12-2013

Development of a Coupled Fluid and Colloidall Particle Transport Model

Scott Ripplinger
Utah State University

DEVELOPMENT OF A COUPLED FLUID AND COLLOIDAL PARTICLE TRANSPORT MODEL

by

Scott Ripplinger

A thesis submitted in partial fulfillment
of the requirements for the degree

of

MASTER OF SCIENCE

in

Mechanical Engineering

Approved:

Dr. Heng Ban Major Professor	Dr. Byard Wood Committee Member
Dr. Christine Hailey Committee Member	Dr. Mark R. McLellan Vice President for Research and Dean of the School of Graduate Studies

UTAH STATE UNIVERSITY
Logan, Utah

2013

Abstract

Development of a Coupled Fluid and Colloidal Particle Transport Model

by

Scott Ripplinger, Master of Science

Utah State University, 2013

Major Professor: Dr. Heng Ban
Department: Mechanical and Aerospace Engineering

Colloidal systems have received various analytic treatments, though many have involved a statistical average of properties over time and few have provided detail about what happens with individual particles. The statistical models provide answers as to what is happening in a system, but the level of detail in particle tracking simulation provides answers as to why it is happening. This work provides an integrated lagrangian particle tracking and Computational Fluid Dynamics (CFD) solution for tracking micro-sized solid particles within a fluid flow. This is accomplished using tools provided in the OpenFOAM toolkit as a basis for further development. This simulation code is tested for functionality by looking at the individual elements being simulated, including drag, buoyancy, collision with walls, collision between particles, and wall attraction due to colloidal forces. The overall stability of the code is observed by simulating a case with many thousands of particles. Particle adsorption onto the microchannel surface is observed and compared with data from other studies. This provides a versatile code for simulating colloidal suspensions of solid particles in a liquid. By using this code as a basis, additional solvers can be developed for a variety of applications which involve solid micro particles being transported in liquids.

(79 pages)

Public Abstract

Development of a Coupled Fluid and Colloidal Particle Transport Model

by

Scott Ripplinger, Master of Science

Utah State University, 2013

Major Professor: Dr. Heng Ban
Department: Mechanical and Aerospace Engineering

A colloidal system usually refers to when very small particles are suspended within a solution. The study of these systems encompasses a variety of cases including bacteria in ground water, blood cells and platelets in blood plasma, and river silt transport. Taking a look at these kinds of systems using computer simulation can provide a great deal of insight into how they work. Most approaches to date do not look at the details of the system, however, and are specific to given system. In this study a program called OpenFOAM is used as a basis to build a computer simulation tool that is flexible and that provides a detailed look at what is happening with all of the particles within the colloidal solution. This code is run through a series of tests to verify its usefulness.

To my wife Kimberly, and children, Jonathan, Sarah, and Abigail...

Acknowledgments

This work was made possible by support from Dr. Thomas Hauser and Dr. Heng Ban, and by computational resources made available by the Space and Missile Defence Command Simulation Center and the Utah State University Division of Research Computing.

Scott Ripplinger

Contents

List of Tables

List of Figures

Nomenclature

Δt	simulation time step
ϵ	wall normal elasticity term
κ	Debye-Huckel length parameter
λ	wavelength
$\left\langle \Delta \mathbf{r}^B \Delta \mathbf{r}^B \right\rangle$	Gaussian distribution for Brownian motion
$\hat{\mathbf{e}}_{ij}$	unit vector pointing from particle i to j
$\hat{\mathbf{n}}_k$	unit vector normal to wall
$\mathbf{c}_1$	particle pre-collision velocity
$\mathbf{c}_1^*$	particle post-collision velocity
$\mathbf{c}_m$	velocity of center of mass of system of colliding particles
$\mathbf{D}_{ij}$	mutual diffusivity of particles
$\mathbf{F_b}$	particle bouyancy vector
$\mathbf{F_{colloidal}}$	colloidal force vector
$\mathbf{F_D}$	particle drag force vector
$\mathbf{F_g}$	gravity force vector
$\mathbf{F_T}$	total force vector acting on a particle
$\mathbf{F_w}$	particle weight vector
$\mathbf{F}_{vdw}$	Van der Waals force vector
$\mathbf{g}$	gravitational acceleration vector
$\mathbf{u_i^t}$	velocity vector of particle i at time t
$\mathbf{U}$	fluid velocity vector
μ	fluid static viscosity
μ	wall tangential elasticity term
ν	fluid kinematic viscosity
ψ_i	surface potential of surface i
ρ	fluid density

ρ_p particle density

ε permittivity of fluid medium

A Hamaker constant

a_i radius of particle i

C_D coefficient of drag

D_c particle drag term

d_p particle diameter

e charge of an electron

F_h hydrophobic force

$F_{colloidal}$ total colloidal force

f_j fluid source term in Navier-Stokes equation

$F_{sph-sph}$ force on a sphere due to a nearby sphere

$F_{sph-wall}$ force on a sphere due to a nearby wall

h separation distance between surfaces

k Boltzmann constant

m_p particle mass

P fluid pressure normalized by density

R mean radius of curvature

Re Reynolds number

T absolute temperature

t time

u_j fluid velocity vector component in the j direction

V_p particle volume

V_S interaction energy between surfaces

x_j position vector component in the j direction

y_i reduced surface potential of surface i

z valency of the counter ions

Chapter 1

Introduction

Colloidal systems exist all around us in real life. A rain cloud is a colloidal system. So is quicksand, smoke, blood, and even milk. In each of these cases there are small particles or droplets suspended in a liquid or gaseous medium. These particles and droplets are small enough that the surface forces between them and any other surrounding surfaces become significant.

The American Heritage Dictionary defines a colloid as "a system in which finely divided particles, which are approximately 10 to 10,000 angstroms in size, are dispersed within a continuous medium in a manner that prevents them from being filtered easily or settled rapidly." The chemistry of the continuous medium and the dispersed particles determine the strength of the Van der Waals and Electric Double Layer effects. For particles of this size these effects can become significant and affect the way the system behaves.

There are two ways in which colloidal systems have been simulated. The first is by statistical modeling. Data is obtained experimentally and then a best matching function is applied. The constants used with this function must be fitted from the data. Unfortunately, this makes the approach somewhat inflexible. The equation and constants used may only be reused when simulating a system that is very similar to the original experiment from which the data was obtained. Yet for these particular types of systems, the statistically fit equations can yield reliable results quickly.

In addition to a limited scope, statistical models also fail to provide any more detail about how a system is working than the observed data upon which it is based. All those physical details are smoothed out by the statistically matched equations. The net effect can be modeled, but may not offer any insight into why the system behaves the way it does.

An alternate approach is to model the detailed physics of the system. Rather than

observing the general properties in bulk, individual colloidal particles can be tracked. This could then be used to extrapolate data on a larger scale. This type of simulation would allow for a great deal of flexibility as the physics are approached at a more refined level. Rather than being tied to a single type of system, different types of force models can be applied to the particles based on what is appropriate.

The scale between these two methods varies greatly. Statistical models will be much better suited for simulation of large systems. The physics based particle tracking models will give finer detail, but as the size and complexity of the system increase this approach becomes impractical. Domain setup would be too complex and the number of particles would become too great to be handled by modern computer memory and processing architectures.

Both of these methods have their place. In fact, both methods can be used together, the statistical model to obtain general answers and the particle tracking model to gain insight into why a colloidal system is behaving a certain way. Some of these details could conceivably then be used to further refine and tune the statistical model.

The majority of particle tracking colloid simulation performed so far has been with droplets. Applications have primarily been for simulation of fuel injection sprays and atmospheric clouds. Solid particle tracking software has been considerably more limited. Most implementations so far have been on a small scale, used primarily as a proof of concept. None have provided a basis for a comprehensive colloidal solver which can be adapted and enhanced. This is the purpose of this thesis.

Some previous work by Gschaider and Hauser and Allen [1] has been done in developing solid particle tracking software coupled with a Computational Fluid Dynamics (CFD) solver. Gschaider linked together libraries available in the OpenFOAM open source CFD toolkit to create a solver for solid particles entrained in an inviscid laminar fluid. Hauser & Allen took this work one step further by adding colloidal forces to the particles. This work will be further enhanced by adding additional colloidal models, more vigorous coupling between fluid and particle momentum, and usage of updated OpenFOAM libraries. The code will be set up to enable easy insertion of additional force models for the particles and adaptation

of different OpenFOAM fluid solvers.

Chapter 2

Review of Relevant Works

Colloidal systems have been a part of a number of studies. Particle transport through porous media is observed by McDowell-Boyer et al. [2]. Jenny and Smith [3] studied the colloidal aspects of clay pan formation. Aggregation and coalescence of particulates due to colloidal forces was observed by Ho and Higuchi [4]. Adamczyk et al. [5] conducted experiments using impinging jet flows containing colloidal particles to study the effects of flow intensity on particle adsorption kinetics. Each of these cases show how studies have focused on the behavior of colloidal particles in a system under various circumstances by experimentation and observation.

Simple particulate systems are not the only applicable areas. Microbiology transport also becomes an area in which colloidal effects need to be considered. Bolster et al. [6] provide work in which the colloidal particles are actually single-celled organisms, *E. coli* and *C. jejuni*. They make the point that water is often tested for the presence of *E. coli* as an indicator organism for *C. jejuni*, which is the actual organism of interest as a pathogen. Their study looks at the degree to which *E. coli* and *C. jejuni* transport properties through a porous media actually differ. Hornberger et al. [7] provide another observation of bacteria transport through aquifer sands. Bradford and Bettehar [8] studied the transport and deposition behavior of *Cryptosporidium parvum* oocysts in water travelling through sand columns of various grain sizes. In addition to these studies of bacteria transport in ground water, numerous studies have been made on the flow of blood cells, including work by Schmid-Shoenbein et al. [9]. In each of these cases the cells suspended in either water or blood plasma constitute a colloidal system.

The examples given so far of studies based on experimental observation. There are several possible approaches to simulating a colloidal system. Many of the methods developed

and used to solve these problems today use statistical models. They ignore the details of the individual particles within a system and focus instead on the macroscopic aspects of importance to the study. Tan et al. [10] go an extra step by including simulated data to compare with their experimental results for aquifer bacteria transport. Another model is offered by Barton and Ford [11]. And again, Duffy et al. [12] perform calculations for residence times of bacteria in a porous media. These models use criteria such as the porosity of the soil and fluid properties and velocity to determine the mean rate at which bacteria is transported. These methods, and others like them, have provided reliable results for their specific applications, as demonstrated by comparison with experimental results, with those results being limited to the macroscopic scope of the problem.

More detailed methods of simulating micro-particle flows use a Lagrangian approach for particle tracking and an Eulerian fluid solution. The particles are tracked individually, each experiencing a range of forces. A simple Lagrangian particle tracking algorithm, called *icoLagrangianFoam*, was implemented into the OpenFOAM CFD package by Bernhard Gschaider. This solver utilizes the existing *icoFoam* incompressible fluid solver and injects particles which are carried by fluid drag forces. This method does not include surface forces. Work by Hauser and Allen [1] resulted in the *icoColloidalFoam* solver which augmented *icoLagrangianFoam* to include Van der Waals forces, electric double layer forces, inter-particle collisions, and hydrodynamic shear.

Theoretical text on colloidal and surface interactions are provided by Israelachvili [13], and Israelachvili and McGuiggan [14]. The first provides an in depth look at molecular and surface forces, deriving them using thermodynamic and chemical theories. The second is more a qualitative description of the pertinent forces. Both discuss DLVO theory as well as solvation forces (hydration and hydrophobic forces for water).

Simulation of colloidal interactions was performed by Marshall [15] using discrete-element methods. Particle velocity and rotation is tracked due to fluid drag, particle collision and van der Waals forces. Simulation is performed for a periodic micro channel flow and includes wall adhesion. A similar study is performed by Unni and Yang [16] where parti-

cle tracking is done using the Langevin equation. Hydrodynamic, Van der Waals, electric double-layer and gravity forces are all incorporated into the Langevin equation, along with a term for Brownian motion. The simulation is validated using surface deposition data from a micro channel flow experiment. Simulation using Lagrangian particle tracking was also done by Longest et al. [17] to simulate the transport of blood cells in a non-Newtonian carrier fluid. Emphasis is placed on hydrodynamic effects in a non-Newtonian fluid and colloidal forces are not considered.

The effects of the hydration and hydrophobic forces are least understood when compared to other colloidal forces. The necessity of this force was determined when it was shown that the DLVO theory did not account for forces present between particles and surfaces with water contact angles outside the range of 15-60 degrees [18]. Simulation of particle transport involving highly hydrophobic or hydrophylic surfaces would necessitate the accounting of these forces.

While various empirical formulae have been developed to describe each of these phenomena, a detailed look at the behavior of individual particles may be useful in understanding them better. Empirical formulae generally require constants which are determined based on experimental data for a very specific case. While these are useful for common applications, finding the necessary constants needed to simulate additional cases may be cumbersome. In fact, the empirical model may not even apply to all cases. A particle based model needs physical properties for the surfaces involved and the fluid that separates them. These properties are often tabulated though they may also be calculated based on the chemistry of the surfaces and fluid being considered. Thus a particle model may be adapted to new cases more easily than an empirical model.

Empirical models typically require less computation time in a CFD code when compared to a coupled Lagrangian particle tracking algorithm. In such an algorithm, individual particles are identified by their diameter, position and velocity. The position and velocity of each particle is updated each time step based on the forces acting on these particles. The particle field also obtains data from the fluid field to calculate hydrodynamic forces on each

Table 2.1: Comparison of models included in works cited.

	Marshall	Unni & Yang	Gschaider	Hauser & Allen	Ripplinger
Particle translation	x	x	x	x	x
Particle rotation	x				
Particle-wall collisions	x	x	x	x	x
Particle-particle collisions	x	x		x	x
Particle-wall colloidal		x		x	x
Particle-particle colloidal					
Fluid drag	x	x	x	x	x
Fluid-particle momentum coupling					x
Brownian motion		x			
Open source (OpenFOAM)			x	x	x

particle. The particles within the field also reference each other and the surrounding walls in order to determine the surface forces between them. The resulting computational load increases exponentially as particles are added to the simulation.

The force models used for particles have been studied for some time, but they have rarely been implemented on a large scale to track individual particles within a flow. But with computational power being greater than ever such a task has become more feasible. While still being more computationally expensive, a particle tracking model with several thousands of particles is not unreasonable.

Chapter 3

Objective and Scope

The purpose of this thesis is to provide the basis for a detailed colloidal transport solver which can be expanded to serve many different purposes. Using OpenFOAM as a foundation will allow others to further augment this code to suit their purposes. Potential opportunities exist to build simulators for studies in river soil transport or blood platelet transport, among other things. OpenFOAM provides many tools which can be combined with the colloidal transport model, and the model itself can also be altered to include additional force models and interactions.

The main objectives are as follows:

- Produce a colloidal transport model that is compatible with OpenFOAM and integrated with a basic fluid solver that has the following capabilities:
  - Includes various models for the Van der Waals force as provided by Czarnecky, Schenkel and Kitchener, and Gregory for a particle near a wall
  - Includes models for the Electric Double Layer force as provided by Gregory for a particle near a wall
  - Accounts for wall collisions and inter-particle collisions
  - Provides a drag model that is coupled with the fluid solver
  - Provides a rudimentary injection model
- Test the functionality of each subroutine in the model
- Compare results from simulation of particle adsorption onto surfaces in a microchannel with those obtained by Unni and Yang

The Van der Waals and Electric Double Layer forces will become active when a particle gets close to a wall surface. Multiple models will be selectable in order to demonstrate the differences between them. Wall collisions are already incorporated into OpenFOAM, but a particle-particle collision model is also developed. A basic sphere drag model is provided with OpenFOAM as well, but the drag effects are not coupled with the fluid solver, so this is also implemented here. The injection model developed is simple and intended to generate particle flow for a microchannel demonstration simulation.

With the completed simulation tool, each of the elements above have been tested as independently as possible. The drag model is tested with a single particle far from any wall surfaces. The particle is placed in free-fall and the terminal velocity determined and compared with expected values. A particle aimed at a wall, with colloidal forces turned off, and two particles aimed at each other is used to test wall and particle collisions. A single sphere is then allowed to approach a wall under the force of gravity until the colloidal forces take effect using each Van der Waals and Electric Double Layer force model provided. Finally, a larger scale demonstration using many thousands of particles flowing through a micro-channel is used to demonstrate the overall functionality of the simulation tool with a high particle count, and to observe rates of particle adsorption onto a surface.

This tool provides a unique approach to solving the colloidal transport problem by offering broad applicability and a platform for further development. By adapting this tool to the various fluid solvers provided in OpenFOAM capabilities such as turbulence and multiphase fluid flow can be easily added. The colloidal transport tool itself can easily be augmented to provide different force models and other attributes which may prove useful for a particular application.

Chapter 4

Theory

In order to simulate a colloidal system, the physical interactions at play must be understood and appropriate models identified. These models must account for the behavior of the fluid medium, individual particles, and the interaction between particles and the fluid, each other, and the surroundings. An overview of the models and equations used in the simulations is given here, along with some scientific background on how these models are derived. Some additional models which are not used at this time, but which may have application in future versions of the developed code, are also presented here.

4.1 Governing equations

The principal conditions to be satisfied by a Newtonian fluid as it flows are described by the Navier-Stokes equations. In the case of an incompressible fluid with a viscosity assumed to be constant, Currie [19] offers the following reduced set of equations:

$$\rho\frac{\partial u_j}{\partial t} + \rho u_k\frac{\partial u_j}{\partial x_k} = -\frac{\partial p}{\partial x_j} + \mu\frac{\partial^2 u_j}{\partial x_i \partial x_j} + \rho f_j \tag{4.1}$$

The subscripts on the velocities u_j and directions x_j given here follow the tensor notation standard. As stated already, the fluid density ρ and static viscosity μ are treated as constants. The f_j term is a source term used to account for any body forces applied to a fluid element. If this equation is normalized by the fluid density it becomes:

$$\frac{\partial u_j}{\partial t} + u_k\frac{\partial u_j}{\partial x_k} = -\frac{\partial P}{\partial x_j} + \nu\frac{\partial^2 u_j}{\partial x_i \partial x_j} + f_j \tag{4.2}$$

where P is the pressure divided by the density. In this form, rather than requiring two fluid terms, the density and static viscosity, only one term, the kinematic viscosity ν is needed.

This represents a set of three equations, one for each directional component. There are, however, four unkowns with the three velocity vector components u_j and the density normalized pressure P. The velocity components can be solved for if the pressure field is first treated as a constant. In order to solve for the pressure field the Pressure Implicit Splitting of Operators (PISO) algorithm [20] is used.

The forces acting on an individual particle considered here include drag, gravity and colloidal forces:

$$\mathbf{F}_T = \mathbf{F}_D + \mathbf{F}_g + \mathbf{F}_{colloidal}. \tag{4.3}$$

The algorithm used in the OpenFOAM *solidParticle* class involves only drag calculations. Particle velocities are updated for each time step as:

$$\mathbf{u}_i^{t+dt} = \frac{\mathbf{u}_i^t + D_c \mathbf{U} dt}{1 + D_c dt}, \tag{4.4}$$

where $\mathbf{u}_i^t$ is the velocity vector of the i^{th} particle at time t, $\mathbf{U}$ is the local fluid velocity vector and D_c is a drag term which will be defined later. This equation can be rearranged as:

$$\mathbf{u}_i^{t+dt} = \mathbf{u}_i^t + D_c \left(\mathbf{U} - \mathbf{u}_i^{t+dt} \right) dt. \tag{4.5}$$

It can now be seen that this is a simple update of the particle velocity using the acceleration due to drag on the particle. The calculation of the drag acceleration merely makes use of an updated particle velocity in this instance. The additional forces are easily added to this equation as:

$$\mathbf{u}_i^{t+dt} = \mathbf{u}_i^t + \left(D_c \left(\mathbf{U} - \mathbf{u}_i^{t+dt} \right) + \frac{\mathbf{F}_g + \mathbf{F}_{colloidal}}{m_p} \right) dt, \tag{4.6}$$

where m_p is the mass of the particle. This equation can then be changed back to its original form:

$$\mathbf{u}_i^{t+dt} = \frac{\mathbf{u}_i^t + \left(D_c \mathbf{U} + \mathbf{F}_g / m_p + \mathbf{F}_{colloidal} / m_p \right) dt}{1 + D_c dt}. \tag{4.7}$$

It can be seen that adding any additional forces is now a simple task by dividing by the particle mass and adding it to the terms within the parantheses. Should a force which is

dependent on an updated velocity, such as the drag here, be desired, then it must involve a term similar to D_c with units of $1/s$. This term would be multiplied by dt and added to the terms in the denominator.

4.2 Drag and buoyancy

The injected particles are strongly affected by the fluid in which they are carried. Particles experience buoyancy effects based on their density relative to that of the fluid. They also experience drag which results from a discrepancy between particle and fluid velocities. The drag force, in particular, can be dominating where the fluid velocity is high.

The net gravitational force represents the difference between the weight and the buoyancy force as:

$$\mathbf{F}_g = \mathbf{F}_w - \mathbf{F}_b = \rho_p V_p \mathbf{g} - \rho V_p \mathbf{g} = (\rho_p - \rho) V_p \mathbf{g}, \tag{4.8}$$

where ρ_p is the density of the particle, ρ is the density of the fluid, V_p is the volume of the particle, and $\mathbf{g}$ is the acceleration of gravity vector. A spherical particle may therefore have a net gravitational force of:

$$\mathbf{F}_g = \frac{\pi}{6} d_p^3 (\rho_p - \rho) \mathbf{g}, \tag{4.9}$$

with d_p being the particle diameter.

The drag force described in the previus section can be defined as:

$$\mathbf{F}_D = D_c \left(\mathbf{U} - \mathbf{u} \right) m_p, \tag{4.10}$$

where $\mathbf{U}$ is the fluid velocity, $\mathbf{u}$ is the particle velocity and m_p is the particle mass. The D_c term is defined as:

$$D_c = 24 \frac{\nu}{d_p} \cdot 0.15 Re^{0.687} \cdot \frac{3}{4} \frac{\rho}{d_p \rho_p}, \tag{4.11}$$

where ν and ρ are the fluid kinematic viscosity and density, respectively, ρ_p is the particle density and Re is the relative Reynolds number calculated as:

$$Re = \frac{|\mathbf{U} - \mathbf{u}| d_p}{\nu}. \tag{4.12}$$

In the calculation of D_c the $0.15Re^{0.687}$ term can be approximated as Re in the case that $Re < 0.01$. For this condition these expressions are equivalent to the traditional form of the drag equation with a drag coefficient:

$$\mathbf{F}_D = \frac{\pi}{8} d_p^2 \rho C_D \left|\mathbf{U} - \mathbf{u}\right| \left(\mathbf{U} - \mathbf{u}\right), \tag{4.13}$$

with:

$$C_D = \frac{24}{Re}. \tag{4.14}$$

For the application being considered now, the f_j term in Equation 4.2 can account for the change in momentum due to particles within the fluid element. The calculation of this term may be calculated based on Newton's Third Law: For every action there is an equal and opposite reaction. The force which the fluid imparts on the particle is the drag force, so the change in particle momentum due to drag is then accounted for in the fluid with f_j. This momentum change is summed for all particles within a fluid cell and then divided by the density and fluid cell volume as:

$$f_j = \frac{\sum m_p \left(u_j - U_j\right) D_c}{\rho V}, \tag{4.15}$$

where u_j and U_j are the velocity components of the particle and local fluid velocities, respectively.

4.3 Surface forces

The primary forces, beyond any hydrodynamic or gravitational forces, acting on colloidal particles are referred to as *surface* forces. These forces largely fall under the domain of electrostatic forces resulting from molecular arrangement on the particle and wall surfaces, as well as in the fluid medium. These forces are further subdivided into different categories, the most important of which are van der Waals, electrostatic double layer, and solvation forces. Solvation forces are most active when particularly hydrophobic or hydrophylic surfaces are submerged in an aqueous solution. Electrostatic double layer forces are present

when surfaces are separated by a solution containing electrolytes. Van der Waals forces are always present and typically represent a strong attractive force between surfaces, depending on the surface properties and fluid medium. These last two forces, the electrostatic double layer force and the van der Waals force, are the main components making up the colloidal surface forces.

4.3.1 Van der Waals force

The dominating interactions between molecules include chemical bonds, metallic bonds, and ionic bonds. Beyond these there are many other interactions which exist on a weaker level, but often over longer ranges. These interactions include dipole interactions where there exists an electrical polarity in a molecule, creating an electric field that can influence neighboring molecules. These, and some other forces, often determine properties of some materials, such as surface tension, boiling and melting points, etc. They also compose one of the major forces responsible for molecules aggregating in a medium, the van der Waals force. Israelachvili [13] points out that the Van der Waals interaction between molecules is composed of three distinct forces: the *induction* force, the *orientation* force and the *dispersion* force.

The *induction interaction*, also known as the *Debye interaction*, results from a polar molecule in proximity to a non-polar molecule. As the polar molecule approaches the non-polar molecule it *induces* a dipole. The permanent and induced dipoles generate an electric field around each other creating an attractive force. The *orientation interaction*, or *Keesom interaction*, is similar with the exception that both molecules are permanent dipoles.

The *dispersion force* may be thought to arise from the fact that even in non-polar atoms which have a time averaged dipole of zero, at any given instant there exists a finite dipole. This instantaneous dipole generates an electric field that polarizes any nearby neutral atom, inducing a dipole moment in it. The resulting interaction between the two dipoles gives rise to an instantaneous attractive force between the two atoms, and the time average of this force is finite.

All three of these have interaction energies which vary with the inverse sixth power of the separation distance. The sum of these make up the *van der Waals interaction*:

$$w_{vdw}(r) = -C/r^6 = -\left[C_{ind} + C_{orient} + C_{disp}\right]/r^6. \tag{4.16}$$

With the exception of small highly polar molecules, such as water, the dispersion forces tend to dominate over the induced and dipole-dipole interactions.

The van der Waals force is not only applicable to small molecules and atoms. Large molecules and micro-particles are also significantly influenced by this force. Force models for larger bodies may be obtained by assuming the molecular van der Waals force is of the form $w_{vdw}(r) = -C/r^6$, and the further assumption of *additivity*. Integrating these forces between all the molecules in two spheres gives the simplified unretarded Hamaker expression [21]:

$$V_s = -\frac{Aa_ia_j}{6(a_i + a_j)h}, \tag{4.17}$$

$$A = \pi^2 C\rho_i\rho_j, \tag{4.18}$$

where V_s is the interaction energy between the surfaces, h is the minimum separation between the spheres, a_i and a_j are the sphere radii, A is the Hamaker constant, and ρ_i and ρ_j are the molecule number densities of the two materials. For the purposes of this study the Hamaker constant will not be calculated, but rather empirical values found in literature will be used.

Gregory [22] shows that the unretarded Hamaker expression is insufficient except for very close separation distances, i.e. $h \ll d$. Several different expressions are compared to exact solutions for cases involving semi-infinite parallel plates, sphere-sphere, and sphere-plate interactions. Gregory offers the following expression for the interaction energy between two unequal spheres of radii a_i and a_j:

$$V_s = -\frac{Aa_ia_j}{6(a_i + a_j)h}\left[1 - \frac{bh}{\lambda}\ln\left(1 + \frac{\lambda}{bh}\right)\right], \tag{4.19}$$

with a value of $b = 5.32$. The negative of the derivative of the interaction energy with respect to separation distance is taken to find the force:

$$F_{sph-sph} = -\frac{\partial V_s}{\partial h} = -\frac{Aa_i a_j}{6(a_i + a_j)h^2} \frac{1}{5.32\frac{h}{\lambda} + 1}. \tag{4.20}$$

The sphere-wall interaction force can be obtained by setting $a_j = \infty$:

$$F_{sph-wall} = \lim_{a_j \to \infty} F_{sph-sph} = -\frac{Aa_i}{6h^2} \frac{1}{5.32\frac{h}{\lambda} + 1}. \tag{4.21}$$

The combined van der Waals force on a particle due to all other particles and walls is then:

$$\mathbf{F}_{vdw} = \sum_{k=1}^{N_w} -\frac{A_k a_i}{6h_k^2} \frac{1}{5.32\frac{h_k}{\lambda} + 1} \hat{\mathbf{n}}_k + \sum_{j=1}^{N_p} -\frac{A_j a_i a_j}{6\left(a_i + a_j\right) h_j^2} \frac{1}{5.32\frac{h_j}{\lambda} + 1} \hat{\mathbf{e}}_{ij}. \tag{4.22}$$

Gregory also compares an interpolated model offered by Schenkel and Kitchener [23] for unequal spheres:

$$V_s = -\frac{Aa_i a_j}{6\left(a_i + a_j\right) h} \frac{1}{11.12\frac{h}{\lambda} + 1}, \tag{4.23}$$

as well as a modified version for a sphere near a semi-infinite flat plate:

$$V_{sp} = -\frac{Aa}{6h} \frac{1}{14\frac{h}{\lambda} + 1}. \tag{4.24}$$

These two equations can then be differentiated with respect to h as before to find the force on the particle:

$$\begin{aligned} \mathbf{F}_{vdw} \quad = \quad & \sum_{k=1}^{N_w} -\frac{A_k a_i}{6h_k^2} \left[\frac{1}{14\frac{h}{\lambda} + 1} + \frac{14h}{\lambda\left(14\frac{h}{\lambda} + 1\right)^2} \right] \hat{\mathbf{n}}_k \\ & + \sum_{j=1}^{N_p} -\frac{A_j a_i a_j}{6\left(a_i + a_j\right) h_j^2} \left[\frac{1}{11.12\frac{h}{\lambda} + 1} + \frac{11.12h}{\lambda\left(11.12\frac{h}{\lambda} + 1\right)^2} \right] \hat{\mathbf{e}}_{ij}. \end{aligned} \tag{4.25}$$

An exclusive sphere to plate model by Czarnecki [24] is also reviewed. The energy of interaction is give as:

$$\begin{aligned} V_{sp} &= A\left[\frac{2.45\lambda}{60\pi}\left(\frac{h-a}{h^2}-\frac{h+3a}{(h+2a)^2}\right)\right. \\ &\quad -\frac{2.17\lambda^2}{720\pi^2}\left(\frac{h-2a}{h^3}-\frac{h+4a}{(h+2a)^3}\right) \\ &\quad \left.+\frac{0.59\lambda^3}{5040\pi^3}\left(\frac{h-3a}{h^4}-\frac{h+5a}{(h+2a)^4}\right)\right]. \end{aligned} \tag{4.26}$$

Differentiation with respect to h gives:

$$\begin{aligned} \mathbf{F}_{vdw} &= \sum_{k=1}^{N_w} -A\left[\frac{2.45\lambda}{60\pi}\left(\frac{2a-h}{h^3}+\frac{4a+h}{(2a+h)^3}\right)\right. \\ &\quad -\frac{2.17\lambda^2}{720\pi^2}\left(\frac{6a-2h}{h^4}+\frac{10a+2h}{(2a+h)^4}\right) \\ &\quad \left.+\frac{0.59\lambda^3}{5040\pi^3}\left(\frac{12a-3h}{h^5}+\frac{18a+3h}{(2a+h)^5}\right)\right]\hat{\mathbf{n}}_k. \end{aligned} \tag{4.27}$$

A comparison of various sphere-wall models are shown in Figure 4.1. While the models of Gregory and of Schenkel and Kitchener are comparable, with Schenkel and Kitchener being a slightly stronger force, the Czarnecki model is distinctly different, beginning to display higher attractive forces at closer separation distances. As discussed by Gregory, the Czarnecki model has a restriction of $h > \lambda/4\pi$ for valid results. At separations above this value of about 8 nm the Czarnecki model gives results which are even more accurate than the alternatives. Conversely, the Gregory and the Schenkel and Kitchener models are valid only under the restrictions of $0 < h < \lambda/\pi$ and $h \ll a$. Specifically, Gregory states that these two expressions are accurate for separations up to 10% of the particle radius. If the Czarnecki model is to be used for separations greater than 8 nm, then either the Gregory or the Schenkel and Kitchener models can only be used for particles with diameters of 160 nm or greater before the Czarnecki model is no longer able to bridge the gap. For the purposes of this study, however, the mean particle diameters studied are the order of 1 μm, well above

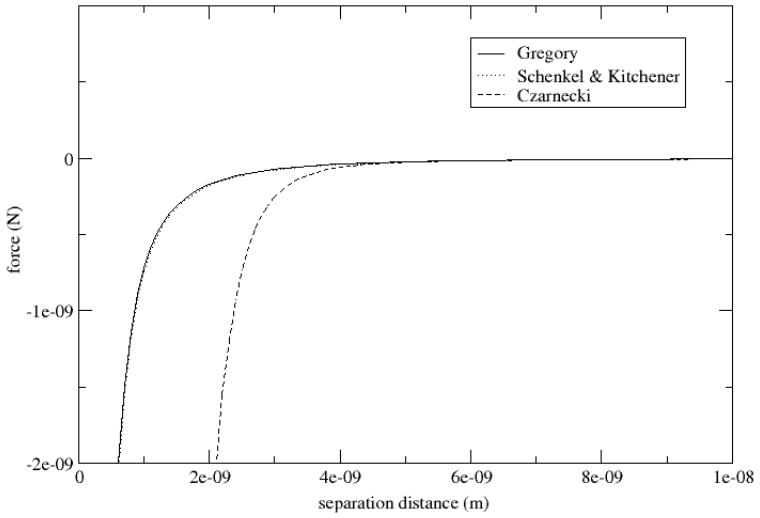

Fig. 4.1: A comparison of Van der Waals force models between a $1\mu m$ polystyrene sphere and a flat silica surface separated by a distance x. The intermediate medium is a NaCl aqueous solution of molality 0.01 M and a pH of 7. The solid line represents Eq 4.21, the dotted line represents the wall portion of Eq 4.25, and the dashed line represents Eq 4.27.

this limit.

For sphere-sphere interaction the Gregory and the Schenkel and Kitchener models perform reasonably well for larger particle diameters, such as 1 μm. For particles of diameters an order of magnitude smaller these models begin to deviate from exact solutions more rapidly with increasing separation distances. For these smaller particle sizes other models exist which may be superior, but these will not be considered at this time. While there still exists a restriction of separation distance being less than 10% of the particle radius, the forces on these particles begin to become relatively weak at such separations and deviations from exact solutions have less of an effect.

4.3.2 Electrostatic double layer force

The physical source of the electrostatic double layer force arises from the ionization of surfaces when submerged in a polar fluid, such as water. The ionized surface induces a

structure of counter ions known as the *electrical double layer* (EDL). This consists of the Stern layer and diffuse layer. The Stern layer is adjacent to the surface and includes ions bound to the surface. The diffuse layer is next to the Stern layer and contains freely moving ions. Israelacvili [13] points out that the origin of the repulsive force between two similarly charged surfaces in a solvent containing counter ions and electrolyte ions is osmotic, not electrostatic. When an initially uncharged surface is placed in water the surface groups dissociate and the counter ions leave the surface against the Coulombic force pulling them back. This is because the repulsive osmotic pressure between the counter ions dominates over the electrostatic attraction. Thus, when two such surfaces are brought together the osmotic pressure of one forces the counter ions onto the other against their entropic equilibrium. This results in a net repulsion between the two surfaces.

Gregory [25] offers expressions for the double layer interaction which are based on a constant surface potential assumption and a constant surface charge density assumption. He also points out that in practice neither case is likely to be true. Instead, they may be thought of as extremes, with the true solution lying somewhere between. A linear superposition approximation (LSA) of the two solutions is often used as a compromise between the two. The interaction energy between parallel plates is described by the linear Poisson-Boltzmann equation for a constant charge as:

$$V_s = \frac{nkT}{\kappa}\left[\left(y_1^2 + y_2^2\right)(\coth \kappa h - 1) + 2y_1 y_2 \frac{1}{\sinh \kappa h}\right], \tag{4.28}$$

where n is the number of ions per unit volume, k is the Boltzmann constant, T is the absolute temperature, κ is the Debye-Huckel reciprocal length parameter, and y_1 and y_2 are the reduced surface potentials. The reduced surface potentials are defined as:

$$y_i = \frac{ze\psi_i}{kT}, \tag{4.29}$$

with z representing the valency of the counter ions, e is the charge of an electron, and ψ_i represents the surface potentials. The Debye-Huckel reciprocal length parameter is then:

$$\kappa^2 = \frac{2e^2nz^2}{\varepsilon kT}, \tag{4.30}$$

where ε is the permittivity of the fluid medium.

Converting this into a force function between two unequal spheres is very simple when using the Deryagin method:

$$F_{edl}(h) = \frac{2\pi a_1 a_2}{a_1 + a_2} V_s(h), \tag{4.31}$$

where $V_s(h)$ is the energy of interaction per unit area of flat plates separated by a distance h. Combining Equation 4.31 with Equation 4.28 results in:

$$F_{sph-sph} = \frac{2\pi a_1 a_2}{a_1 + a_2} \frac{nkT}{\kappa} \left[\left(y_1^2 + y_2^2 \right) (\coth \kappa h - 1) + 2y_1 y_2 \frac{1}{\sinh \kappa h} \right]. \tag{4.32}$$

This can again be modified for a sphere near a flat plate by setting $a_2 \to \infty$:

$$F_{sph-wall} = 2\pi a_1 \frac{nkT}{\kappa} \left[\left(y_1^2 + y_2^2 \right) (\coth \kappa h - 1) + 2y_1 y_2 \frac{1}{\sinh \kappa h} \right]. \tag{4.33}$$

The combination of Equation 4.32 and Equation 4.33 gives the total force on the i^{th} particle as:

$$\begin{aligned} \mathbf{F}_{edl} \quad = \quad & 2\pi a_i \frac{nkT}{\kappa} \left\{ \sum_{k=1}^{N_w} \left[\left(y_i^2 + y_k^2 \right) (\coth \kappa h_k - 1) + 2y_i y_k \frac{1}{\sinh \kappa h_k} \right] \hat{\mathbf{n}}_k \right. \\ & \left. - \sum_{j=1}^{N_p} \frac{a_j}{a_i + a_j} \left[\left(y_i^2 + y_j^2 \right) (\coth \kappa h_j) + 2y_i y_j \frac{1}{\sinh \kappa h_j} \right] \hat{\mathbf{e}}_{ij} \right\}, \end{aligned} \tag{4.34}$$

where $\hat{\mathbf{n}}_k$ represents the unit normal vector on the k^{th} wall and $\hat{\mathbf{e}}_{ij}$ represents the unit vector pointing from particle i to particle j.

Gregory also provides the following expression for the interaction energy between parallel plates based on the linear Poisson-Boltzmann equation for a constant potential:

$$V_s = \frac{nkT}{\kappa}\left[\left(y_1^2 + y_2^2\right)(1 - \coth\kappa h) + 2y_1 y_2 \frac{1}{\sinh\kappa h}\right]. \tag{4.35}$$

Again, using the Deryagin method and combining for forces due to nearby plates and spheres we get:

$$\begin{aligned}\mathbf{F}_{edl} &= 2\pi a_i \frac{nkT}{\kappa}\left\{\sum_{k=1}^{N_w}\left[\left(y_i^2 + y_k^2\right)(1 - \coth\kappa h_k) + 2y_i y_k \frac{1}{\sinh\kappa h_k}\right]\hat{\mathbf{n}}_k \right. \\ &\left. - \sum_{j=1}^{N_p}\frac{a_j}{a_i + a_j}\left[\left(y_i^2 + y_j^2\right)(1 - \coth\kappa h_j) + 2y_i y_j \frac{1}{\sinh\kappa h_j}\right]\hat{\mathbf{e}}_{ij}\right\}. \end{aligned} \tag{4.36}$$

Equation 4.36 is also the same used by Hauser and Allen [1].

Lastly, Gregory gives the linear superposition approximation (LSA) expression for the interaction energy between parallel plates as:

$$V_s = \frac{64nkT}{\kappa}\gamma_1\gamma_2 \exp\left(-\kappa h\right), \tag{4.37}$$

where $\gamma_1 = \tanh\left(y_1/4\right)$ etc. Application of the Deryagin approximation and combination of force terms acting on a single particle results in:

$$\mathbf{F}_{edl} = 128\pi a_i \gamma_i \frac{nkT}{\kappa}\left[\sum_{k=1}^{N_w}\gamma_k \exp\left(-\kappa h_k\right)\hat{\mathbf{n}}_k - \sum_{j=1}^{N_p}\frac{\gamma_j a_j}{a_i + a_j}\exp\left(-\kappa h_j\right)\hat{\mathbf{e}}_{ij}\right]. \tag{4.38}$$

Unlike the Van der Waals models discussed in the previous section, the electric double-layer force models given here differ greatly. Gregory [25] offers a series of plots comparing these models to each other as well as to the exact solutions for the constant potential and constant charge cases. The surface potentials of two plates are varied to show the effect that this has on the various models. It is shown that the force model based on Equation 4.28 has poor agreement to the exact solution for the constant charge case. The force model based on Equation 4.35, however, agrees quite well with the exact solution for the constant

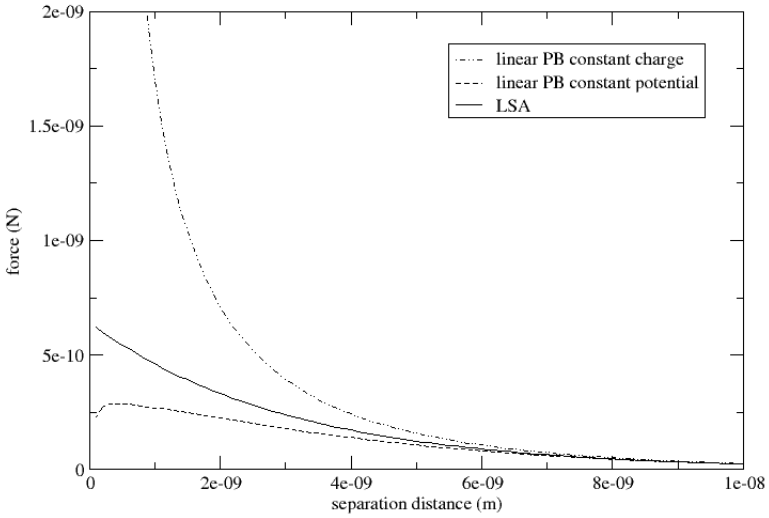

Fig. 4.2: A comparison of electric double-layer force models between a $1\mu m$ diameter polystyrene sphere and a flat silica surface separated by a distance x. The intermediate medium is a NaCl aqueous solution of molarity 0.01 M and a pH of 7. The linear Poisson-Boltzmann approximation for the constant surface charge (Eq. 4.34) and constant surface potential (Eq. 4.36) assumptions are compared with a linear superposition approximation (Eq. 4.38).

potential case. Figure 4.2 compares Equations 4.34, 4.36, and 4.38 for a polystyrene particle near a silica flat plate.

4.3.3 Solvation forces

In addition to the van der Waals and electrostatic double layer forces, some systems need to consider solvation forces. Israelachvili and McGuiggan [14] describe this force as resulting from the polar arrangement of fluid molecules between two surfaces in close proximity. For hydrophylic surfaces the water molecules arrange themselves in an antiparallel orientation near the surfaces, as shown in Figure 4.3A. This alignment results in a repulsive trend between the surfaces. Similarly, hydrophobic surfaces create a layer of parallel aligned water molecules which result in an attractive force, as shown in Figure 4.3B. These repulsive and attractive forces are also explained by a change in density of the fluid in the gap region

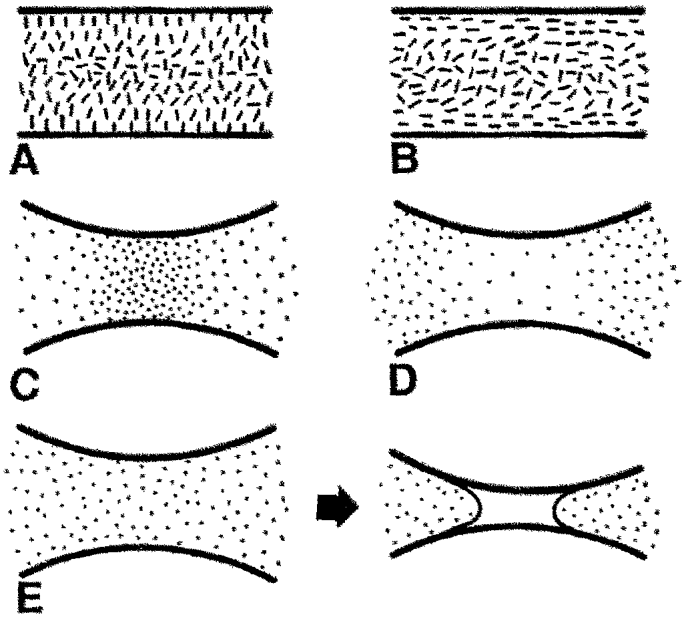

Fig. 4.3: Schematic illustration of how solvation forces arise

between the two surfaces. For a hydrophylic surface the density tends to increase creating a repulsive force, as shown in Figure 4.3C. A hydrophobic surface will repel water molecules, decreasing the density and encouraging an attractive force as in Figure 4.3D.

Solvation forces, like van der Waals forces, are oscillatory with distance. Also like van der Waals forces they have a monotonic component which can be mathematically modeled. These forces appear to decay exponentially with distance, having a characteristic decay length of 6 to 15 Å. This makes hydration and hydrophobic forces long ranged compared to van der Waals forces, allowing them to potentially dominate the other surface forces. Subramanian [18] has compiled a list of different force models for the hydrophobic effect :

$$F_h/R = C_0 exp(-H/D_0), \tag{4.39}$$

$$F_h/R = -K_{123}/6H^2, \tag{4.40}$$

$$F_h/R = C_1 exp(-H/D_1) + C_2 exp(-H/D_2). \tag{4.41}$$

In these equations the hydrophobic force F_h is normalized by the mean radius of curvature, R, and H represents the surface separation. Equation 4.39 is a simple exponential model where C_0 is a pre-exponential factor and D_0 is the decay length. Equation 4.40 expresses the hydrophobic force in a power law form, which is the same form as the van der Waals force. This form allows F_h and F_{vdw} to easily be added by integrating the K_{123} hydrophobic force constant with the Hamaker constant of the van der Waals force. A third option is to express $F_h$as a double-exponential force law, as in Equation 4.41.

Solvation forces are still somewhat of a mystery with research on the topic still actively proceeding. Empirical data for these equations is very limited for the time being. While there have been special cases observed where Van der Waals and EDL models do not predict colloidal forces accurately, these instances are limited to the regime of hydrophobic surface interaction.

For the purposes of this thesis hydrophobic forces will be neglected. This is not an entirely accurate assumption as the polystyrene latex particles Unni and Yang used in their experiment do have a hydrophobic tendency. Solvation forces should play a minor role in the attraction of the colloidal particles to a silica wall, but colloidal grouping between the particles may not be as accurately simulated without hydrophobic forces included. For this reason it would be wise to include the hydrophobic force in future studies.

4.3.4 Brownian motion

One of the principal factors considered by Unni and Yang [16] in their simulation is the effects of considering Brownian motion. This motion is a result of colloidal particles colliding with fluid molecules and has a Gaussian distribution with zero mean. The variance is a function of the mutual diffusivity of the particles, $\mathbf{D}_{ij}$. This distribution is defined as:

$$\left\langle \Delta \mathbf{r}^B \Delta \mathbf{r}^B \right\rangle = 2\mathbf{D}_{ij}\Delta t, \tag{4.42}$$

where Δt represents the simulation time step being used.

Brownian motion represents only small perturbations in colloidal movement and is often eclipsed when significant surface forces or hydrodynamic forces are present. The random nature of Brownian motion was a necessary aspect in the simulations performed by Unni and Yang. Simulations performed for the purpose of this thesis will, however, include a random particle injection algorithm, so inclusion of Brownian motion will not be necessary at this stage.

4.3.5 Combined effects

The total colloidal force is expressed as:

$$F_{colloidal} = F_{edl} + F_{vdw}. \tag{4.43}$$

If hydrophobic effects were also considered this would simply be:

$$F_{colloidal} = F_{edl} + F_{vdw} + F_h. \tag{4.44}$$

The DLVO theory of colloidal stability (named for Derjaguin and Landau [26], Verwey and Overbeek [27]) results in the following four conditions, as per Israelachvili [13]:

1. In the case that the surfaces in dilute electrolyte are highly charged they tend to repel each other. This repulsive force has a peak, normally at 1 to 4 nm distance, called the *energy barrier*.

2. The potential energy between surfaces in contact is known as the *primary minimum*. In the case of surfaces in a highly concentrated electrolyte their exists a *secondary minimum* outside of the energy barrier. The energy barrier is at times too high to overcome, so a colloidal system will often either sit in the secondary minimum or remain totally dispersed. A totally dispersed system is called *kinetically stable*.

3. When the surface potentials become weak the energy barrier becomes significantly lower. At a certain point the the energy barrier will disappear entirely and colloidal particles will rapidly attract and join with each other. This is called an *unstable* colloid.

4. The electric double layer force is completely dependant on the surface potentials. When these potentials become negligible the interaction between the surfaces becomes identical to the Van der Waals interaction and the surfaces attract each other.

The points here discuss the effects that the electrostatic double layer and van der Waals forces have for various electrolyte concentrations. In fact, the van der Waals force remains relatively constant for various electrolyte concentrations while the electrostatic double layer force tends to be stronger with the higher surface potentials associated with higher electrolyte concentrations.

When DLVO theory alone is considered, the van der Waals force typically represents an attractive force while the electric double-layer force is repulsive. The combination of these two is what results in the *primary minimum, secondary minimum* and *energy barrier*. Figure 4.4 shows the total colloidal force between a polystyrene particle near a silica surface in an electrolyte solution. The primary minimum exists at contact where Van der Waals forces dominate and cannot be seen in the Figure. The *energy barrier* can easily be seen at about 2.5 nm. The secondary minimum is further out and can be seen better in Figure 4.5at about 25 nm.

4.4 Collisions

The probability of particle collisions with walls or each other is limited due in large part to the colloidal energy barrier present in an electrolyte solution. Even in a pure aqueous solution in which Van der Waals forces dominate, the slow flow of the fluids considered and the relatively low number of particles injected make the likelihood of collision low. The enhanced treatment of this possibility, as seen in the works of Wu, et al [28] and Allahyarov et al [29] is therefore not included in this work. However, the current algorithm does incorporate

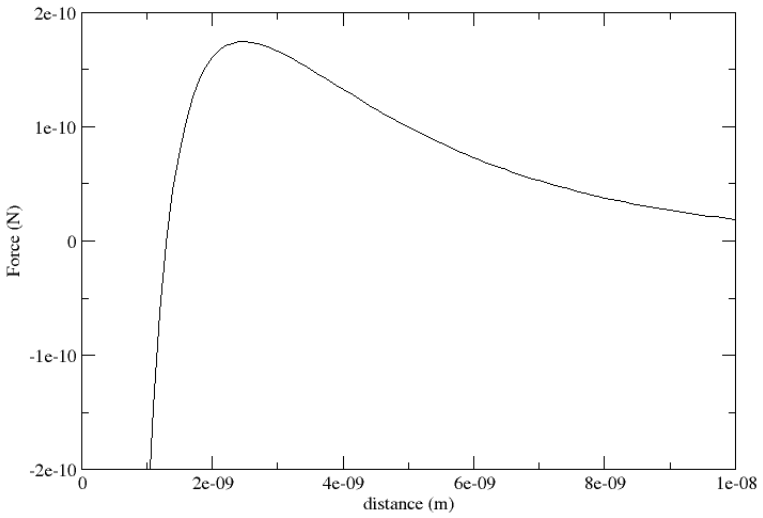

Fig. 4.4: The total DLVO force between a 1 μm diameter polystyrene-latex particle and a silica wall submerged in a 0.01 M NaCl aqueous solution. The Van der Waals contribution is computed using Eq. 4.21 and the EDL contribution is computed using Eq. 4.38.

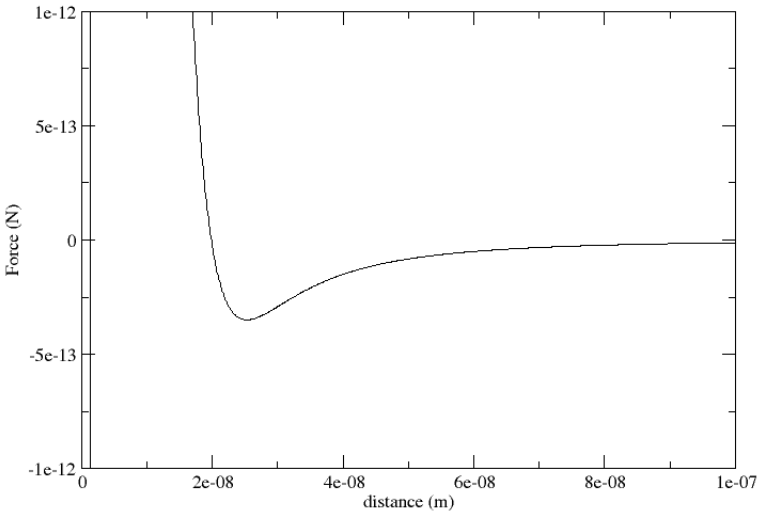

Fig. 4.5: A detail of 4.4 showing the secondary minimum.

a straight forward binary, elastic, hard sphere collision model for interparticle collision, as well as a simple inelastic collision model for wall collisions.

4.4.1 Wall collisions

The wall collision model used in this work is the same provided in the OpenFOAM 2.2.x solidParticle class. Particles are first flagged as to whether they are crossing a cell boundary. If that boundary is a wall patch then the resultant velocity of the particle is calculated using an inelastic collision model. First the magnitude of the velocity component normal to the wall is found as:

$$u_n = \mathbf{u} \cdot \hat{\mathbf{n}}, \tag{4.45}$$

where $\mathbf{u}$ is the particle velocity and $\hat{\mathbf{n}}$ is the unit normal vector to the wall. If the value of this is greater than zero than the particle is approaching the wall and must be reflected. The particle velocity is recalculated as:

$$\mathbf{u}_{post-collision} = \mathbf{u}_{pre-collision} - u_n \left(1 + \epsilon\right) \hat{\mathbf{n}}, \tag{4.46}$$

with ϵ being an elasticity term with $\epsilon = 1$ being fully elastic and $\epsilon = 0$ completely absorbing the impact energy of the particle.

In addition to energy being absorbed in the normal direction another term, μ, absorbs energy in the tangent direction. In this case a value of $\mu = 0$ indicates that no energy absorbed while $\mu = 1$ absorbs all tangent energy. Calculating the new velocity first requires the velocity component tangent to the wall, which is:

$$\mathbf{u}_t = \mathbf{u} - u_n \hat{\mathbf{n}}. \tag{4.47}$$

The adjusted particle velocity is thus calculated to be:

$$\mathbf{u}_{post-collision} = \mathbf{u}_{pre-collision} - \mu \mathbf{u}_t. \tag{4.48}$$

These adjusted velocities are both calculated whenever a particle diameter breaks the plain of a wall patch. Again, the normal velocity is only recalculated if the particle is found to still be approaching the wall.

4.4.2 Particle collisions

For hard sphere molecules, when the distance of closest approach between two molecules is less than:

$$d_{12} = 1/2(d_1 + d_2), \tag{4.49}$$

the two molecules will collide. In addition, to ensure that two spheres which have already collided in a previous time step do not have their velocities calculated again should they not escape from each others diameters a condition is imposed which negates implementation of the collision if the two spheres satisfy the range requirement but are moving away from each other.

Elastic collisions imply that there is no exchange of translational or internal energy, and are thus applicable only between collisions of monatomic species. For two molecules with mass and pre-collision velocities m_1, m_2 and $\mathbf{c}_1$, $\mathbf{c}_2$ respectively, conservation of linear momentum and energy require:

$$m_1\mathbf{c}_1 + m_2\mathbf{c}_2 = m_1\mathbf{c}_1^* + m_2\mathbf{c}_2^* = (m_1 + m_2)\mathbf{c}_m, \tag{4.50}$$

$$m_1\mathbf{c}_1^2 + m_2\mathbf{c}_2^2 = m_1\mathbf{c}_1^{*2} + m_2\mathbf{c}_2^{*2}, \tag{4.51}$$

where $\mathbf{c}_1$and $\mathbf{c}_1^*$ represent the pre and post collision velocities respectively, and $\mathbf{c}_m$ represents the velocity of the center of mass:

$$\mathbf{c}_m = \frac{m_1\mathbf{c}_1 + m_2\mathbf{c}_2}{m_1 + m_2}. \tag{4.52}$$

Equation 4.50 indicates that $\mathbf{c}_m$ does not vary with pre and post collision interactions. The pre and post relative velocities are expressed as:

$$\mathbf{c}_r = \mathbf{c}_1 - \mathbf{c}_2, \tag{4.53}$$

$$\mathbf{c}_r^* = \mathbf{c}_1^* - \mathbf{c}_2^*. \tag{4.54}$$

From Equations 4.50, 4.52, 4.53, and 4.54 we obtain:

$$\mathbf{c}_1 = \mathbf{c}_m + \frac{m_2}{m_1 + m_2}\mathbf{c}_r, \tag{4.55}$$

$$\mathbf{c}_2 = \mathbf{c}_m - \frac{m_1}{m_1 + m_2}\mathbf{c}_r, \tag{4.56}$$

$$\mathbf{c}_1^* = \mathbf{c}_m + \frac{m_2}{m_1 + m_2}\mathbf{c}_r^*, \tag{4.57}$$

$$\mathbf{c}_2^* = \mathbf{c}_m - \frac{m_1}{m_1 + m_2}\mathbf{c}_r^*. \tag{4.58}$$

From Equations 4.55 and 4.56 it is clear that in this center of mass frame of reference, the pre-collision velocities $\mathbf{c}_1 - \mathbf{c}_m$ and $\mathbf{c}_2 - \mathbf{c}_m$ are antiparallel, and further, if these molecules are point centers of force, then the force between them initially lies in the same plane as the pre-collision velocities. The anti-parallelism also occurs for the post-collision velocities of Equations 4.57 and 4.58. From these last equations we obtain:

$$m_1\mathbf{c}_1^2 + m_2\mathbf{c}_2^2 = (m_1 + m_2)\mathbf{c}_m^2 + m_r\mathbf{c}_r^2, \tag{4.59}$$

$$m_1\mathbf{c}_1^{*2} + m_2\mathbf{c}_2^{*2} = (m_1 + m_2)\mathbf{c}_m^2 + m_r\mathbf{c}_r^{*2}, \tag{4.60}$$

where the reduced mass is given as:

$$m_r = \frac{m_1 m_2}{m_1 + m_2}. \tag{4.61}$$

Therefore, from Equations 4.51, 4.59, and 4.60 we find that:

$$\mathbf{c}_r^* = \mathbf{c}_r. \tag{4.62}$$

We are now in a position to completely specify the post collision velocities, provided with the masses and pre-collision velocities. For a spherical coordinate system, with θ as the polar and ϕas the azimuthal angles, respectively, the probability of $\mathbf{c}_r^*$ pointing into an element of solid angle $d\omega = sin\theta\, d\theta\, d\phi$ is uniform. Since $d\omega$ may also be expressed as $d\omega = -d(cos\theta)d\phi$, therefore, the probability of $\mathbf{c}_r^*$ pointing over $cos\theta$ and ϕ is also uniform, and distributed over -1 to 1, and 0 to 2π, respectively. From the inverse cumulative method for sampling from a uniform distribution, the values of $cos\theta$ and ϕ may be sampled as:

$$cos\theta = 2ranf - 1, \tag{4.63}$$

$$\phi = 2\pi ranf, \tag{4.64}$$

where $ranf$ is a uniformly sampled random number between zero and one. The three components of $\mathbf{c}_r^*$ in the Cartesian directions are easily obtained from spherical coordinates as:

$$x = c_r^* sin\theta cos\phi, \tag{4.65}$$

$$y = c_r^* sin\theta sin\phi, \tag{4.66}$$

$$z = c_r^* cos\theta. \tag{4.67}$$

The components of the post-collision velocities can thus be found with the help of Equations 4.57 and 4.58 as:

$$u_1^* = \frac{m_1 u_1 + m_2 u_2}{m_1 + m_2} + \frac{m_2}{m_1 + m_2} c_r^* sin\theta cos\phi, \tag{4.68}$$

$$u_2^* = \frac{m_1 u_1 + m_2 u_2}{m_1 + m_2} - \frac{m_1}{m_1 + m_2} c_r^* sin\theta cos\phi, \tag{4.69}$$

$$v_1^* = \frac{m_1 v_1 + m_2 v_2}{m_1 + m_2} + \frac{m_2}{m_1 + m_2} c_r^* sin\theta sin\phi, \tag{4.70}$$

$$v_2^* = \frac{m_1 v_1 + m_2 v_2}{m_1 + m_2} - \frac{m_1}{m_1 + m_2} c_r^* sin\theta sin\phi, \tag{4.71}$$

$$w_1^* = \frac{m_1 w_1 + m_2 w_2}{m_1 + m_2} + \frac{m_2}{m_1 + m_2} c_r^* cos\theta, \tag{4.72}$$

$$w_2^* = \frac{m_1 w_1 + m_2 w_2}{m_1 + m_2} - \frac{m_1}{m_1 + m_2} c_r^* cos\theta. \tag{4.73}$$

As stated before, the collision velocities are recalculated each time step in the event that the diameters of two particles overlap. It is possible, however, that the recalculated velocities are insufficient for the particles to completely separate. In order to avoid having the code go through these calculations again, thus throwing the two particles back in towards each other, an additional condition must be met to ensure that only particle which overlap and are approaching each other are flagged for collision. Two particles are considered to be approaching each other if the dot product of their respective velocities is negative. Otherwise the program knows that while two particles may be overlapping, they are moving away from each other and likely collided in the previous time step.

Chapter 5

Implementation and Code Development

In order to reduce the complexity of the software development aspect of this work it was decided that much of the baseline capabilities needed could be found in third party software. The focus of this work is not meant to be on the development of a rudimentary incompressible flow solver or on a Lagrangian particle tracking algorithm. Therefore, it was deemed appropriate to identify and utilize an existing code which included these attributes. OpenFOAM was selected due to the code being freely available under the GTK General Public License, having the required baseline tools as well as having a large and active online user community.

5.1 OpenFOAM overview

OpenFOAM stands for Open Field Operation and Manipulation. It is an open source CFD software package produced by a commercial company, OpenCFD Ltd. It has a large user base across most areas of engineering and science, from both commercial and academic organizations. OpenFOAM has an extensive range of features to solve anything from complex fluid flows involving chemical reactions, turbulence and heat transfer, to solid dynamics and electromagnetics. Work was begun on the *colloidalFoam* solver, developed in this work, using OpenFOAM version 1.4.1. The code was later upgraded to version 2.2.x.

The core technology of OpenFOAM is a flexible set of efficient C++ modules. These are used to build a wealth of: *solvers*, to simulate specific problems in engineering mechanics; *utilities*, to perform pre- and post-processing tasks ranging from simple data manipulations to visualization and mesh processing; *libraries*, to create toolboxes that are accessible to the solvers/utilities, such as libraries of physical models. Available solvers are included for incompressible, compressible, laminar, turbulent, multiphase, reacting, and non-reacting

flows.

OpenFOAM is supplied with numerous pre-configured solvers, utilities and libraries and so can be used like any typical simulation package. However, it is open, not only in terms of source code, but also in its structure and hierarchical design, so that its solvers, utilities and libraries are fully extensible.

OpenFOAM uses finite volume numerics to solve systems of partial differential equations ascribed on any 3D unstructured mesh of polyhedral cells. The fluid flow solvers are developed within a robust, implicit, pressure-velocity, iterative solution framework, although alternative techniques are applied to other continuum mechanics solvers. Domain decomposition parallelism is fundamental to the design of OpenFOAM and integrated at a low level so that solvers can generally be developed without the need for any 'parallel-specific' coding.

The underlying libraries and solvers provide an excellent foundation for development of new solvers. The tools and utilities, as well as the parallel nature of OpenFOAM, make it much easier to build and run cases using the newly developed solvers. OpenFOAM also has some inherent compatibilities with other software packages for mesh generation and visualization, such as open source packages Gmsh and ParaView, as well as many commercial applications.

Additional information on OpenFOAM can be found on the code's website, www.openfoam.org, including a user guide and a comprehensive C++ source guide.

Each case being solved using an OpenFOAM solver is run from a directory with the same general structure. Some of the files required by each may vary, but there are some commonly used by all. The basic directory tree of a case directory is shown in Figure 5.1.

The *system* directory contains the set of OpenFOAM "dictionaries" for setting parameters associated with the solution procedure itself, such as setting discretisation schemes, simulation time step, data output parameters, etc. Dictionaries for various OpenFOAM utilities are also placed here, such as the *decomposePar* utility for decomposing the domain to be run in parallel on multiple processors.

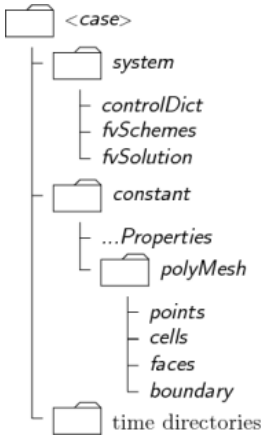

Fig. 5.1: OpenFOAM case directory tree structure

The *constant* directory contains a full description of the case mesh in a *polyMesh* sub-directory and files specifying the physical properties for the application concerned. For the *icoFoam* solver this consists of the *transportProperties* dictionary file which contains fluid properties. When the *solidParticleCloud* class is added, and for the *colloidalFoam* solver, two more dictionaries are added specifying a gravity vector (*environmentalProperties*) and properties relating to the particle cloud (*particleProperties*).

The "time" directories are named based on the simulated time at which data is written and contains individual files for the field data of the problem. In the case of *icoFoam* this includes the files p and U for the pressure field and velocity vector field, respectively. For simulations with particles each time directory has a *lagrangian* directory which contains the data files for the particle cloud including diameter, position and velocity. As most simulations start at a time of $t = 0$ the initial run usually consists of a single time directory *0* which contains initial values and boundary conditions. Additional time directories are written at specified intervals as the simulation runs.

5.2 OpenFOAM prototypes

As stated before, OpenFOAM includes a variety of solvers and libraries. These libraries are separated into various C++ classes which can be used independently or together in simple or complex solvers. While there are no solvers for modeling of colloidal particles included

with OpenFOAM, there are some solvers and classes which can be used as prototypes.

5.2.1 The *icoFoam* solver

One of the most basic CFD solver found in the OpenFOAM package is *icoFoam*. This is a simple transient solver for incompressible, laminar flow of Newtonian fluids. At this stage of developement of the *colloidalFoam* solver a complex fluid flow was not of interest, so *icoFoam* serves as an ideal prototype for the fluid solution component of the colloidal particle model.

The *icoFoam* solver solves for the velocity and pressure field of the supplied mesh. The pressure field is normalized by the fluid density, thus eliminating density as a term in the equation. Kinematic viscosity becomes the only fluid constant necessary to obtain a solution.

The structure of the icoFoam.C file is of particular interest and mimicked in colloidalFoam.C. The *main*() function begins by reading in the run parameters and the mesh and then including the createFields.H file. This file contains all the calls to OpenFOAM

After including the necessary OpenFOAM libraries a loop is opened to begin the time-step iterations. The equation to be solved is defined and solved within this loop using a combination of basic solvers provided within OpenFOAM that correspond to the Navier-Stokes equations for transient incompressible laminar flow. This definition and solution is shown in Algorithm 1. As can be seen this solution uses a time derivative, divergence, Laplacian and gradient to solve for an updated fluid velocity field. There are various numerical schemes available to use in solving each of these elements of the velocity equation. The specific numerical scheme can be chosen and set in a dictionary file at runtime.

After solving for the velocity field icoFoam.C goes on to correct the pressure field using the PISO algorithm. This algorithm also uses several basic OpenFOAM tools to solve the necessary equations using various numerical schemes. As this portion of the code is not altered in the implementation of *colloidalFoam*, the details of it are not presented here. The numerical schemes occupy a layer of OpenFOAM that is not touched in any code modifications and are also not presented here.

Algorithm 1 Definition and solution of the velocity equation (UEqn) in icoFoam.C

```
00056           fvVectorMatrix UEqn
00057           (
00058               fvm::ddt(U)
00059             + fvm::div(phi, U)
00060             - fvm::laplacian(nu, U)
00061           );
00062
00063           solve(UEqn == -fvc::grad(p));
```

Another file used by *icoFoam* is createFields.H. This file is primarily responsible for reading in the pressure and velocity fields as well as the lone fluid constant, *nu*. This becomes important when modifying the solver in any way that requires reading in additional fields or constants, as is the case for *colloidalFoam*.

5.2.2 The *solidParticleCloud* class

There are various solvers in OpenFOAM which utilize a Lagrangian particle tracking algorithm to model sprays or molecular dynamics. These all use a common set of Lagrangian classes, including the *cloud* and *particle* classes. One class that stems from these two base classes is the *solidParticleCloud* class, along with the *solidParticle* class. While not utilized by any of the included solvers, this class does provide a framework for a solver that propogates solid particles through a fluid medium.

The elements of the *solidParticle* and *solidParticleCloud* classes include drag, bouyancy and collision with walls. The particles are treated as rigid spheres with a unique diameter, position and velocity. All particles are given a common density value for bouyancy calculation purposes. These aspects of the class provided a good foundation to build on for the purpose of modeling colloidal particles.

Each particle is capable of querying the local fluid properties based on the cell it is in. The default configuration of the *solidParticle* class involves obtaining interpolated values for the fluid velocity, density and kinematic viscosity at the particle location.

5.2.3 Integration of *solidParticleCloud* and *icoFoam*

In its default form the *icoFoam* solver is not set up to directly integrate the *solidParticleCloud* class. With a few modifications, though, a new solver can be set up which combines the functionality of an incompressible, laminar flow solver and a lagrangian solid particle tracking algorithm. This was the first step before enhancing the *solidParticleCloud* class and adding in colloidal forces. As discussed earlier the *icoFoam* solver was selected due to its simplicity so that the focus could remain on the colloidal particle model. However, this does not preclude the use of other fluid solvers that may better suit a particular application.

There are a few differences between the default setup of *icoFoam* and *solidParticleCloud*. The former requires only the kinematic viscosity of the fluid to be read in as a constant. The latter is set up to read in variables specific to the particles and cloud on its own, but requires both the fluid viscosity and density as a field variable from the fluid mesh. A gravity vector is another requirement in order to compute bouyancy forces. In order to make *icoFoam* compatible with *solidParticleCloud* the createFields.H file is modified so that a fluid density value is read in from the same location as the fluid viscosity. These are both then used to initialize scalar fields for the properties. This is a bit redundant considering that the fields are of a constant value, but it also allows for the *solidParticle* library to be used as compiled with the OpenFOAM installation.

5.3 Development of the *colloidalFoam* solver

A solver based on *icoFoam* with solid particles has been presented which utilized the existing OpenFOAM class *solidParticleCloud* and minimal changes to the *icoFoam* source code. More extensive changes to the solver source code, as well as all changes made to the source code of the *solidParticleCloud* and *solidParticle* classes, will now be discussed.

Previous to this point the solver and classes being considered were *icoFoam*, *solidParticleCloud* and *solidParticle*, along with their associated files, icoFoam.C, solidParticleCloud.C, etc. From this point on the new solver *colloidalFoam* will be referred to instead of *icoFoam*. The *solidParticleCloud* and *solidParticle* classes become *colloidParticleCloud* and *colloidParticle* and all associated files will likewise be referred to based on this name change

Algorithm 2 Altered createFields.H excerpt to accomodate the *solidParticle* library

```
dimensionedScalar nu1
(
    transportProperties.lookup("nu")
);

dimensionedScalar rho1
(
    transportProperties.lookup("rho")
);


volScalarField rho
(
    IOobject
    (
        "rho",
        runTime.timeName(),
        mesh,
        IOobject::MUST_READ,
        IOobject::NO_WRITE
    ),
    mesh
);

volScalarField nu
(
    IOobject
    (
        "nu",
        runTime.timeName(),
        mesh,
        IOobject::MUST_READ,
        IOobject::NO_WRITE
    ),
    mesh
);
```

(icoFoam.C becomes colloidalFoam.C, etc.).

5.3.1 Initial enhancements

Before moving on to include colloidal surface forces in this new solver, a few additional enhacements were added.

nu and *rho* as constants

The *solidParticle* class required that the fluid viscosity and density be available as field variables. This is useful in the case that it is used in conjuction with a solver that considers variations in these properties but becomes an unnecessary burden when they can be treated as constants. The createFields.H file was modified in order to create *nu* and *rho* fields for the fluid based on constant values read in from a dictionary file. In order to simplify the code and reduce memory requirements these field creations may be removed, but changes must now be made to the solidParticle.C file, which will now be known as colloidParticle.C. The change is simple and involves replacing the field interpolation functions *td.rhoInterp().interpolate(cpw)* and *td.nuInterp().interpolate(cpw)* with the fluid property access functions *td.spc().rhoc()* and *td.spc().nuc()*. These property access functions are written into the colloidParticleCloudI.H file so that the cloud is able to pass these values to the individual particles.

Adding basic particle injection

The solver as-is is capable of reading in an initial cloud of particles and then propogating these particles. This is satisfactory for running the test cases meant to observe the behavior of one or two particles, but is not capable of running a case in which particles are injected throughout the simulation. Some of the OpenFOAM lagrangian libraries include injection algorithms, but the *solidParticleCloud* class did not have any type of injection associated with it. Rather than try to mimic some of these complex injection models, a simple injection subroutine was created for the *colloidParticleCloud* class.

The *particleProperties* dictionary includes an *injection* section for user supplied parameters to control particle creation. These terms include: a mean and variance for the number of particles to be injected per time step; a mean initial velocity vector and a scalar velocity variance; two sets of coordinates designating the two opposing corners of a rectangular block in which particles are created; a mean and variance for the particle diameter; a start and end time for the injection. Using OpenFOAM's random number generator the number, diameter, initial position and velocity of the particles to be injected in a given time step are calculated. This is done for every time step from the designated start time to the stop time.

While this injection method is crude it is sufficient to ensure continuous creation of particles in a channel flow test case. The main issue with this is that the rate at which particles are injected is dependent not only on the injection parameters read in, but also on the time step chosen for the whole simulation. With some foresight, however, the mean number of particles injected each time step can be adjusted to acheive the desired injection rate for a given time step size.

Fluid-particle momentum coupling

The solid particles now integrated into a fluid solver are able to obtain the information needed from their location within the fluid field to calculate the local drag force. While this allows the fluid to influence the particles, it does not take into consideration any influence that the particles can have on the fluid. For many cases the particles will be sparse and will have negligible effect on the fluid. However, some may wish to consider a case in which particles can begin to accumulate enough that they impede fluid flow. This consideration can be met by adding a source vector to the UEqn in colloidalFoam.C which is equal to the net change in momentum of all the particles within a fluid cell.

Addition of particle collisions

While the *solidParticle* class models particle collision with the wall, there is no subroutine for handling particles colliding with each other. This consideration is similar to that of adding particle effects on the fluid in that both are relatively insignificant when the particle

field is sparse. It is also similar in that it becomes necessary for cases in which particles begin to become more dense in some fluid cells.

Some of the OpenFOAM lagrangian libraries and classes include various collision algorithms. These are mostly for spray modeling in which the particles that collide can combine. While this does not match the needs of a solid colloid particle model, the code provides a good example of integrating a collision model into a cloud-based class.

Once it has been determined that the two particles being considered are indeed colliding, the program enters the specific collision algorithm designated in the *particleProperties* runtime dictionary. The only options made available in *colloidalFoam* are hard sphere collision or no collision. The hard sphere collision model has been described earlier. The coding of this algorithm is straightforward and will not be described here.

5.3.2 The colloidalParticleCloud class

Now that many of the behaviors expected of spherical particles in a fluid have been accounted for, we move on to the consideration of colloidal forces. Multiple models for Van der Waals and Electrical Double Layer forces have been discussed, each requiring several environmental variables and the calculation of distances between surfaces.

Exclusion of inter-particle forces

In discussing the various models for the Van der Waals and Electrical Double Layer forces, equations were derived for interaction between a sphere and a flat surface as well as between two spheres. The attraction or repulsion between colloidal particles can be a significant consideration for several modeling applications when the rate of particle coalescence is important. While inclusion of this feature would remain a key goal of future development, the implementation of this involves some significant hurdles. Each particle in each time step would need to reference every other particle being simulated and calculate the distance to them to determine whether they are within range of the forces. Then the forces between the particle being considered and all the particles within range must be calculated and resolved into a net force vector. As the number of particles being simulated goes up,

the number of computations required each time step rises exponentially, significantly slowing the simulation. As such the feature was deemed out of scope for the current version of *colloidalFoam*.

Calculating separation from wall

All colloidal force models being considered require the distance h between the particle surface and the wall. In order to do this the position vector of the particle in question is compared to points on a given wall surface patch. The distance between the particle center and the nearest point on the wall patch is then calculated. This is done for all wall patches in the mesh with the distance for the current and the previous patch being compared each time and the smallest being kept. The distance h is needed later as a scalar value, but a unit vector Sf is also calculated to determine the direction of the force vector later. Once all wall patches have been cycled through and the smallest value of h determined, the value is corrected by subtracting half the diameter of the particle. This process is shown in Algorithm 3.

In querying all wall patches for every particle this approach does tend to eat up CPU cycles. For a given number of wall patches in the fluid mesh the compute cycles will rise linearly with an increase in the number of particles being simulated. This is still more manageable than the exponential increase in computation needed if a similarly direct approach were used to calculate distances between particles. A method for identifying the nearest wall patch and nearest neighboring particles could drastically cut down on the runtime and make inter-particle force simulation feasible. Unfortunately, none were identified and work progressed in order to demonstrate the capabilities of the simulation, rather than focus on the computational performance of the algorithm.

Inclusion of multiple colloidal force models

Various models representing the Van der Waals and Electric Double-Layer forces have been discussed, each having their own strengths and weaknesses. The option to choose which models are used adds flexibility to the solver as well as potential to easily add additional

Algorithm 3 Wall separation distance calculation

```
forAll(patches,patchI)
{
    const polyPatch& pPatch = patches[patchI];
    const pointField& points = pPatch.points();

    if(pPatch.type() == "wall")
    {
        forAll(pPatch,faceI)
        {
            pointHit whoo = pPatch[faceI].nearestPoint(position(),points);
            hDistance[newh] = whoo.distance();

            if(hDistance[newh] < hDistance[oldh])
            {
                h = hDistance[newh];
                hDistance[oldh] = hDistance[newh];
                Sf = position() - whoo.rawPoint();
                Sf /= mag(Sf);
            }
        }
    }
}
h -= d_/2.;
```

models and functions. The OpenFOAM dictionary file *particleProperties* is modified to include the colloidal properties of the case being run as well as a specification for which model is used. Options included in this code are *Gregory*, *SchenkelKitchener* and *Czarnecky* for the Van der Waals force and *constantCharge*, *constantPotential* and *LSA* for the Electric Double-Layer force.

Once the separation distance between a particle and the nearest wall is calculated in *colloidalParticle.C* it is determined whether the particle is within the range of influence. For the case that $h < 10d$, where d is the particle diameter, the calculation of colloidal forces proceeds. The stipulation of $h > 0$ is also added to ensure that should a particle somehow end up crossing the wall boundary that forces are not calculated for a negative separation distance, producing erroneous values. Otherwise the total colloidal force on the particle is assumed to be zero and the drag and bouyancy forces on the particle are calculated on their own. The limit of ten times the diameter of the particle is somewhat arbitrary, though based on plotted force functions for particles with the properties being considered this limit shows to be a good cutoff value with colloidal forces becoming negligible compared to other forces acting on the particle. This can be changed easily if needed for cases in which this value is not appropriate.

Using the scalar value h calculated previously forces are calculated using the particle-wall portion of Equations 4.22, 4.25 or 4.27 (for the VDW force) and 4.34, 4.36 or 4.38 (for the EDL force). The scalar values of the VDW and EDL forces are added together and multiplied by the unit vector Sf to obtain the total colloidal force vector and then divided by the particle mass m to obtain the colloidal contribution to the acceleration vector of the particle as shown in Algorithm 4. Because the equations used to calculate the colloidal force approach infinity as h approaches zero, a maximum force limit is hard-coded in to avoid a divide by zero error in the code and to keep the particle from having too great an acceleration. This implimentation is imperfect and the value of this limit must be chosen depending on the colloidal properties, the surface elasticity as well as the time step being used to avoid particle colliding with the wall at too great a velocity and bouncing back too

Algorithm 4 Total colloidal force vector calculation and combination with other forces to be integrated and obtain the new velocity vector of the particle.

```
vector Fcolloid = vdwWall*Sf + edlWall*Sf;
if(mag(Fcolloid) > Flimit)
{
    Fcolloid /= mag(Fcolloid);
    Fcolloid *= Flimit;
}

Ucolloid = Fcolloid/m_;
U_ = (U_ + dt*(Ucolloid + Dc*Uc + (1.0 - rhoc/rhop)*td.g()))/(1.0 + dt*Dc);

td.spc().smoment()[celli] += m_*(U_ - Uc)*Dc*dt;
```

far.

Algorithm 4 also shows the calculation of the new particle velocity using the total colloidal, drag and bouyancy forces. This is calculated using the integration method described in the previous chapter which resulted in Equation 4.7. The change in momentum of this particle due to drag is also calculated and stored to be used later in calculating the source term for the fluid as described in Equation 4.15.

Chapter 6

Simulation Setup

6.1 Simulation setup and parameters

6.1.1 Particle and fluid properties

All simulations represent a suspension of polystyrene-latex particles in a NaCl electrolyte solution. The particle sizes, fluid velocity and electrolyte concentration vary for several verification studies, but fluid temperature, density and viscosity, as well as the Hamaker constant used to compute Van der Waals force values all remain constant. This is meant to mirror the properties of the system used by Unni and Yang [16] and values for these properties were obtained from their report. They are shown in Table 6.1. The zeta values used for computing the Electric Double-Layer force vary with the electrolyte concentration. Values for these at the concentrations considered are shown in Table 6.2.

6.1.2 Fluid simulation grids

The particle transport algorithm used is connected to a CFD simulation and must have a grid associated with the physical domain of the fluid. Several of the verification simulations in this chapter do not necessitate a large complex grid. For the simulations demonstrating collisions, drag and a single particle approaching a wall, the domain used was simply a cube. The size of the cube was set to 1 meter dimensions on all sides as this would be large enough

Table 6.1: Fluid and colloidal properties

Property	Value
Kinematic viscosity (ν)	0.9×10^{-6} m^2/s
Solution density (ρ)	1050 kg/m^3
Latex-NaCl-glass Hamaker constant (A_k)	0.91×10^{-20} J

Table 6.2: Particle-wall and inter particle zeta potentials corresponding to various electrolyte (NaCl) concentrations.

NaCl concentration	Zeta potentials	
	ζ_w	ζ_p
0.1 M	-18 mV	-14 mV
0.01 M	-23 mV	-20 mV
0.001 M	-28 mV	-24 mV

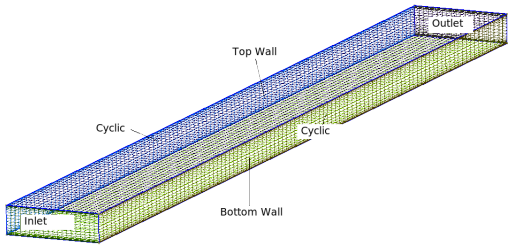

Fig. 6.1: Micro channel surface grid and boundary conditions

to avoid any unwanted boundary interference. All six boundaries were treated as walls with a constant grid size of 0.1 m. For those simulations of interaction with a wall (wall collision and wall colloidal capture) the particle will be placed near the center of one of the wall boundaries. For the drag and particle-particle collision simulations the particles are placed near the center of the cube.

A final set of simulations run uses a geometry meant to mimic the geometry of the microchannel experiment used by Unni and Yang [16]. Their experiment involved two parallel glass plates 0.3 mm apart. The simulation grid is shown in Figure 6.1 and consists of a constant velocity inlet, a pressure outlet, wall surfaces on the top and bottom and cyclic sides. The simulated length is 10 mm and the width is 1 mm. The gravity vector points downward from the top wall. Particles are injected near the inlet.

6.2 Collision models

6.2.1 Wall collision

The wall collision model offered in the solidParticle class is used in the solver developed here. This model uses two user inputs, ϵ and μ, as terms for absorption of energy at impact in the wall normal and tangential directions, respectively. The cases discussed here demonstrate the use of these terms to obtain a fully elastic ($\epsilon = 1$, $\mu = 0$) collision, fully inelastic ($\epsilon = 0$, $\mu = 1$) collision and variations of the two. In order to better observe the effect of the collision alone the gravity vector is set to zero and the fluid density is greatly reduced to minimize drag effects.

The physical significance of the two terms given can be interpreted as normal elasticity (ϵ) and a kind of surface roughness term (μ). These values may be calibrated by the user to best represent the interaction between particles and surrounding surfaces for a specific case.

6.2.2 Particle collision

The particle collision algorithm is based on the assumptions of conservation of energy and momentum. These can be verified by noting the velocities of two particles before and after collision and then calculating and comparing the total kinetic energy and momentum of the system (of two particles) before and after the collision. This is tested using two particles of equal size and density fired towards each other at equal but opposite velocities. In one case the particles approach each others centers (head-on collision) and in another case they approach each others tangents (offset collision).

6.3 Drag model

The drag model, including coupling with the fluid, is tested by comparing the speed of a simulated free falling particle with the expected terminal velocity analytic solution. Terminal velocity is defined as the point at which an object falling under the force of gravity no longer accelerates due to the drag force equalling the weight. For a spherical particle

with diameter d and density ρ_p the weight is defined as

$$W_{particle} = (\rho_p - \rho) V_p g = (\rho_p - \rho) \frac{\pi}{6} d^3 g \tag{6.1}$$

where V_p is the particle volume and g is the acceleration due to gravity. Assuming the condition of Stokes flow, i.e. $Re < 0.01$, which is reasonable for the small particle diameters and densities being dealt with, the drag force is

$$F_D = \frac{\pi}{8} d^2 \rho C_D u^2 \tag{6.2}$$

$$C_D = \frac{24}{Re} = 24 \frac{\nu}{ud} \tag{6.3}$$

where ρ and ν are the fluid density and kinematic viscosity, respectively, and u represents the velocity of the particle in the fluid. Combining Equations 6.2 and 6.3 and setting $F_D = W_{particle}$ will show us the terminal velocity of a free falling sphere:

$$F_D = \frac{\pi}{8} d^2 \rho \cdot 24 \frac{\nu}{u_{term} d} u_{term}^2 = 3\pi d \rho \nu u_{term} = W_{particle} = (\rho_p - \rho) \frac{\pi}{6} d^3 g$$

Rearranging to solve for u_{term} gives us

$$u_{term} = \frac{d^2 g}{18\nu} \left(\frac{\rho_p}{\rho} - 1 \right). \tag{6.4}$$

The free falling particle is simulated within a cubic domain. The six sides of the cube are all walls, creating an enclosed space so the fluid does not gain any net momentum, but it is also large enough that the particle can achieve terminal velocity well before approaching any walls. The particle begins at rest in the center of the domain and is allowed to fall under the force of gravity. The simulation is allowed to run until a near constant velocity is acheived.

As the purpose of these models is to simulate micro-scale particles for which colloidal forces will be significant, the particles and fluid simulated have properties similar to what is

Table 6.3: Values for testing drag model

Variable	Value(s)
Particle diameter, d	0.1, 1 and 10 μm
Particle density, ρ_p	2650 kg/m^3
Fluid density, ρ	1050 kg/m^3
Fluid kinematic viscosity, ν	$0.9 \times 10^{-6} m^2/s$

being simulated for the colloidal tests. These properties are listed in Table 6.3.

With the fluid velocity field being affected by particles passing through each cell there may arise some grid dependency as the particle drag is calculated based in part on the local fluid velocity. To ensure that the terminal velocity solution is independent of the grid a Richardson extrapolation study is done. The cubic domain is 1mm x 1mm x 1mm and evenly divided in each dimension by 10, 20 and 40 to provide cell sizes of $100\mu m$, $50\mu m$ and $25\mu m$, respectively. All simulations are run until a terminal velocity u_{term} is found and the apparent order p, approximate relative error e_a, extrapolated relative error e_{ext}, and fine grid convergence index GCI_{fine} is calcualted and reported.

6.4 Colloidal models

The colloidal models used to evaluate the interaction force between a particle and a wall surface are tested within the framework of the CFD model and compared to expected plotted values. The simulation involves a single particle being placed a half diameter distance above a wall surface. The gravity vector is directed perpendicular into the wall surface such that the particle begins to descend when the simulation commences. As the particle approaches the colloidal force algorithm is triggered and the colloidal force becomes more significant the closer the particle gets to the wall. The simulation is run for a length of time significant enough to allow the particle to either collide with the wall or approach the energy barrier, depending on the environment variables used in the simulation. The variable being used for comparison is the final at rest separation distance between the surfaces of the sphere and wall.

The Van der Waals force models used are those provided by Gregory, Schenkel and

Kitchener, and Czarnecky [22]. While the Czarnecky model is recommended as more accurate at distances of $h > \lambda/4\pi$, with the Gregory and Schenkel-Kitchener models being more accurate in closer ranges, these models are tested separately in order to compare how well the simulated particle matches expected results for each given model. All simulations are run with the same set of environmental variables used in the calculation of the Van der Waals force.

The electrostatic double layer (EDL) force models used are the constant charge and constant potential derivations of the linear Poisson-Boltzmann offered by Gregory [25], as well as the linear superposition approximation (LSA). Simulations and direct computations are performed for cases with three different NaCl concentrations of the suspending fluid. All possible combinations of the three Van der Waals models and three EDL models are used to calculate the total colloidal force.

6.5 Large scale simulations

With the functionality of each element of the *colloidalFoam* models independently verified as described, it makes sense to test the solver on a more practical scale to ensure that the code can run stable with collisions, drag, bouyancy, and colloidal forces all active and thousands of particles being simulated. The micro-channel geometry used by Unni and Yang has been selected as a test case for this purpose. As described in an earlier chapter a simple particle injection algorithm has been added to the solver for the purpose of this test.

The Gregory Van der Waals model and the LSA EDL model is used as a baseline in this set of simulations. The simulation time step dt and the average number of particles injected per time step $ppts$ are both varied in accordance with Table 6.4. The Van der Waals and EDL models are also varied using a base time step and injection rate. The simulation is run for a total simulated time of 2.5 seconds to ensure that particles have the chance to propogate from the inlet through to the outlet and fill out the physical domain. The total number of particles in each simulation is reported as well as any issues or instabilities observed.

In addition to these large scale simulations run to check general code stability, an

Table 6.4: Parameters varied for micro-channel simulations

Case #	VDW model	EDL model	dt	$ppts$	Mean injection rate
1	Gregory	LSA	0.0001	2	$20,000 \cdot s^{-1}$
2	Gregory	LSA	0.0002	2	$10,000 \cdot s^{-1}$
3	Gregory	LSA	0.0001	4	$40,000 \cdot s^{-1}$
4	Gregory	LSA	0.0002	4	$20,000 \cdot s^{-1}$
5	Gregory	Constant Potential	0.0001	2	$20,000 \cdot s^{-1}$
6	Gregory	Constant Charge	0.0001	2	$20,000 \cdot s^{-1}$
7	Schenkel & Kitchener	LSA	0.0001	2	$20,000 \cdot s^{-1}$
8	Czarnecky	LSA	0.0001	2	$20,000 \cdot s^{-1}$

additional simulation of the microchannel was run to observe the rate of particle adsorption onto the microchannel surfaces. The time step used for this simulation is 0.0001 seconds, with a mean injection rate of 60,000 particles per second. Particles are given a set diameter of 0.5 μm. The Gregory VDW model and the LSA EDL model are used. All other parameters remain the same. These changes are meant to more closely approximate the set up for the experiment and simulations performed by Unni and Yang [16]. The surface coverage due to particle adsorption is calculated and compared with simulated results from Unni and Yang. This is done by summing the cross-sectional areas of all adsorbed particles and then dividing by the total surface area of the wall. A particle is considered adsorbed once the value of h/a_p reaches the primary energy minimum separation, H_0. This value is set to $H_0 = 0.002$ in accordance with what Unni and Yang used in their study.

Chapter 7

Results and Discussions

7.1 Collision models

7.1.1 Wall collision

Figure 7.1 shows the path of a particle which initially approaches a wall at a 45° angle. In the fully elastic case it can be seen that the particle is perfectly reflected off the wall at a 45° angle. The fully inelastic case shows the particle captured on the wall surface at the point of contact. For the normally inelastic condition the particle is again captured on the wall, but it also continues to slide along the surface of the wall, uninhibited in the tangential direction. The tangentially inelastic case shows the particle reflecting off the wall, but with all tangential energy absorbed at impact.

7.1.2 Particle collision

The resulting paths of the particles for each case are shown in Figure 7.2. The random term used to close the conservation equations can be seen to affect these collisions. In the case of the head-on collision it would be expected that the particles would rebound directly back from each other, but instead they are thrown to the sides. The offset collision would be expected to have the particles reflect off of the 45° plane of collision, but the random element precludes this.

In both of these cases the velocities of the particles are taken immediately before and after the collision takes place. These velocities are shown in Table 7.1 and Table 7.2 along with the calculated momentum and kinetic energy of the whole system for both before and after the collision. It can be seen that the net momentum of the two-particle system is zero both before and after the collision

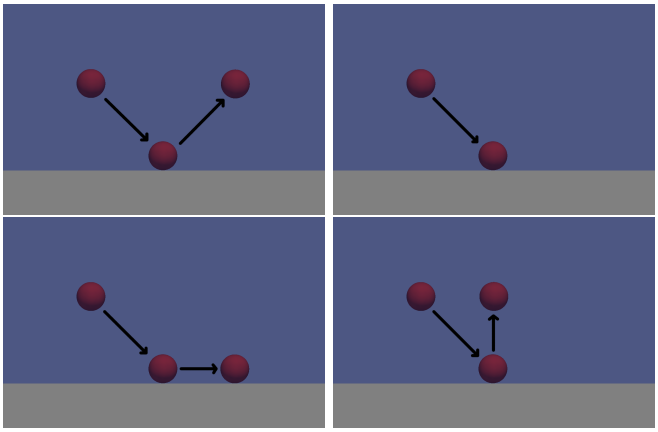

Fig. 7.1: Representations of fully elastic (top left), fully inelastic (top right), normal inelastic (bottom left) and tangentially inelastic (bottom right).

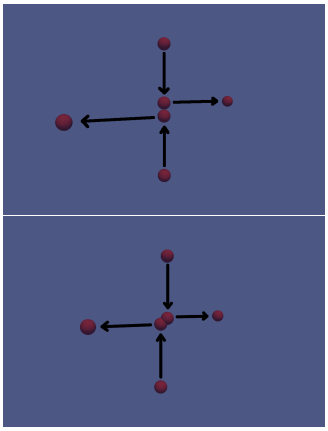

Fig. 7.2: An instance of head-on collision (top) and offset collision (bottom). The random element of the particle collision model can clearly be seen.

Table 7.1: Comparison of pre- and post-collision momentum and kinetic energy for the head-on collision case.

	Head-on	
	Particle 1	Particle 2
Initial velocity	$-0.05\hat{k}$	$0.05\hat{k}$
Final velocity	$-0.0375\hat{i} + 0.0331\hat{j} + 0.0015\hat{k}\backslash$	$0.0375\hat{i} - 0.0331\hat{j} - 0.0015\hat{k}$
Particle mass	4.189e-12	4.189e-12
Initial momentum	0.0	
Final momentum	0.0	
Initial K.E.	1.047e-11	
Final K.E.	1.047e-11	

Table 7.2: Comparison of pre- and post-collision momentum and kinetic energy for the offset collision case.

	Offset	
	Particle 1	Particle 2
Initial velocity	$-0.05\hat{k}$	$0.05\hat{k}$
Final velocity		
Particle mass	4.189e-12	4.189e-12
Initial momentum	0.0	
Final momentum	0.0	
Initial K.E.	1.047e-11	
Final K.E.	1.047e-11	

Table 7.3: Terminal velocity results using different grid sizes

Mesh Size, h	Terminal Velocity, u_{term}
$100\mu m$	0.922966 $\mu m/s$
$50\mu m$	0.925141 $\mu m/s$
$25\mu m$	0.932389 $\mu m/s$

7.2 Drag model

For a 1 μm particle, given the properties listed in Table 6.3, the terminal velocity would be expected to be 0.922751 $\mu m/s$, as calculated by Eq. 6.4. Simulating this same particle using a cell width of 100 μm shows the particle reaching a terminal velocity of 0.922966 $\mu m/s$. The only major difference between what occurs in the simulation and what is calculated analytically here is that in the simulation the fluid velocity is affected by the particle. However, with a difference of less than 0.03% between these two values the effects of fluid coupling do not significantly alter the drag effects on the particle.

The values obtained from the Richardson extrapolation study are reported in Table 7.3. Using these results and the methods detailed in the JFE Statement on the Control of Numerical Accuracy an apparent order of $p = 1.737$ is obtained. An approximate relative error of $e_a = 0.0078$ is found between the medium and fine grids, and an extrapolated relative error of $e_{ext} = 0.0033$ is also found. The fine grid convergence index is $GCI_{fine} = 0.0042$. These figures indicate little variance in the solution as the mesh size is varied.

7.3 Colloidal models

Simulations were run using all nine combinations of Van der Waals and EDL force models, and surface properties for three different electrolyte concentrations of 0.1 M, 0.01 M and 0.001 M. The results of the simulated particle were compared with a force plot to determine the distance off the wall of the energy barrier.

For the 0.1 M case the plot indicates that the combined colloidal force for all models is attractive and contains no energy barrier point. The simulations confirm this as in all cases the particle collides with the wall repetitively. A small time step was necessary in order to capture the movement of the particle as it came closer to the wall and accelerated under greater force. The force models used approach an infinite attractive force very near the wall without taking into account the repulsive contact force between the wall and particle. In order to avoid an infinity error in the simulation the total colloidal force was limited. The level of capturability of the particle is controlled by adjusting the elasticity factors for the wall. The values for the normal and tangential elasticities were both set to 0.5. Under these circumstances the particle approached the wall and made contact and was continually reflected a short distance before being pulled back.

The combined colloidal force plots for the 0.01 M and 0.001 M cases all exhibit an energy barrier. The distance from the wall at which the total force is zero (including gravity and bouyancy effects) was calculated using each force model combination and compared to the resting position of the simulated particle. Comparison of the predicted and simulated particle separation for fluid electrolyte concentrations of 0.01 M and 0.001 M are shown in Tables 7.4 and 7.5, respectively.

7.4 Large scale simulations

The various micro-channel cases presented in the previous chapter were run successfully. Particle injection began at the beginning of the simulation, and the particle stream traversed the domain before simulation termination in each case. A sequence of images showing the progression of the particle stream through the domain for Case 1 in Table 6.4 is shown in Figure 7.3.

Table 7.4: Equilibrium distance of particle from wall for 0.01 M concentration electrolyte solution.

Van der Waals - EDL	Direct Computation	Simulation Result	Relative Error
Gregory - Const. charge	$1.99880E-08$	$2.00188E-08$	$1.540925E-03$
Gregory - Const. potential	$1.99741E-08$	$2.00050E-08$	$1.547003E-03$
Gregory - LSA	$1.98379E-08$	$1.98689E-08$	$1.562665E-03$
Sch.-Kitch. - Const. charge	$2.02801E-08$	$2.03121E-08$	$1.577901E-03$
Sch.-Kitch. - Const. potential	$2.02673E-08$	$2.02993E-08$	$1.578898E-03$
Sch.-Kitch. - LSA	$2.01278E-08$	$2.01599E-08$	$1.594809E-03$
Czarnecky - Const. charge	$1.95627E-08$	$1.95940E-08$	$1.599984E-03$
Czarnecky - Const. potential	$1.95460E-08$	$1.95774E-08$	$1.606467E-03$
Czarnecky - LSA	$1.94049E-08$	$1.94364E-08$	$1.623301E-03$

Table 7.5: Equilibrium distance of particle from wall for 0.001 M concentration electrolyte solution.

Van der Waals - EDL	Direct Computation	Simulation Result	Relative Error
Gregory - Const. charge	$9.57200E-08$	$9.58524E-08$	$1.383201E-03$
Gregory - Const. potential	$9.57187E-08$	$9.58511E-08$	$1.383220E-03$
Gregory - LSA	$9.51629E-08$	$9.52958E-08$	$1.396553E-03$
Sch.-Kitch. - Const. charge	$9.82027E-08$	$9.83378E-08$	$1.375726E-03$
Sch.-Kitch. - Const. potential	$9.82017E-08$	$9.83369E-08$	$1.376758E-03$
Sch.-Kitch. - LSA	$9.76501E-08$	$9.77857E-08$	$1.388631E-03$
Czarnecky - Const. charge	$9.80099E-08$	$9.81471E-08$	$1.399859E-03$
Czarnecky - Const. potential	$9.80099E-08$	$9.81461E-08$	$1.399873E-03$
Czarnecky - LSA	$9.74475E-08$	$9.75852E-08$	$1.413069E-03$

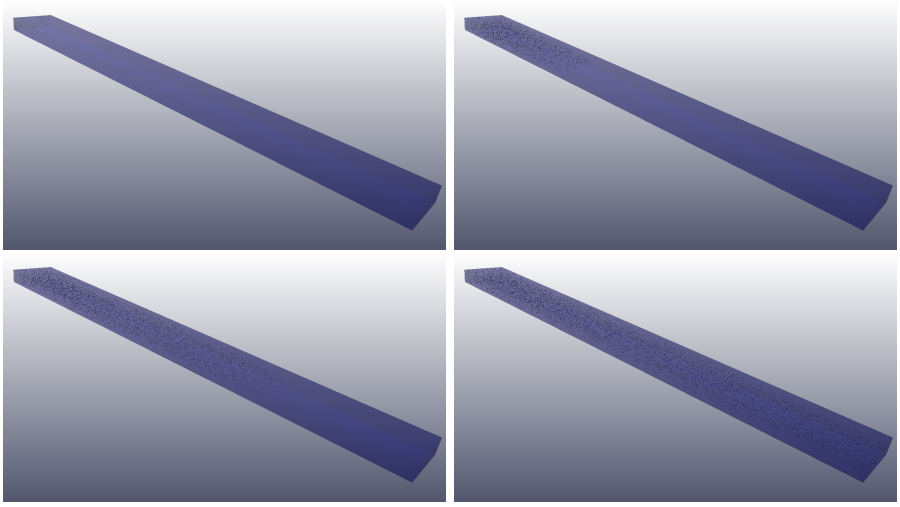

Fig. 7.3: Initiation of particle injection into micro channel

Table 7.6: Resultant run time and particle count of micro-channel simulations

Case #	t at simulation end	# of particles
1	1.5071	23,264
2	2.5	12,246
3	1.0456	38,141
4	2.5	22,096
5	1.5078	23,258
6	1.5097	23,266
7	1.5043	23,274
8	1.5073	23,176

Because the micro-channel simulation cases were run as a proof of concept, numerical results of physical significance were not obtained. Instead, in Table 7.6 are listed the simulation time and number of particles within the domain at the time of simulation termination. These correspond with the cases listed in Table 6.4. Due to a bug in the OpenFOAM parallel processing subroutine for particles, all cases were run in serial on a single processor. This resulted in lengthy run-times. A limit of 7 days was placed on code run time, and as a result not all simulations reached the 2.5 second mark.

In order to obtain results of a more useful nature, the total simulated time must increase a great deal. The results provided by Unni and Yang in their simulations and experiment ran for 50 minutes. In order to run a simulation of that scale it was necessary to update the code to the latest version of OpenFOAM in order to enable parallel computation. Once this was accomplished the final simulation described in the previous chapter was run on between 64 and 128 processors. The simulation was then able to produce results much quicker, on the order of minutes of simulated time instead of just seconds. In addition, the wall collision parameters and the colloidal force limit were attenuated in order to ensure that particles would be adsorbed onto the wall, and not merely bounce off. With these modifications the code and simulation were able to produce reasonable results when compared to the results from Unni and Yang. As can be seen from Figure 7.4 the total surface coverage over time is largely linear in both the results of this study and those provided by Unni and Yang. This study appears to have a lower rate of adsorption and a slightly longer start up period than the results provided by Unni and Yang.

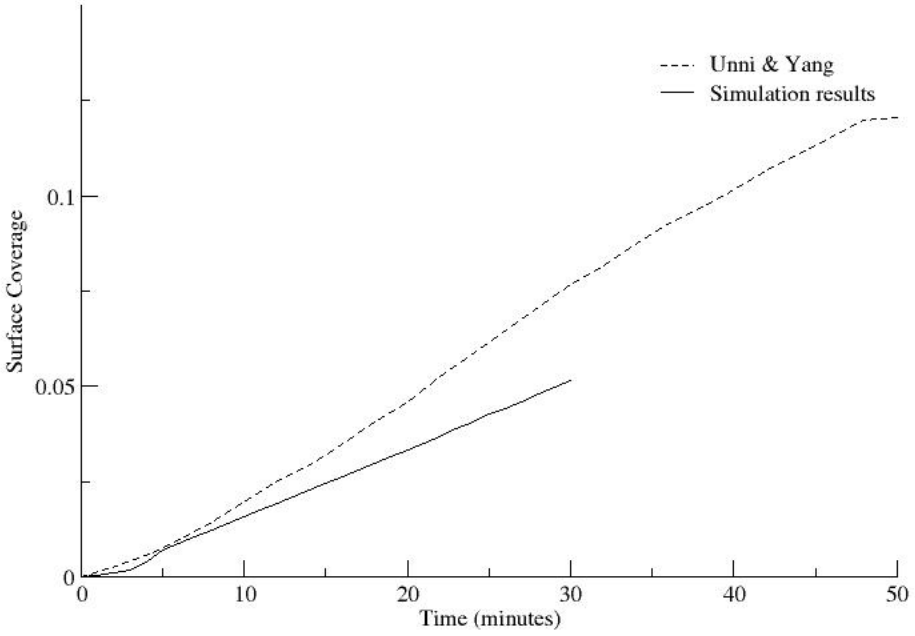

Fig. 7.4: Comparison of surface coverage over time with results provided by Unni and Yang

The primary difference between the simulations performed here and those performed by Unni and Yang are that they included Brownian Motion, while this study does not. The effect of including Brownian Motion appears to have allowed more particles to be captured by the wall as the random motion of particles at the edge of influence of the colloidal forces can be nudged closer. This helps to explain the higher rate of adsorption produced by Unni and Yang. In addition to differences in force and motion models, the simulation set up in both cases is fundamentally different. As was described in the previous chapter, the simulation used in this study involves a microchannel 1mm wide and 10mm long, with particles injected at the fluid inlet area. Unni and Yang use a cubic simulation cell with no inlet effects and with a constant number of particles which are recycled back into the domain as they leave. The inlet region in this study may have a small affect on particle adsorption as the fluid and particles need a short distance in the microchannel to normalize.

Chapter 8

Conclusions

8.1 Objectives met

All objectives listed in Chapter 3 were met as follows:

- A colloidal particle solver was produced using OpenFOAM as basis and which includes:
  - Includes models of the Van der Waals force on a particle near a wall as offered by Czarnecky, Schenkel and Kitchener, and Gregory
  - Includes models of the Electric Double Layer force on a particle near a wall as offered by Gregory
  - Accounts for wall collisions and inter-particle collisions
  - Provides a drag model that is coupled with the fluid solver
  - Provides a rudimentary injection model
- Each function of the model was tested individually. Wall collisions and inter-particle collisions were observed using one and two particles, respectively. All observed behaviors were as expected. The drag model and coupling with the fluid was tested using a free falling particle case which provided results inline with calculated terminal velocities. The behavior of a particle near a wall under the influence of the Van der Waals and Electric Double Layer forces was observed. The separation from the wall due to the primary energy barrier was observed and found to be in line with calculated values. Finally, the injection subroutine was tested using a microchannel case which included particles being injected at the inlet. The particles were added to the domain as expected, and the code was stable while running with a large number of particles.

- The microchannel case was fine-tuned to mimic the physical experiment provided by Unni and Yang, including geometry, chemistry and physical properties of the fluid, particles, and surfaces, and the size and number of particles injected. The surface coverage as particles adsorbed onto the microchannel walls was observed over time in the simulation, and data was compared with data produced by Unni and Yang. While this did not provide an exact match, it did provide results on the same order as those given by Unni and Yang. Differences in the setup and models used account for the difference in values obtained, but in both cases the data follows a largely linear pattern over time.

8.2 Potential future uses and applications

This solver provides a base that is meant to be adaptable and expanded for many different uses. In its current state the code can be used to simulate colloidal particle deposition on surfaces. Usage does not need to be limited to this, however. The usage of complex geometries would allow for the simulation of bacteria transport in ground water. The addition of inter-particle forces, as discussed above, can allow for the simulation of colloidal particle cloud coalescence. Enhanced treatment for a high particle density would allow for the simulation of sediment bed transport. The usage of a non-Newtonian fluid solver, in addition to 6 degree of freedom (6DOF) simulation of particles movement and a drag model for non-uniform shapes, would allow for the simulation of blood platelet transport within veins, arteries, or heart chambers.

With this code being provided online it lays the groundwork for scientists to use it as a basis for several different specific applications, including those listed above and more. Time to complete research can be cut by using this model and building on it. The colloidalFoam solver provides a functional base, and with upgrades to the code base and enhancements of its features can become a powerful and versatile tool for researching colloidal system behavior.

References

[1] Hauser, T. and Allen, J., "Numerical Simulation of the Behavior and Mobilization of Fine-grained Quartz Particles in Porous Media," *OpenFOAM International Conference*, 2007.

[2] McDowell-Boyer, L., Hunt, J., and Sitar, N., "Particle Transport Through Porous Media," *Water Resources Research*, Vol. 22, 1986, pp. 1901–1921.

[3] Jenny, H. and Smith, G., "Colloid Transport Aspects of Clay Pan Formation in Soil Profiles," *Soil Science*, Vol. 39, 1935, pp. 377–379.

[4] Ho, N. and Higuchi, W., "Preferential Aggregation and Coalescence in Heterodispersed Systems," *Journal of Pharmaceutical Sciences*, Vol. 57, 1968, pp. 436–442.

[5] Adamczyk, Z., Siwek, B., and Szyk, L., "Flow-Induced Surface Blocking Effects in Adsorption of Colloid Particles," *Journal of Colloid and Interface Science*, Vol. 174, 1995, pp. 130–141.

[6] Bolster, C. H., Walker, S. L., and Cook, K. L., "Comparison of Escherichia coli and Campylobacter jejuni Transport in Porous Media," *Journal of Environmental Quality*, Vol. 35, 2006, pp. 1018–1025.

[7] Hornberger, G., Mills, A., and Herman, J., "Bacterial Transport in Porous Media: Evaluation of a Model Using Laboratory Observations," *Water Resources Research*, Vol. 28, 1992, pp. 915–938.

[8] Bradford, S. A. and Bettahar, M., "Straining, Attachment, and Detachment of Cryptosporidium Oocysts in Saturated Porous Media," *Journal of Environmental Quality*, Vol. 34, 2005, pp. 469–478.

[9] Schmid-Schoenbein, H., Wells, R., and Schildkraut, R., "Microscopy and Viscometry of Blood Flow Under Uniform Shear Rate (Rheoscopy)," *Journal of Applied Physiology*, Vol. 26, May 1, 1969, pp. 674–78.

[10] Tan, Y., Gannon, J., Baveye, P., and Alexander, M., "Transport of Bacteria in an Aquifer Sand: Experiments and Model Simulations," *Water Resources Research*, Vol. 30, 1994, pp. 3243–3252.

[11] Barton, J. and Ford, R., "Mathematical Model for Characterization of Bacterial Migration Through Sand Cores," *Biotechnology and Bioengineering*, Vol. 53, 1997, pp. 487–496.

[12] Duffy, K., Ford, R., and Cummings, P., "Residence Time Calculation for Chemotactic Bacteria Within Porous Media," *Biophysical Journal*, Vol. 73, 1997, pp. 2930–2936.

[13] Israelachvili, J., *Intermolecular and Surface Forces*, Academic Press, Burlington, MA, 1992.

[14] Israelachvili, J. and McGuiggan, P., "Forces Between Surfaces in Liquids," *Science*, Vol. 241, No. 4867, 1988, pp. 795–800.

[15] Marshall, J., "Particulate Aggregate Formation and Wall Adhesion in Microchannel Flows," *Proceedings of ASME Fluids Engineering Division Summer Meeting 2006*, Vol. 1, 2006, pp. 519–526.

[16] Unni, H. and Yang, C., "Brownian Dynamics Simulation and Experimental Study of Colloidal Particle Deposition in a Microchannel Flow," *Journal of Colloid and Interface Science*, Vol. 291, 2005, pp. 28–36.

[17] Longest, P., Kleinstreuer, C., and Buchanan, J., "Efficient Computation of Micro-Particle Dynamics Including Wall Effects," *Computers and Fluids*, Vol. 33, 2004, pp. 577–601.

[18] Subramanian, V., *Effects of Long-Chain Surfactants, Short-Chain Alcohols and Hydrolyable Cations on the Hydrophobic and Hydration Forces*, Ph.D. thesis, Virginia Polytechnic Institute, 1998.

[19] Currie, I. G., *Fundamentals of Fluid Mechanics*, CRC Press, Boca Raton, FL, 2013.

[20] Issa, R., "Solution of the Implicitly Discretized Fluid Flow Equations by Operator Splitting," *Journal of Computational Physics*, Vol. 62, 1986, pp. 40–65.

[21] Hamaker, H. C., "The London-Van der Waals Attraction Between Spherical Particles," *Physica*, Vol. 4, 1937, pp. 1058–72.

[22] Gregory, J., "Approximate Expressions for Retarded Van der Waals Interaction," *Journal of Colloid and Interface Science*, Vol. 83, 1981, pp. 138–145.

[23] Schenkel, J. and Kitchener, J., "A Test of the Derjaguin-Verwey-Overbeek Theory with a Colloidal Suspension," *Transactions of the Faraday Society*, Vol. 56, 1960, pp. 161–173.

[24] Czarnecki, J., "Van der Waals Attraction Energy Between Sphere and Half-Space," *Journal of Colloid and Interface Science*, Vol. 72, 1979, pp. 361–62.

[25] Gregory, J., "Interaction of Unequal Double Layers at Constant Charge," *Journal of Colloid and Interface Science*, Vol. 51, 1975, pp. 44–51.

[26] Derjaguin, B. and Landau, L., "Theory of the Stability of Strongly Charged Lyophobic Sols and of the Adhesion of Strongly Charged Particles in Solutions of Electrolytes." *Acta Physico Chemica URSS*, Vol. 14, 1941, pp. 633–62.

[27] Verwey, E. J. W. and Overbeek, J. T. G., *Theory of the Stability of Strongly Charged Lyophobic Colloids. The Interaction of Sol Particles Having an Electric Double Layer.*, Elsevier, 1948.

[28] Wu, J., Bratko, D., and Prausnitz, J., "Interaction Between Like-Charged Colloidal Spheres in Electrolyte Solutions," *Proceedings of the National Academy of Sciences of the United States of America*, Vol. 95, 1998, pp. 15169–15172.

[29] Allahyarov, E., D'Amico, I., and Owen, H., "Attraction Between Like-Charged Macroions by Coulomb Depletion," *Physics Review Letter*, Vol. 81, 1998, pp. 1334–1337.