by performing the time-integration up to the midpoint of the time step neglecting contact constraints on purpose. Subsequently, the shadow copies need to be updated to the new positions and velocities and migrations of bodies to neighboring processes have to be treated in a second message exchange communication step. Next, collisions