Brownian Disks Lab: User's Manual (v1.1)

Pablo Domínguez-García.
Dpto. Física Interdisciplinar.
Universidad Nacional de Educación a Distancia (UNED).
Madrid, Spain.

October 18, 2019

CONTENTS

Chapter 1

BASIC USE OF BROWNIAN DISKS LAB.

1.1 Motivation.

Brownian Disks Lab (BDL) is a software developed in Easy Java/Javascript Simulations (EjsS) [1] for the generation of movement of two-dimensional Brownian disks subjected to different external applied forces and interactions between particles. This software develops a model of colloidal fluid using Brownian Dynamics (BD) simulations to compare with time-lapse microscopy experiments. We use a computational multi-platform environment with visual inspection of the movement of colloidal particles ($n \leq 500$) in real time. This software is designed to be experimental-like [2, 3, 4] by emulating a suspension of micro-spheres suspended in a Newtonian fluid observed using a typical time-lapse microscopy set-up in a two-dimensional or pseudo-2D configuration. In analogy with a image-based time-lapse microscopy lab, BDL allows to change external parameters such as concentration of particles in the suspension, viscosity of the fluid, focal distance, size of the spheres, temperature of the bath, etc.

In these simulations, we use real units for the input parameters (SI units), instead of the typical dimensionless quantities, so the simulated lab is quite similar to a real experiment. In a time-lapse microscopy experiment, the positions of the objects are recorded in units of the screen (pixels) and a conversion factor is needed to obtain the real positions in micrometers, so in BDL we will also need a similar calibration factor to convert screen to meter-units. Finally, in an image-based experiment, an analysis and filtering of the images has to be done before tracking the particles, but in BDL each particle is already labeled and the trajectories of each object may be obtained easily after the simulation ends. Additionally, our physical model allows to introduce external forces in the spheres with the aim of testing different experimental conditions and interactions observed in colloidal physics experiments. In the published short paper of this software, we used an example related to optical trapping forces, but here we will describe different types of forces, such as the generated between the spheres which are in contact (hard or soft spheres), and long-range forces which seem to appear in charged and confined colloids.

In the next sections, we will explain the basic running of BDL through its interface. In the subsequent chapters, we will describe the physics and theoretical model underlying the application, how to use the data obtained from the simulation, and finally some details about the Java code developed.

1.2 Running the software.

Before initializing the program, we must check if Java is installed in the user's computer. If not, please download Java[1]. In case of using Debian/Ubuntu Linux, you can install a Open Java virtual machine version (JRE)[2]. When the Java virtual machine (JRE) is installed in the system, **BDL can be executed by double-clicking** in the file named `Brownian_disks_lab_v1.1.jar` or through a Linux command terminal:

```
java -Xmx2048m -jar Brownian_disks_lab_v1.1.jar
```

The option -Xmx2048m in the command line increases the maximum available memory to 2GB for storing the data of the particles' positions. In most cases, this increase of the user's available memory will not be necessary, because in the software we can choose to automatically save pieces of data for liberating memory.

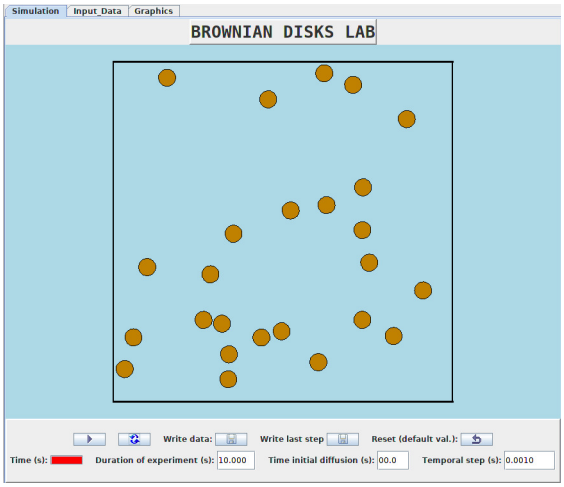

Figure 1.1. Screen capture of the initial view of BDL.

When executing, BDL shows a window like the one in the figure 1.1, where we can see 25 disks (10 by default) inside a box delimited by black lines. Their initial position is random inside the box. The disks will begin to move diffusively if we click on the play button.

There are three panels in the top part of the window. The one showing the motion of the disks is called "Simulation" (fig. 1.1). This panel also contains a few but important basic inputs, such as the total duration of the experiment and the internal temporal

[1] `http://www.java.com/es/download/`
[2] sudo apt-get install default-jre

step used in the resolution of the differential equations. By clicking on the panels in the top part of the window, we can access to the rest of input data and to some graphics describing the developing of the simulations. The following section details the contents of every panel.

1.3 Panels.

Three panels are used in BDL: the first ("Simulation") contains the moving disks and some basic inputs, the second ("Input") allows the modification of the input parameters needed for the simulation, and the third ("Graphics") includes several graphics to evaluate how the simulation is developing. We will describe every element contained in those panels.

Important note: When clicking in the fields to change numerical or text values, the field will turn yellow. Then, you need to press "Enter" to change the input value in the field.

1.3.1 Simulation panel.

Figure 1.1 is a typical image of the "Simulation" panel. In the center of the panel, we can see the box where the disks move according with the theoretical model. The box is limited by black lines. At the bottom, there are two rows. The first row contains the following options (left to right):

- *Play* (▶/ ▮▮): Play the simulation. Press again to pause.
- *Refresh* (🔄): The simulation initializes but without deleting the input values and options previously chosen.
- *Write data* (🖫): Saves all the position of the disks into a text file in the hard drive of the user's computer. An input file containing all the input parameters used in the simulation is also saved (see section 1.4). The path and name of the file are defined in the "Input" panel (see section 1.3.2).
- *Write last step* (🖫): Saves only the position of the disk in the last step (temporal step) of the simulation. Saves also the input file. This can be useful for reading the file and beginning a new simulation using the last configuration of another one (see section 1.4).
- *Reset* (↺): Initialize the simulation to default values.

The second row contains:

- A bar to indicate visually how much time of the experiment has been elapsed (red if empty).
- *Duration of the experiment*: total duration in seconds of the experiment. This number is limited by the maximum number of lines (events) set to 5×10^5. If dt is small enough and the number of particles is high, the total duration of the experiment is automatically reduced to a maximum value, according to the maximum number of lines.
- *Time initial diffusion*: this is an extra time that can be added to the simulation. During that time, the particles will diffuse, but their trajectories are not recorded. It is recommendable to use when the disks are very close in the initial configuration.
- *Temporal step*: it is the internal time step of the simulations. In case of using an optical trap, this value is changed automatically to $dt = 10^{-4}$ s, to avoid bad definition of the trapping force.

1.3.2 Input Data panel.

The "Input Data" panel can be view in the figure 1.2.

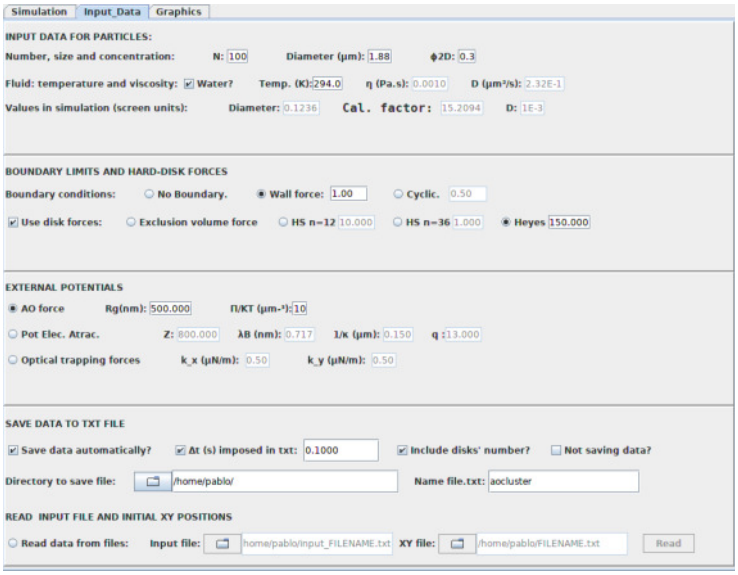

Figure 1.2. Screen capture of the Input panel of BDL.

The section *INPUT DATA FOR PARTICLES* contains the following elements, regarding the properties of particles and the fluid, from top to bottom, and from left to the right:

- *Number, size and concentration*:
  - N: number of particles. Limited from 1 to 500.
  - *Diameter*: Diameter of the disks in real units, in micrometers (μm).
  - ϕ_{2D}: Two-dimensional concentration of particles in the box. This number can be used to imitate a focal distance value, to see the particles closer or farther.
- *Fluid: temperature and viscosity*:
  - *Water*: it indicates if we are using water as the surrounding fluid. If so, viscosity (and the diffusion coefficient) are calculated directly though temperature, T, according to the following expressions:

    $$\eta[\text{Pa.s}] = 10^{-3} \times (0.2727 + 1.4893 \exp(273 - T)/28.693); \quad D = k_B T/6\pi\eta a$$

    where k_B is the Boltzmann constant and a is the radius of the particles. All magnitudes are given in SI units. We are assuming that the disks are two-dimensional projections contained in a equilibrium layer where the viscosity value is referred to the 3D bulk fluid.
  - η[Pa.s]: if water, its value is given by the formula expressed before. If not, the value has to be introduced.
  - $D[\mu\text{m}^2/s]$: diffusion coefficient in standard units. This value is automatically calculated from the viscosity, radius of the disks, and temperature of the fluid according to the expression given before.
- *Values in the simulation (screen units)*: Not editable, they are calculated from the previous values and are showed only for information
  - *Diameter*: Diameter of the disk in units of the screen.
  - *Cal. factor*: **Important**: the saved (x, y) positions values have to be multiplied by this calibration factor to be converted in micro-meter units.
  - D: Diffusion coefficient in squared units of the screen per second.

The section *BOUNDARY LIMITS AND HARD-DISK FORCES* summarizes the direct interaction between disks in contact and how they behave when reaching the limit of the box. This section has the following elements, from top to bottom, and from left to the right:

- *Boundary conditions*:
  - *No boundary*: Particles go out of the box. Positions are recorded in any case.
  - *Wall Force*: A force proportional and opposed to the velocity of the disk approaching the wall. The number in the numeric field is a factor which multiplies that force. By increasing that value, the force generated by the wall increases when a disk is in contact with it. By default, this number is 1.0, which provides good results at high concentration values.

  - *Cyclic*: Or Pac-Man conditions. The particles touching or surpassing a box side disappear and appear from the opposite side. The numeric factor is the proportion of the radius of the disk needed for applying the condition of appearing from the opposite side. If this number is equal to 1, means that the disks can exceed the limits of the walls by a length equal to one radius before appearing from the opposite side. If 0, the cyclic condition is applied when the edge of a disk touches the walls. It is set to 0.5 (half of the radius) by default.

- Forces between disks in contact:
  - *Use disk forces*: If marked, we will use a repulsive force between the borders of the particles in contact. If not, the particles will overlap.
  - *Exclusion volume force*: Simple exponential exclusion force (see section 2.1.1).
  - *HS n=12*: Hard-sphere potential with $n = 12$ (see section 2.1.1).
  - *HS n=36*: Hard-sphere potential with $n = 36$ (see section 2.1.1).
  - *Heyes*: Heyes and Melrose's method of separating disks (see section 2.1.1).

The section *EXTERNAL POTENTIALS* summarizes the interaction at long distances between disks. This section contains the following elements:

- *AO force*: Marked to use Asakura and Oosawa potential (fields are explained in section 2.1).
- *Pot. Elec. Atrac.*: Phenomenological pair potential proposed by Grier and Han for an attractive electrostatic potential (for further details, see section 2.1).
- *Optical trapping forces*: Restoring forces with stiffness constants k_x and k_y for x and y coordinates. These values are given in μN/m, which are the usual in optical tweezers (explained in sections 2.3 and 2.4).

The *SAVE DATA TO TXT FILE* section indicates the input instructions needed to save particle's positions into a text file:

- *Save data automatically?*: Marked as true to save data in the text every 10 lines. This is an advisable option to save system's memory for long runs. If this field is checked, the save button of the "Simulation Panel" appears as not enabled.
- Δt *(s) imposed in txt*: For saving the data only for disks' jumps in a certain lapse of time. If it is not marked, every data for the time step of the simulation is stored. This is the mimic of the temporal step in an actual time-lapse microscopy experiment. The dt value in the "Simulation" panel is the internal step of the simulations, while this Δt can be used to saving the data for controlled time-lapses. Of course, $dt \leq \Delta t$.

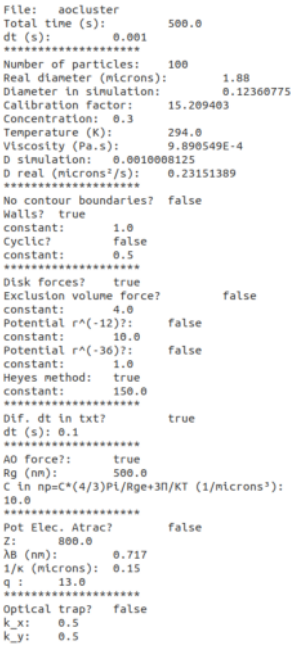

Figure 1.3. Example of file "input_FILENAME.txt", which saves the input information used in the simulation.

- *Include disks' number?*: If enabled, the text file will include a final row with a number identifying every particle. If disabled, that data column is not included.
- *Not saving data?*: This option allows a free mode run, and the disks move without saving in memory any data regarding their motion. This mode allows to run the simulation without any temporal limitation. In any case, the last step of the simulation can be saved by using one of the "write" buttons in the Simulation Panel.
- *Directory to save file*: If clicked, a window will appear for selecting the directory where to save the text file. It can be also edited by writing the desired path of the file. By default, this path is the personal user's directory indicated by the operating system.
- *File name*: The name of the text file without the text extension (it will be *.txt). By default is "FILENAME" and the file will be "FILENAME.txt" stored in the user's directory. The input file will be named using this string.

Finally, we have a *READ INPUT FILE AND INITIAL XY POSITIONS* section. Here, we can read an input file previously generated by the software and also a file containing particle's positions. From this last file, the software will read only the disks'

positions of the last step. Then, BDL will load the initial positions of the particles and plot their positions in the Simulations Panel:

- *Read data from files*: If selected, it activates the following fields:
- *Input file*: Location of the input file to read. It must have the same structure that the input file generated by the software. Fig. 1.3 shows a complete input file. This field can not be empty if "Read data from files" is selected.
- *XY file*: Location of the XY file for reading disks' positions. This file can be a complete XY file generated by the software or a simple file with only one step (one temporal step). The input file and this file are read at the same time and both have to be coherent with each other (same number of particles per step). If this field is empty, only the input file will be read. See section 1.4.1 for further instructions.

1.3.3 Graphics panel.

The last panel is named "Graphics" (figure 1.4). In this panel, we can check the basic statistics of Brownian motion of the disks, and the Boltzmann statistics when we use an optical trap (see section 2.4). In this panel, the trajectory of one disk and the number of particles inside the simulation box are also shown. All the graphs evolve with time.

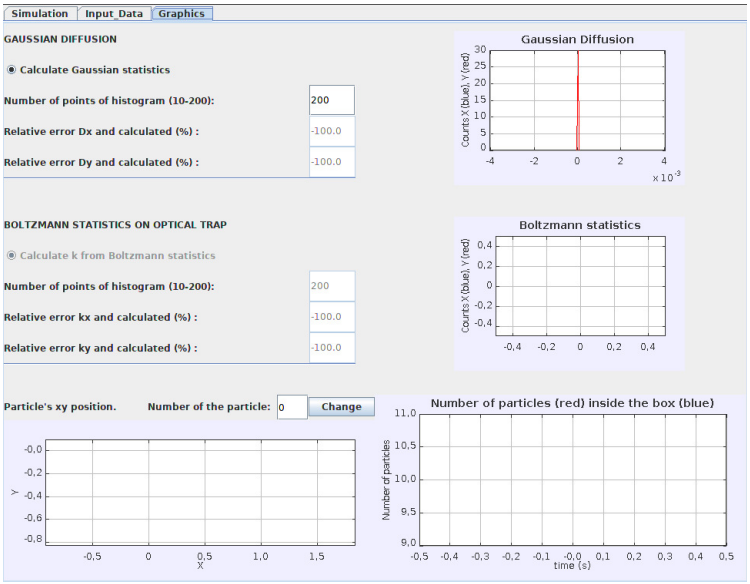

Figure 1.4. Screen capture of the Graphics panel of BDL.

In the top part of the panel, we calculate the **Gaussian statistics** (see section 2.4) related to the jumps of the particles. The contents of the graph are: total counts in the y-axis, and $\Delta x = x(t + dt) - x(t)$ and $\Delta y = y(t + dt) - y(t)$ in the x-axis, blue and red lines respectively. These jumps are given in units of the screen, not in microns. This calculation of the histogram is made in every step of the simulation, so we can see how the distribution is generated with time. In the left zone, we can enable or disable this calculation by pressing "Calculate Gaussian statistics". We can also define the number of points of the histogram (this should be done prior to the initialization of the simulation).

In the not-editable fields on the left, the diffusion coefficient, D, is estimated from the Gaussian distribution. The numerical fields give the relative error in percentage between the simulated value and the theoretical one. The vale of D is obtained by calculating the full width at half maximum (FWHM) of the distribution at every lapse of time dt. The FWHM which is related with the standard deviation, σ, by:

$$\mathrm{FWHM} = 2\sqrt{2\ln 2}\,\sigma \tag{1.1}$$

and the diffusion coefficient can be obtained using σ (see eq. 2.16). This value is only an approximation and will only indicate a general view or the Brownian motion.

In the graph below, a similar approach is made regarding the **Boltzmann statistics** (see section 2.4) related to the optical forces (restoring forces). The distribution of the positions of the particles, if the optical force is connected, will also follow a Gaussian distribution (see chapter 2). This graph only appears if we have activated the optical trap in the "Input" panel. From this Gaussian distribution, we can obtain the stiffness of the restoring forces, k_x and k_y, (see section 2.3) by calculating the σ value for the distribution and using eq. 2.20. An example of Gaussian distributions when the optical trap is connected can be seen in Fig. 2.1.

The graph at the left-bottom shows the random motion of one of the disks in the xy plane. The input field allows to change the particle whose trajectory is shown in this graph. If an optical force is activated, we will obtain a elliptical trajectory (circular if $k_x = k_y$). In the right-bottom figure, the software plots the number of particles inside the simulation box. Fig. 2.1 in chapter 2) shows an example of these graphs plot during a run.

1.4 Saving and reading data.

The aiming of BDL is to work like a virtual experiment and, with the exception of the illustrative graphs in the "Graphics" panel, no additional statistical calculations are made using the disks' positions. The XY positions of the particles can be stored in a text file for further analysis using standard algorithms already published in the literature.

The name and complete path of the file which contains the information of every

step of the simulation (XY positions, label of the particles, time, etc) is indicated in the "Input" panel. By default, this file will be saved in the default user's directory as "FILENAME.txt". This data file contains the positions of the particles in the xy plane in units of the screen. To convert them to real units (microns), we need to multiply them by the calibration factor, shown in the "Input" panel or in the input file which contains all the input parameters of the simulation. Therefore, the calibration of the xy values has to be done in the subsequent analysis of the data, since it is not performed by BDL, as would happen in an actual video-microscopy experiment.

```
X       Y       time    image
1.2179681       1.543363        0.001   1
1.8735429       1.761465        0.001   1
1.0191354       1.0680419       0.001   1
1.220979        1.5471739       0.002   2
1.8749844       1.7601007       0.002   2
1.0178947       1.0698857       0.002   2
1.2240225       1.5439817       0.003   3
1.8744522       1.7579651       0.003   3
1.0172138       1.073257        0.003   3
```

Figure 1.5. Text file with disks' positions and image label, without individuals labels

When saving the XY positions of the disks in the text file, all data is moved to positive values to avoid negative values. This is done to avoid some problems which may appear in certain statistical calculations. The box according to the saved data in the text file is now situated between 0 and 2, in units of the software screen.

The saved text file contains a third column which indicates the "image" of the particle. Fig. 1.5 shows a small example of three particles for three different times (images), from 0.001 s to 0.003 s. The columns which are shown are x coordinate (in "screen units", therefore it needs calibration), y coordinate ("screen units"), t in seconds, and the "image" label. In the "Input" panel, we can choose to add a fifth column in the text file. This column will contain numbers labeling every disk, something useful for the analysis of the disks' trajectories. The output file is shown in fig. 1.6, where it appears a fifth column with a number assigned to every particle.

An important point, regarding the time value stored in this file, is that we can save the data only of the disks' jumps in a certain lapse of time, as happens in an actual time-lapsed microscopy experiment. By default, every data for the internal time step, dt, of the simulation is stored, but we can select instead to save only the data corresponding to a defined temporal step Δt (in every case $dt \leq \Delta t$).

The file containing the data of the position of the disks can be read using standard algorithms developed using Interactive Data Language (IDL; ITT Visual Information Solutions, Boulder, CO) available for the analysis of video-microscopy images of particles [5]. Using that open-source software, it is possible to analyze the data contained in the XY file to obtain the relevant statistical information from the particle's positions.

X	Y	time	image	label	
1.0919802		0.8022397	0.001	1	1
1.046047		0.2729417	0.001	1	2
1.8725507		0.32902852	0.001	1	3
1.0910589		0.8038743	0.002	2	1
1.0433153		0.27200577	0.002	2	2
1.8732527		0.33087787	0.002	2	3
1.0897889		0.80303144	0.003	3	1
1.0433853		0.27356777	0.003	3	2
1.8735228		0.3301462	0.003	3	3

Figure 1.6. Text file with disks' positions and image label but with individuals labels

The second txt file saved by the sofware is named as "input_FILENAME.txt", and it is saved along with the described text data file containing the XY positions of the disks ("FILENAME.txt"). These two files are saved in the user's hard disk by pressing the "save" button in the "Simulations" panel, or automatically by enabling the radio button named *Save data automatically?* located in the "Input" panel (this last option will save the stored data every 10 lines in the text file and will liberate memory space used in the simulations).

1.4.1 Read files

The input file has a defined structure (see fig. 1.3), allowing BDL to read the file for recovering input parameters previously used. To read only the input file, we have to proceed as shown in fig. 1.7, where the XY file path is left empty. Press "Enter" in both fields and then press the button "Read" and the information contained in the input file will be added to the "Input panel" of BDL.

Figure 1.7. "Input panel": it reads only the input file.

If we use the free running mode (not saving data) we can run BDL during hours without saving data. Pressing the pause button, we will be able to press the button "Write last step" and the software will save a file automatically named as "last_step_FILENAME.txt". This file contains the positions of the disk in the very last step performed by BDL. This last step file (or a generic "FILENAME.txt") can be read by BDL by introducing the file path in the fields of the "Input panel", as indicated in fig. 1.8. It is mandatory to read an input file, which must be coherent with the XY file by providing both the same number or particles per step. Press "Enter" in both fields and then the button for reading the position of the particles. The saved position of the disks will be plot in "Simulations

panel".

Figure 1.8. "Input panel": it reads the input file and the xy file.

This methodology allows to run the model during a long period of time, save a final configuration, read it, and begin a new experiment using that previous configuration of the particles' positions as a initial configuration. For example, selecting an AO force and running BDL during several hours in the "not saving data" mode, we obtain a cluster of particles like the one shown in fig. 1.9. This configuration of particles can be saved by pressing the button "Write last step", reset the software, and then read the last step file (and input file) using the methodology explained above. Then, we can perform a new simulation (applying different interactions to the particles, saving the evolution of the position of the disks, etc) using this cluster as the initial configuration of the particles.

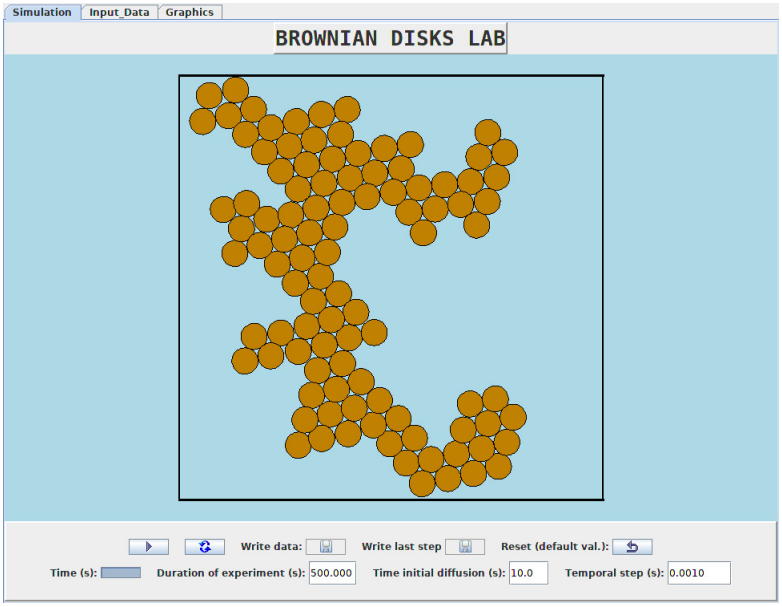

Figure 1.9. Cluster of particles obtained in the free running mode.

Chapter 2

THEORETICAL MODEL AND DISK INTERACTIONS.

BDL is based in Brownian dynamics simulations, which are a simplification of Stokesian dynamics, but neglecting hydrodynamic interactions (HI) between particles. We insert in the model different attractive potentials to inspect some attractive behavior observed experimentally, and a restoring force to simulate an external optical trap. Finally, we explain how to calculate the diffusion coefficient and the stiffness of the optical trap from the distribution of temporal jumps and positions.

2.1 Brownian dynamics simulations

The Langevin equation for a set of N spheres of radius a and mass m immersed in a medium of viscosity η and density ρ is

$$m\frac{d\mathbf{v}}{dt} = \mathbf{F}_{\mathrm{H}} + \mathbf{F}_{\mathrm{B}} + \mathbf{F}_{\mathrm{E}} + \mathbf{F}_{\mathrm{D}} \tag{2.1}$$

where $\mathbf{v}$ is the velocity vector, $\mathbf{F}_{\mathrm{H}}$ are the hydrodynamic forces, $\mathbf{F}_{\mathrm{B}}$ are the stochastic forces, $\mathbf{F}_{\mathrm{E}}$ are external forces over the particles in the fluid and $\mathbf{F}_{\mathrm{D}}$ are hard-disk forces that avoid the particles from overlapping. This last term is needed because we neglect full hydrodynamic interactions, $\dot{\mathbf{v}} \sim 0$, and therefore the hydrodynamic contribution is reduced to $\mathbf{F}_{\mathrm{H}} = -6\pi\eta a\,\mathbf{v}$, *i.e.*, the Stokes drag of an isolated particle. When using HI [6], the lubrication forces avoid the particles for overlapping. The great difficulty in Brownian dynamics simulations is the election of the inter-particle potential to prevent the superposition of the disks.

The interaction forces between the particle i with the particle j can be expressed by their effective interaction potential V:

$$\mathbf{F}_i = -\nabla \sum_{i \neq j} V(|\mathbf{r}_i - \mathbf{r}_j|)$$

where, from now on, we will call $r \equiv |\mathbf{r}_i - \mathbf{r}_j|$. Thus, the equations of movement are given by:

$$\frac{d\mathbf{r}}{dt} = \frac{1}{\xi}\left(\mathbf{F}_{\mathrm{E}} + \mathbf{F}_{\mathrm{B}} + \mathbf{F}_{\mathrm{D}}\right) \tag{2.2}$$

where $\xi = 1/M = 6\pi\eta a$, being ξ the Stokes's friction coefficient and M the mobility of the particles. The Brownian or stochastic force is characterized by $\langle \mathbf{F}_{\mathrm{B}} \rangle = 0$ and by:

$$\langle \mathbf{F}_{\mathrm{B}}(0)\, \mathbf{F}_{\mathrm{B}}(t) \rangle = 2k_B T \xi\, \boldsymbol{\delta}(t) \tag{2.3}$$

where $\boldsymbol{\delta}(t)$ is the unit tensor. To impose that condition, we use a random vector $\mathbf{n}$ whose values are contained in the interval $[-1, 1]$ and set:

$$\mathbf{F}_{\mathrm{B}} = \sqrt{2dk_B T\xi/dt}\, \mathbf{n} \tag{2.4}$$

where here d is the dimension, and dt will be the time step in the simulations [7].

2.1.1 Contact forces

A simple expression for the separation potential consists in a excluding volume force which depends of the separation r between two disks as:

$$\mathbf{F}_{\mathrm{D}}(r) = A\, \exp\left\{-B\left(\frac{r}{2a} - 1\right)\right\}\, \hat{\mathbf{r}} \tag{2.5}$$

where $\hat{\mathbf{r}} \equiv (\mathbf{r}_i - \mathbf{r}_j)/r$ is the unitary vector in the radial direction for particles i and j with $i \neq j$. In our simulations, $A = 1/2N$ where N is the total number of particles, and $B = 10$ by default. This exclusion volume force depends on the temporal step, and it is needed at least $dt = 0.01$ s for working.

The forces between disks in contact are better obtained through potentials like:

$$V_{Dn}(r) = \epsilon \left(\frac{d}{r}\right)^n \tag{2.6}$$

where $\epsilon \equiv k_B T$ and d is the diameter of the disks. Different elections of n, such as $n = 36, 12, 6...$ provide different "soft" sphere fluids, being $n \to \infty$ a step function, *i.e.*, hard-sphere (disk, in 2D) fluid. For further details, see [8, 9, 10, 11]. When this kind of potential is used, the time step has to be smaller when n grows. If time step were too large, the two particles could overlap and an intense excluding force would appear, which provides non-physical results. Forces from this potential can be written as:

$$\mathbf{F}_{Dn}(r) = A_n\, n\, d^n\, \frac{\epsilon}{r^{n+1}}\, \hat{\mathbf{r}} \tag{2.7}$$

where A_n is a constant. An alternative method was developed by Heyes and Melrose [8] by introducing an effective force between the two particles i and j which are in contact along the direction α. Thus, we have:

$$\mathbf{F}_{DH\alpha i} = -\sum_j k_H\, (\mathbf{r}_{ij} - 2a)\, \frac{r_{\alpha ij}}{\mathbf{r}_{ij}} \tag{2.8}$$

where $r_{\alpha ij}$ is the α component of $\mathbf{r}_{ij}$ and k_H is a constant. This effective force simply separates the two particles along the line which joins their centers. These forces can be increased or decreased in BDL by changing the parameter which appears in the "Input panel".

2.2 Attractive Potentials

In this version of the software, we have included the two attractive interactions used to explain observations of attractive interactions on suspensions of confined equally-charged micro-spheres, which do not agree with the standard theory of colloidal interactions, the DLVO theory [12, 13]. The observations by Tata et al. regarding the existence of an attractive minimum in the electrostatic potential [14] can be fitted by a phenomenological pair potential proposed first by Grier and Han [13]:

$$U_{\mathrm{GH}}(r) = k_B T\, Z^*\, \lambda_B \left(Z^* \frac{e^{-\kappa r}}{r} - 4q\, \frac{e^{-\kappa r/2}}{r} \right) \tag{2.9}$$

where $k_B T = 1/\beta$, $Z^* = Z \exp(\kappa a)/(1+\kappa a)$, Z is the effective charge on the particle, q an effective space charge number, $\lambda_B = \beta e_0^2/(4\pi\epsilon)$ is the Bjerrum length (usually 0.717 nm in water) for a medium of dielectric constant ϵ at temperature T, e_0 is the elementary charge, κ^{-1} is the Debye-Hückel screening length which is given by $\kappa^2 = 4\pi\lambda_B n_0$ in an electrolyte having a concentration of n_0 monovalent ions. In ref. [15], the authors observed a minimum in the potential using silica particles with a separation between walls of $h = 9\,\mu$m. The fit of eq. (2.9) to the experimental data on that case provides $Z \sim 800$, $\kappa^{-1} \sim 150\ \mu$m, $q \sim 13$ [13].

Asakura and Oosawa [16] and later Vrij [17] propose a colloid-colloid attractive potential based in the depletion attraction between colloids immersed in non-adsorbing ideal polymer coils. The AO potential for two spheres in a gas of smaller polymer spheres of radius R_g is:

$$U_{\mathrm{AO}}(r) = \Pi_p \frac{4\pi}{3} R_{\mathrm{AO}}^3 \left(\frac{\lambda}{\lambda - 1}\right)^3 \left[1 - \frac{3}{2}\frac{r}{a\lambda} + \frac{1}{2}\left(\frac{r}{a\lambda}\right)^3\right] \tag{2.10}$$

Eq.(2.10) is only applicable if $a \leq r \leq a + 2R_{\mathrm{AO}}$. Here $\lambda = 1 + 2R_{\mathrm{AO}}/a$, and R_{AO} is a parameter that is usually taken as $R_{\mathrm{AO}} = R_g$, but more exact expressions have been proposed [18]. This attraction generates a depletion zone of polymer among particles, varying the osmotic pressure of the polymers near the colloid surface.

In previous works, we have analyzed the rheological properties of the liquid surrounding the disks under the influence of these potentials [2]. These mechanical properties are expressed by means of the complex modulus $G^*(\omega) = G'(\omega) + iG''(\omega)$, where G' and G'' are the elastic and loss modulus [19], and where ω is the frequency of a oscillatory strain. The complex modulus $G^*(\omega)$ is obtained from the mean-squared displacement of the particles through the Generalized Stokes-Einstein relation (GSER) [20] together with the Mason's approximation [21]. The analysis of particle's trajectories, statistical and microrheological calculations can be made by means of open-source routines [22].

2.3 Optical trapping

The previous expressions are forces between pair of particles (applied only to close neighborhoods). We include in BDL an external force applied to all the disks in the experiment, a restoring force defined by the potential:

$$U_{\mathrm{k}}(r) = \frac{1}{2}\, k_r \,(r - r_0)^2 \tag{2.11}$$

where k_r is the stiffness or spring constant of the force $F(r) = -k_r \times r$, r can be one of the two coordinates, x or y, and r_0 is the initial position of the particle, x_0 or y_0. Therefore, we define two spring constants, k_x and k_y, expressed in typical units of μN/m, as it can be seen in the "Input Data Panel".

The potential defined by eq. (2.11) is the harmonic approximation for a trapping potential generated by optical tweezers setups [23]. The assumption that the potential generated by the focused laser beam of the optical tweezers under the probe is harmonic is valid for the central region of the potential well [24, 25].

2.4 Gaussian and Boltzmann statistics.

The simulations return data files that are processed using microrheology standard algorithms. The most simple expression of microrheology consists in deducing the value of the fluid viscosity η by means of the Stokes-Einstein relation for self-diffusion coefficient

of a particle with radius a [26]:

$$D_0 = \frac{k_B\, T}{6\pi\eta a} \tag{2.12}$$

where η is the viscosity of the medium. The mean square fluctuation (MSD) of a particle in d dimensions in a purely viscous material is given by:

$$\langle \Delta r^2(t) \rangle = \langle (r(t+\tau) - r(t))^2 \rangle = 2dD_0\, t \tag{2.13}$$

The diffusion coefficient can be obtained by:

$$D_0 = \frac{1}{2d}\, \frac{d\langle \Delta r^2(t) \rangle}{dt} \tag{2.14}$$

The MSD is not calculated in this software, but can be obtained using the data file of disk's positions generated by BDL. For a free diffusion of the particles, we can obtain the diffusion coefficient from the probability distribution of the disk's jumps, Δr, for a fixed time lapse τ for d dimensions:

$$\rho_D(\Delta r, \tau) = \frac{1}{(4\pi D_0 \tau)^{d/2}}\, \exp\left(-\frac{\Delta r^2}{4D_0\tau}\right) \tag{2.15}$$

where the one-dimensional variance of the Gaussian distribution, σ^2 is equal to:

$$\sigma^2 = 2D_0\tau \tag{2.16}$$

By knowing the jumps distribution and the temporal step between jumps, it is possible to estimate the diffusion coefficient, as explained in section 1.3.3. This expression, ρ_D, is also called "Gaussian propagator" or "van Hove autocorrelation function" [27]. If we assume there is no difference in the diffusion process by using τ (lapse time) or t (absolute time), the moments of the distribution can be obtained from the propagator by:

$$\delta r^n(t) \equiv \langle |\Delta r(t)|^n \rangle = \int |\Delta r|^n\, \rho_D(\Delta r, t)\, d^d(\Delta r)$$

By integrating that expression ($d = 1$) using eq. (2.15):

$$\delta r^n(t) = \frac{1}{\sqrt{\pi}} \Gamma\left(\frac{n+1}{2}\right)\, (4D_0 t)^{n/2}$$

Then, the MSD is calculated using $n = 2$:

$$\delta r^2(t) \equiv \mathrm{MSD}(t) = 2D_0 t \tag{2.17}$$

which is analogous to eq. (2.14) for non-anomalous diffusion.

Similarly, it is possible to analyze the optical force potential through Boltzmann statistics [28] on the position of the Brownian particle. Theoretically, the average motion of the laser-trapped dielectric particle in the Rayleigh limit can be described by means of the probability density distribution $\rho(x, t)$ which obeys the Fokker-Planck equation [29, 30]. This equation can be written [31] in a simple one-dimensional form as:

$$\frac{d\rho(x)}{dx} = \frac{1}{k_B T} F(x)\rho(x) \tag{2.18}$$

If we use an optical trap modeled as a restoring force with spring constant k as eq. (2.11), the solution of eq. (2.18) is a Gaussian function on x:

$$\rho_k(r) = \left(\frac{k}{2\pi k_B T}\right)^{d/2} \exp\left(-\frac{1}{2}\frac{k_r r^2}{k_B T}\right) \tag{2.19}$$

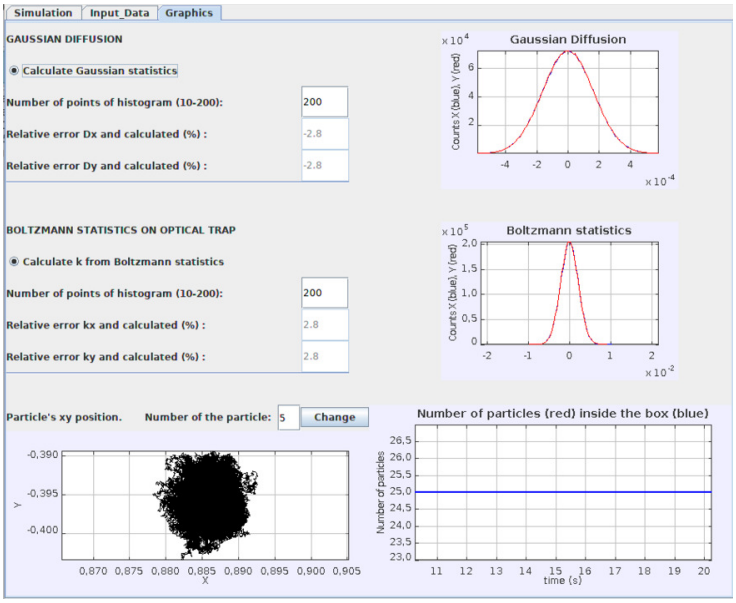

Figure 2.1. Screen capture of "Graphics panel" illustrating Gaussian and Boltzmann statistics.

In fact, the spatial probability density $\rho_k(x)$ is related to the conservative potential $E(x)$ in thermal equilibrium by means of the Boltzmann distribution $\rho_k(x) = Z^{-1}\, e^{-E(x)/k_B T}$, where Z is the partition function and, for this case, $E(x) = \frac{1}{2}kx^2$. Again, the variance of the distribution allows to obtain the stiffness of the trap by using:

$$\sigma^2 = \frac{k_B T}{k} \tag{2.20}$$

In the general case of stochastic motion confined by an harmonic potential (when the optical trap is connected), the solution of the Fokker-Plank equation leads to a more complicated distribution. This distribution includes a memory effect on the jumps of the disks [29]. However, for short values of τ, the distribution is identical to eq. (2.15), whereas for long values of the temporal step, it approximates to eq. (2.19).

In fig. 2.1, we show an example of the calculation of these distributions during the motion of the disks. The stiffness and diffusion coefficient, k and D_0, are obtained through the variance of the perfectly Gaussian distributions, ρ_D and ρ_k. Both numerical values are obtained within relative errors less than 3%. These distributions are calculated using the internal temporal step, dt, which is at least 10^{-4} s when using the optical trap. This assures that the Gaussian distribution of jumps is not linked to the influence of the optical trap and that D_0 can be calculated according to eq. (2.15).

Chapter 3

CODE DETAILS.

In this chapter, we will explain some details about the Java code in BDL. The source code can be read using the file named:

`Brownian_disks_lab_v1.1.ejs`

---

which has to be opened using "Easy Java/Javascript Simulations". Please, download and install the **versión 5.3** of the software[1] to run the "Brownian_disks_lab_v1.1.ejs" file provided as source code. Read the EjsS's manual for further instructions about this program[2].

By opening the file with EjsS, it is easy to understand the basic structure of the BDL's code. Basically, the simulations of the disks work through the following equations that appear in the Model section of EjsS:

$$\frac{dx[i]}{dt} = \text{force}(i, t, x, y, \text{diameter}, \text{diameter}, vx, vy, \text{true})$$
$$\frac{dy[i]}{dt} = \text{force}(i, t, x, y, \text{diameter}, \text{diameter}, vx, vy, \text{false})$$

according to the Langevin equation (eq. 2.2). Here, the method "force()" contains all the forces defined in the last chapter, but divided by the Stokes's friction coefficient $\xi = 1/M = 6\pi\eta a$. These equations are numerically resolved by EjsS using a Euler-Richardson algorithm without the need of extra implementations. Other algorithms are available, but they do not provide different results for these simulations.

All the forces are easily implemented as Java methods, with the only issue of taking into account the "screen" units of the simulation. An additional explanation is required concerning the Brownian force, defined as a white noise according to eq. (2.4). The random numbers are generated using the Box-Muller transformation [32] by the following expressions for $\mathbf{n}$:

---

[1] `http://www.um.es/fem/EjsSWiki/Main/Download`

[2] `https://www.um.es/fem/EjsWiki/uploads/Download/EjsS_Manual.pdf`

$$n_x = \sqrt{-2\ln(a)}\cos(2\pi b)$$
$$n_y = \sqrt{-2\ln(a)}\sin(2\pi b)$$

where a and b are random numbers generated in Java using `double a = Math.random()` and `double b = Math.random()`. This method allows to obtain random variables with a Gaussian distribution from uniformly distributed random variables.

3.1 Using EjsS code without EjsS

Running the software using the jar file generated from EjsS executes by default the graphic user interface (GUI) of BDL. We explain in this section how to modify the BDL code without EjsS, and how to run the simulations without the use of a GUI, only by executing scripts. Using this methodology, we can execute the simulations in remote servers and clusters, by creating jar files which are part of the input parameters. This can be useful for executing at the same time simulations with different input parameters. Before that, we have to learn how to execute the code outside of EjsS.

3.1.1 EjsS code in Eclipse

Eclipse[3] is an integrated development environment (IDE) for computer programming, which is most used for Java developers. We can run our EjsS code in Eclipse using the following instructions:

- First, we have to locate the Java code of our project in EjsS. This can be done, after running the EjsS file in EjsS, by looking in the directory named "output" inside EjsS workspace installation. There, we will find a folder whose name ends by "_pkg" and contains the Java code of our EjsS project.
- This code should be placed in the project previously created by Eclipse, inside the "src" directory of the Eclipse project. We must follow the directory structure defined in the "output" folder of EjsS and copy the whole directory into the Eclipse project, because the name of the directory will be the name assigned to the package on every "class" file.
- Then, we have to copy all the jar files contained in the "bin" directory of the EjsS directory to the "jars" directory inside the Eclipse project. Afterwards, we have

[3] `http://www.eclipse.org/`. The last version, "Helios", gives issues for compiling this software. The version "Oxygen" (from 2018) works better and can be downloaded here: `https://www.eclipse.org/downloads/packages/release/oxygen/3a/eclipse-ide-java-ee-developers`

import the jars files into the Eclipse project by selecting them in “Project/Properties/Java Buid Path/Libraries/Add External Jars”.

- Finally, we must create a new class for running the simulation. Fig. 3.1 shows a screen capture of Eclipse running the BDL code generated by EjsS. We created a class named “BDL_main” which contains a “main” method that executes the simulation.

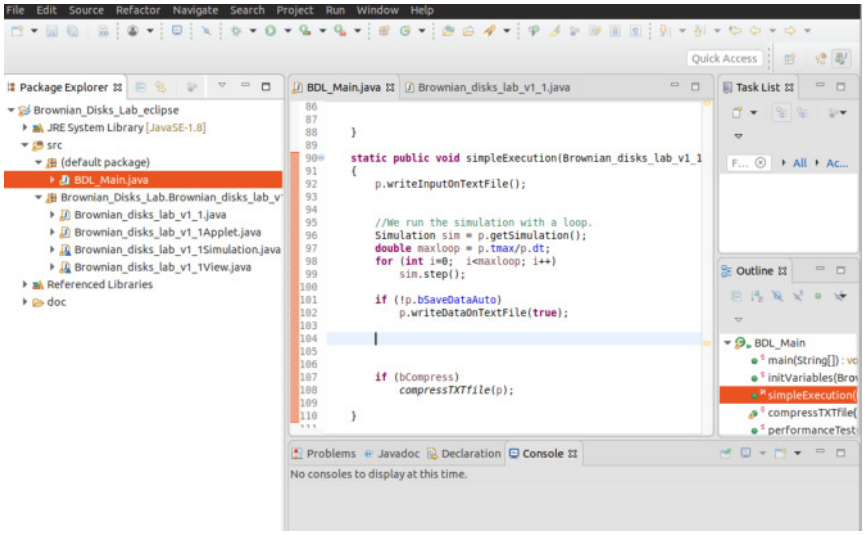

Figure 3.1. Screen capture of the Eclipse IDE (Oxygen version) running the EjsS code.

The simulation itself is loaded inside the Eclipse project by calling the object named “Brownian_disk_lab_v1_1’, which is defined in one of the classes imported from EjsS. This object contains all the methods and variables used in EjsS, so we can define all the input values before running the simulation. For example, if we want to change temperature and radius of the particle, we have to do the following:

```
//Create object p of simulations
Brownian_disks_lab_v1_1 p = new
    Brownian_disks_lab_v1_1(null,null,null,null,(String[])null,false);
//Diameter of the particle in microns
p.d_real=1;
//Temperature
p.T = 273+21;
```

The attributes (methods) for the object `Brownian_disks_lab_v1_1 p` are now part of that Object and can be used in other statics methods of the class “BDL_main” (see Fig.

3.1). If we want to run the simulation, the code should be the following one (Fig. 3.1)

```
//We run the simulation with a loop.
Simulation sim = p.getSimulation();
double maxloop = p.tmax/p.dt;
for (int i=0; i<maxloop; i++)
  sim.step();
```

If we do not want to update simulation view, we have to include the following line:

```
 //We can see the particles moving. If false the view is freezed.
p._setUpdateView(false);
```

This last instruction freezes the view, but it does not make it disappear. This view is always needed when we use EjsS, therefore, if we want to run these simulations in a Linux-based remote server without graphical interface, we can add a virtual frame-buffer for displaying all graphical operations in memory without showing any screen output. Xvfb[4] allows such an operation.

Finally, we export the code in a jar file using Eclipse. The command for running the jar file in a Linux system should be something like:

Listing 3.1: bash version

```
xvfb-run -a java -Xms2000m -Xmx2000m -XX:-UseGCOverheadLimit -jar
    Brownian_disks_lab_v1.1.jar
```

This command allows to run BDL including the input parameters without the need of a graphical interface. In any case, it is always advisable to edit BDL code inside EjsS, and then copy the "_pkg" folder to the Eclipse project.

[4] https://www.x.org/archive/X11R7.6/doc/man/man1/Xvfb.1.xhtml

BIBLIOGRAPHY

[1] F. Esquembre, "Easy java simulations: a software tool to create scientific simulations in java," *Comput. Phys. Commun.* **156**, pp. 199–204, 2004.

[2] P. Domínguez-García, "Microrheological consequences of attractive colloid-colloid potentials in a two-dimensional brownian fluid," *Europhys. J. E. Soft. Matter.* **35**, p. 73, 2012.

[3] P. Domínguez-García and M. A. Rubio, "Single and multi-particle passive microrheology of low-density fluids using sedimented microspheres," *Appl. Phys. Lett.* **102**, p. 074101, 2013.

[4] M. Pancorbo, M. A. Rubio, and P. Domínguez-García, "Brownian dynamics simulations to explore experimental microsphere diffusion with optical tweezers," *Procedia Comp. Sci.* **108**, pp. 166 – 174, 2017.

[5] J. C. Crocker and D. G. Grier, "Methods of digital video microscopy for colloidal studies," *J. Colloid Interface Sci.* **179**, pp. 298–310, 1996.

[6] H. Hess and R. Klein, "Generalizated hydrodynamics of systems of brownian particles," *Appl. Phys.* **32**(2), pp. 173–283, 1983.

[7] R. G. Larson, *The Structure and Rheology of Complex Fluids*, Oxford University Press, NewYork, 1999.

[8] D. M. Heyes and J. R. Melrose, "Brownian dynamics simulations of model hard-sphere suspensions," *J. Non-Newtonian Fluid Mech.* **46**, pp. 1–28, 1993.

[9] D. M. Heyes, "Shear thinning of dense suspensions modelled by brownian dynamics," *Phys. Lett. A* **132**(8-9), pp. 399–402, 1988.

[10] J. Dzubiella, H. Lawen, and C. N. Likos, "Depletion foreces in nonequilibrium," *Phys. Rev. Lett.* **91**(24), p. 248301, 2003.

[11] B. Lin, D. Valley, M. Meron, B. Cui, H. M. Ho, and S. A. Rice, "The quasi-one-dimensional colloid fluid revisited," *J. Phys. Chem. B* **113**, pp. 12742–13751, 2009.

[12] A. E. Larsen and D. G. Grier, "Like-charge attraction in metastable colloidal crystallites," *Nature* **385**(16), pp. 230–233, 1997.

[13] D. G. Grier and Y. Han, "Anomalous interactions in confined charge-stabilized colloid," *J. Phys. Condens. Matter* **16**, pp. 4145–4157, 2004.

[14] B. V. R. Tata, P. S. Mohanty, and M. C. Valsakumar, "Bound pairs: Direct evidence for long-range attraction between like-charged colloids," *Solid State Commun.* **147**, pp. 360–365, 2008.

[15] Y. Han and D. G. Grier, "Confinement-induced colloidal attractions in equilibrium," *Phys. Rev. Lett.* **91**(3), p. 038302, 2003.

[16] S. Asakura and F. Oosawa, "Interaction between particles suspended in solutions of macromolecules," *J. Poly. Sci.* **33**, pp. 183–192, 1958.

[17] A. Vrij, "Polymers at interfaces and the interactions in colloidal dispersions," *Pure and App. Chem.* **48**, pp. 471–483, 1976.

[18] A. A. Louis, P. G. Bolhuis, E. J. Meijer, and J. P. Hansen, "Polymer induced depletion potentials in polymer-colloid mixtures," *J. Chem. Phys.* **117**(4), pp. 1893–1907, 2002.

[19] T. D. Squires and T. G. Mason, "Fluid mechanics of microrheology," *Annu. Rev. Fluid Mech.* **42**, pp. 413–438, 2010.

[20] T. G. Mason, K. Ganesan, J. H. van Zanten, D. Wirtz, and S. C. Kuo, "Particle tracking microrheology of complex fluids," *Phys. Rev. Lett.* **79**, pp. 3282–3285, 1997.

[21] T. G. Mason, "Estimating the viscoelastic moduli of complex fluid using the generalized stokes-einstein equation," *Rheol. Acta* **39**, pp. 371–378, 2000.

[22] J. C. Crocker and B. D. Hoffman, "Multiple particle tracking and two-point microrheology in cells," *Methods in Cell Biology* **83**, pp. 141–178, 2007.

[23] A. Ashkin, "Applications of laser radiation pressure," *Science* **210**(4474), pp. 1081–1088, 1980.

[24] K. Svoboda and S. M. Block, "Biological applications of optical forces.," *Annu. Rev. Biophys. Biomol. Struct.* **23**, pp. 247–85, 1994.

[25] A. C. Richardson, S. N. S. Reihani, and L. B. Oddershede, "Non-harmonic potential of a single beam optical trap," *Opt. Express* **16**(20), pp. 15709–15717, 2008.

[26] A. Einstein, "On the theory of brownian motion," *Ann. Phys.* **19**, pp. 371–381, 1906.

[27] F. Hofling and T. Franosch, "Anomalous transport in the crowded world of biological cells," *Rep. Prog. Phys.* **76**(4), p. 046602, 2013.

[28] E.-L. Florin, A. Pralle, E. H. K. Stelzer, and J. K. H. Hörber, "Photonic force microscope calibration by thermal noise analysis," *Appl. Phys. A* **66**, pp. S75–S78, 1998.

[29] H. H. Risken, *The Fokker-Planck equation: methods of solution and applications*, Springer series in synergetics, Springer-Verlag, Berlin, New York, 1984.

[30] Y. Harada and T. Asakura, "Radiation forces on a dielectric sphere in the rayleigh scattering regime," *Opt. Commun.* **124**(5-6), pp. 529–541, 1996.

[31] T. J. Davis, "Brownian diffusion of nano-particles in optical traps," *Opt. Express* **15**(5), pp. 2702–2712, 2007.

[32] G. E. P. Box and M. E. Muller, "A note on the generation of random normal deviates," *Ann. Math. Statist.* **29**(2), pp. 610–611, 1958.