Generalized drag force for particle-based simulations

Christoph Gissler[a,b,*], Stefan Band[a], Andreas Peer[a], Markus Ihmsen[b], Matthias Teschner[a]

[a] *University of Freiburg, Germany*
[b] *FIFTY2 Technology GmbH*

Abstract

Computing the forces acting from a surrounding air phase onto a particle-based fluid or rigid object is challenging. Simulating the air phase and modeling the interactions using a multiphase approach is computationally expensive. Furthermore, stability issues may arise in such multiphase simulations. In contrast, the effects from the air can be approximated efficiently by employing a drag equation. Here, for plausible effects, the parameterization is important but challenging. We present a drag force discretization based on the drag equation that acts on each particle separately. It is used to compute the effects of air onto particle-based fluids and rigid objects. Our presented approach calculates the exposed surface area and drag coefficient of each particle. For fluid particles, we approximate their deformation to improve the drag coefficient estimation. The resulting effects are validated by comparing them to the results of multiphase SPH simulations. We further show the practicality of our approach by combining it with different types of SPH fluid solvers and by simulating multiple, complex scenes.

Keywords: Physically-based animation, Fluid simulation, Smoothed Particle Hydrodynamics, Multiphase

1. Introduction

Since air is normally invisible, liquids tend to look like single-phase, free-surface flows. It is intuitive to assume that liquids like water do not interact with any other phase, as these interactions are often not directly visible. However, the surrounding air influences the behavior of liquids significantly. Rain drops, for example, reach a terminal velocity or otherwise they would hurt a lot.

Although the interactions of liquids with their surrounding air are important, in the Computer Graphics community, single-phase approaches are often preferred. One of the main reasons is that a single-phase liquid simulation is computationally cheaper. Of the two major approaches used to simulate fluids, the Eulerian and the Lagrangian one, the Lagrangian approach is especially well-suited to simulate free-surface, single-phase flows. This is due to its meshless discretization of the fluid volume with moving sample points, which implicitly track the fluid surface. The Lagrangian Smoothed Particle Hydrodynamics (SPH) method is a popular choice (cf. [1]) which we also use in this work. Additionally, we use the Bullet physics library [2] to simulate dynamic rigid objects and the boundary handling scheme proposed by Akinci et al. [3] which samples rigid objects with particles to simulate fluid-rigid interactions.

In the context of SPH, there exist multiple approaches to capture the effects coming from the interaction between the simulated liquid and the surrounding air. Fully simulating the air phase is possible and logically allows to capture all occurring effects. However, the computation of such multiphase simulations is very expensive since a large volume needs to be sampled with air particles to enclose the liquid. Furthermore, stability problems may occur due to the high density ratio between air and liquid.

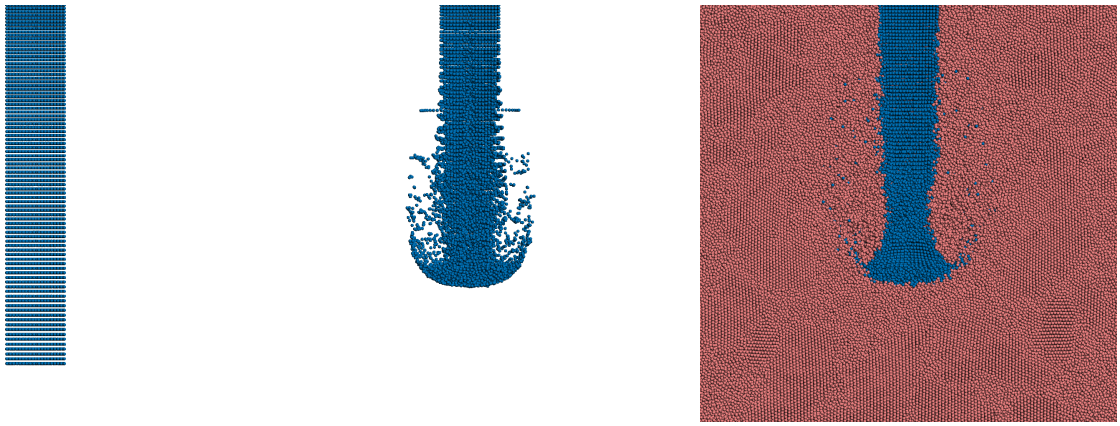
To prevent the cost of simulating the air phase, approaches have been implemented that try to reproduce the observed interaction effects without the need for a fully simulated second phase. Some approaches sample air particles only in regions of interest [4] to capture the behavior of bubbles [5, 6] or spray and foam [7]. These approaches require careful placement and deletion strategies for the air particles and do not model the effects of air acting as a drag force on the surface of the liquid. Simple drag forces are already employed in particle-based fluid simulations, for example in [8]. Accurate parameter estimation is normally not done.

Contribution

This paper is an extended version of [9]. In [9], a drag equation is used to model drag forces acting from an air phase with predefined velocities onto a free-surface SPH fluid. The drag equation is employed to compute one-way coupled forces acting onto the fluid surface. The surface of each fluid particle is calculated and its deformation is modeled to accurately estimate the parameters needed to compute the drag force.

Using the drag equation to model the air-fluid interactions has several advantages. It means that the forces are mainly based on the velocity difference between air and fluid. Accordingly, since explicit pressure values are not computed for the interaction between the phases, typical instabilities that occur due to high density ratios are prevented. Furthermore, the computational cost of the simulation is reduced since the air is not sampled with particles. It is shown that the presented approach

*Corresponding author.
Email addresses: gisslerc@informatik.uni-freiburg.de (Christoph Gissler), bands@informatik.uni-freiburg.de (Stefan Band), peera@informatik.uni-freiburg.de (Andreas Peer), ihmsen@fifty2.eu (Markus Ihmsen), teschner@informatik.uni-freiburg.de (Matthias Teschner)



(a) Single phase simulation. **(b)** Single phase simulation with drag force. **(c)** Multiphase simulation.

Figure 1: Comparison of a single phase simulation without drag force, one with our proposed drag force and a multiphase simulation. Employing our drag force yields a similar result as the multiphase simulation.

produces results that are similar to a full multiphase simulation (see Fig. 1). Realistic scenarios are shown which demonstrate that employing the proposed drag force improves the plausibility of the fluid behavior (see Figs. 2 and 9). Finally, the proposed drag force is combined with different types of SPH fluid solvers.

Since the air volume is not represented by particles, the approach is unable to reproduce effects coming from entrapped air like rising air bubbles. Furthermore, as the air is not simulated and the force is only one-way coupled, effects based on the fluid influencing the air cannot be modeled.

As an extension to [9], we additionally derive an alternative computation of the deformation of a fluid particle and compare both approaches. We show that our approximation results in the same deformation for a single falling fluid particle while requiring less memory. Furthermore, we extend our proposed drag force to also act on rigid objects that are represented by particles. This allows to model the effects of air and wind on rigid bodies. Again, since we do not simulate the air phase, we are unable to reproduce turbulent behavior or lift effects as for example needed to simulate a flying airplane.

Organization

In the next section, we will first discuss related work regarding the proposed drag force. We give a short introduction to the basics of an SPH fluid solver and our used boundary handling in Section 3. In Section 4, we present our method to model the air effects acting on the fluid. This section includes an explanation of the drag equation and detailed sections how we model the droplet deformation and compute the surface area of a particle. In Section 5, we explain the extension of the drag force to air interactions with rigid bodies. In Section 6, we show simulations employing our proposed force. We demonstrate the improvements when using our model by comparing it to simulations without it and to a full multiphase simulation. Finally, we summarize our work and give suggestions for future work in Section 7.

2. Related work

For the simulation of single-phase, free surface liquids, SPH was first introduced to the Computer Graphics community by Müller et al. [10]. Since then, a lot of research has been done to improve the performance and extend the possible effects that can be simulated with SPH. There exist multiple overviews regarding SPH [11] and its use for fluid simulation [1].

There are extensions for SPH to simulate multiple phases and their interaction. In the Computer Graphics community, Müller et al. [5] stress the importance of modeling the interaction between a liquid phase and the surrounding air and first proposed an SPH multiphase formulation. However, due to density discontinuities across the interface, spurious tension effects and large gaps occur between phases. Replacing the usage of density in the SPH formulation with the number density reduces this problem [12, 13]. Nevertheless, using a full multiphase simulation to model the effects between air and liquid is challenging. For a multiphase simulation, a large domain surrounding the liquid has to be sampled with air particles. This is computationally expensive. Furthermore, the high density ratio between air and liquid leads to high accelerations of the air phase. To get a stable simulation, small time steps are needed. Additionally, as detailed in [13, 14], SPH particles order themselves in stable lattice structures. This damps the buoyancy of single particles submerged inside another phase [6].

Instead of fully simulating the second phase, approaches exist that only generate air particles in regions of interest. Most of the work deals with simulating entrapped air bubbles. For example, in [5], air particles are generated at the liquid-air interface and deleted if they do not interact with liquid anymore. The interactions between the air and liquid particles are computed with a multiphase SPH model. When the air particles are generated, their velocity is set to the velocity of the surrounding liquid particles. This means that no strong pressure or friction forces occur between the phases. To prevent stability problems, Müller et al. [5] increase the density of air particles to reduce the density ratio between the phases. Additionally, they introduce an artificial buoyancy force such that the entrapped air parti-

cles behave plausible. Similarly, Ihmsen et al. [6] generate air particles on the fly to model bubbles. Instead of using a multiphase SPH model like in [5], all interaction forces are computed based on velocity differences between the phases. This brings improvements when dealing with high density ratios. Similarly, our approach also only relies on the velocity difference between the phases to compute the interaction forces. One of the coupling forces used in [6] is a drag force. It scales linearly in the velocity difference and is used to model friction forces between the phases. Rising bubbles are also modeled in [15, 16]. Here, a drag force is used to couple the bubbles with the surrounding liquid, too. Both the Stokes law and the drag equation are used in [15] to compute the drag force based on the Reynolds number of the bubble. In contrast to our work, in [15], the drag coefficient needed for the drag equation is set to a constant value and the deformation of single particles is not modeled. In [4], similar to [5], air particles are only generated near the liquid-air interface. However, in [4], this is not done to model entrapped air. Instead, the air particles are used to improve the SPH density estimation of particles at the surface of the fluid. No pressure or viscosity forces are evaluated between the phases. In contrast to the aforementioned approaches, which either improve the density estimation or model entrapped air, our approach tries to model the drag effects of air acting on a liquid's surface. Also, our approach does not generate any air particles. Our used drag equation is similar to the one in [15]. We do, however, take special care to compute the drag coefficient and to model particle deformation.

Drag forces acting from the air onto the surface of another material are often used in animation to improve cloth simulations. Bhat et al. [17] combine a linear relationship between velocity differences and forces for tangential areas and a quadratic one for areas exposed in normal direction to the air. They state that the quadratic term is needed in order to capture more realistic effects. Wilson et al. [18] also use a drag equation to model the effects of air on clothes. In their work, the drag coefficient is controlled by the artist using their simulation tool. In their unified, position-based simulation framework [8], Müller et al. also model drag effects on clothes. They employ the aerodynamic forces proposed in [19]. These approaches compute the drag force on the triangles of the respective mesh whereas we compute the drag force on its boundary particles. Müller et al. [8] also use a simple drag force to model the interaction of air with smoke, which is discretized with particles. As a condition when to apply the drag force they use a simple density-based approach, assuming that surface particles have a lower density than internal particles. Their drag force is linear in the velocity difference, modeling friction forces. In order to scale the effect of the force, they use a user-defined parameter. In contrast to their work, we use a drag term that is quadratic in the velocity difference and we take special care to automatically compute all necessary parameters in order to keep user input to a minimum.

3. SPH-based fluid solver

A common approach for SPH-based fluid solvers is to solve the Navier-Stokes equation in order to compute an acceleration for each fluid particle i:

$$\frac{\mathrm{D}\mathbf{v}_i}{\mathrm{D}t} = -\frac{1}{\rho_i}\nabla p_i + \nu \nabla^2 \mathbf{v}_i + \frac{\mathbf{F}_i^{\text{external}}}{m_i}\,. \tag{1}$$

SPH is used to discretize the spatial derivatives in Eq. (1). In order to calculate these derivatives, SPH requires the neighboring particles of each particle.

To compute the pressure gradient and – following – the acceleration due to pressure, the pressure value must be computed for all particles. The pressure acceleration should prevent compression of the fluid. Accordingly, the pressure depends on the density, which is also computed using SPH. Using this density, the pressure can be calculated using an equation of state (EOS) as it is done, for example, for WCSPH [20]. Iterative EOS solvers allow to use larger time steps [21, 22]. Finally, a splitting scheme can be employed, leading to a pressure Poisson equation (PPE). The PPE is solved to get the pressure value. IISPH [23] is an example for this type of pressure solver.

There are also several possibilities for evaluating the acceleration due to viscosity. One way is to compute it explicitly. In this case, there exist various options for discretizing the Laplacian with SPH. In [24], for example, a combination of a finite difference and the kernel gradient is used to improve its robustness with regard to particle disorder. Alternatively, the viscosity acceleration can also be solved using an implicit scheme, examples are [25, 26, 22].

Finally, Eq. (1) contains the external forces $\mathbf{F}_i^{\text{external}}$ which — for example — consist of the gravity. For fluids, we add our proposed drag force to these external forces. This allows us to easily integrate our force into different types of SPH solvers.

For boundary handling, we use the approach by Akinci et al. [3] which samples rigid objects with boundary particles. This allows to compute pressure forces between fluid and boundaries. Rigid-rigid interactions are handled with the Bullet physics library [2]. We apply our proposed drag force for rigid objects on the same boundary particles which are created for the fluid-boundary interactions.

4. Fluid-air interactions

We want to model the forces acting from the surrounding air onto a free-surface liquid. These interactions are defined by pressure, friction and adhesion forces. We do not model the effects of adhesion as it can be approximated by adjusting the surface tension parameters of the liquid. For low relative velocities between air and liquid, drag effects can be modeled using a linear relationship between the relative velocity and the acting forces. The corresponding formula is called the Stokes' law. For higher velocities, a quadratic relationship between relative velocity and force is needed. We focus on modeling the drag effects using this quadratic relationship. The drag equation is a general formula to model these forces acting on an object i moving through the air [27]:

$$\mathbf{F}_i^{\text{drag}} = \frac{1}{2}\rho_a \mathbf{v}_{i,\text{rel}}^2 C_{D,i} A_i\,. \tag{2}$$

ρ_a is the density of the air. $\mathbf{v}_{i,\text{rel}}^2$ is a vector pointing in the direction of the relative velocity difference between air and the object i, while its length is given as the squared length of this velocity difference. It is computed as follows:

$$\mathbf{v}_{i,\text{rel}}^2 = |\mathbf{v}_a - \mathbf{v}_i|^2 \frac{\mathbf{v}_a - \mathbf{v}_i}{|\mathbf{v}_a - \mathbf{v}_i|} = |\mathbf{v}_a - \mathbf{v}_i| (\mathbf{v}_a - \mathbf{v}_i) \ .$$

$\mathbf{v}_i$ is the velocity of particle i and $\mathbf{v}_a$ is the air velocity. In our work, the air velocity is predefined for every position in space. The drag coefficient $C_{D,i}$ and the exposed cross-sectional area A_i vary for each object. They allow to tune the model for differently shaped objects.

Eq. (2) is normally used to compute the aerodynamic drag of rigid objects such as airplanes or cars. It builds upon the dynamic pressure $q = \frac{1}{2}\rho |\mathbf{v}|^2$ and models the molecules of the air phase hitting the object and being stopped in the process. When assuming that all air molecules come to a complete stop when hitting the surface, the drag coefficient $C_{D,i}$ should be chosen as 1. Depending on the shape of the surface, however, not all molecules of the air come to a stop. Accordingly, the drag coefficient varies depending on the shape of the considered object.

Due to its general nature, the drag equation can also be used to compute the drag force acting onto a liquid like water. As an additional complexity compared to rigid objects, a liquid changes its shape continuously. To compute the drag forces acting on an SPH liquid, we compute Eq. (2) independently for each particle. Accordingly, we need to estimate their drag coefficient $C_{D,i}$ and their exposed cross-sectional surface area A_i.

A single particle is the smallest discretized element in an SPH simulation. To more accurately estimate the surface area, we additionally model the deformation of particles. This deformation influences the drag coefficient as well as the cross-sectional area of a particle. To explain the different parts leading to the final computation of Eq. (2), this section is structured as follows: First, as the other parts depend on it, we explain how we model the deformation of a particle. Then, in Subsection 4.2, we use this deformation to compute the drag coefficient for each particle. The exposed cross-sectional area of a particle depends on two parts. First, the unoccluded area of the particle in the direction of the air, explained in Subsection 4.3, and second, the occlusion of a particle by other liquid particles or a rigid boundary. We present our occlusion computation method in Subsection 4.4. Finally, in Subsection 4.5, we summarize the subsections and combine all formulas to compute the drag force acting on a particle.

4.1. Particle deformation

We want to compute the deformation of a fluid particle due to the drag forces from the surrounding air. If a droplet consists of multiple SPH particles, its overall deformation is represented by the position of the particles with respect to each other. Therefore, we consider every particle on its own and do not try to change or model the overall structure of multiple fluid particles that are clustered together.

4.1.1. Theory

To model the deformation of a particle, we follow the Taylor-Analogy-Breakup (TAB) model that was initially proposed in [28] to model engine spray droplet breakup. A fluid droplet in the air oscillates with time t [29, 30]. The TAB model is based on the Taylor analogy [31] which relates this oscillation to a mass-spring system. In [28], the deformation of a droplet is described by the formula of a damped harmonic oscillator. The external force, the spring force and the damping force are dependent on the phase of the droplet and the surrounding air phase. Constants C_F, C_k and C_d are introduced to scale these external, spring and damping forces. An additional constant C_b is used to convert the displacement to a dimensionless deformation y_i. This current deformation y_i of a particle i is then a value between 0 and 1. A value of 0 means there is no deformation (droplet is a sphere) and a value of 1 corresponds to a deformed droplet (droplet is a disk). Accordingly, in [28], the change of y_i over time is described by the following second order ordinary differential equation (ODE):

$$\frac{\mathrm{d}^2 y_i}{\mathrm{d}^2 t} = \frac{C_F}{C_b}\frac{\rho_a}{\rho_l}\frac{\left|\mathbf{v}_{i,\text{rel}}^2\right|}{L^2} - \frac{C_k \sigma}{\rho_l L^3} y_i - \frac{C_d \mu_l}{\rho_l L^2}\frac{\mathrm{d}y_i}{\mathrm{d}t} \ . \tag{3}$$

ρ_a and ρ_l are the rest densities of the air and the liquid. σ is the surface tension coefficient of the liquid and μ_l is the dynamic viscosity of the liquid. By default, we set these values according to literature to $\rho_a = 1.2041$, $\rho_l = 1000$, $\sigma = 0.0724$ and $\mu_l = 0.00102$. For the constants C_F, C_k and C_d, we adopt the values proposed in [28] which are $C_F = \frac{1}{3}$, $C_k = 8$, $C_d = 5$. C_b is set to $\frac{1}{2}$. Finally, L is the radius of the droplet. Since we consider each particle on its own, we take the radius of a sphere corresponding to the volume h^3 of a particle. It is calculated as:

$$\begin{aligned} h^3 &= \frac{4}{3}\pi L^3 \\ \iff L &= \sqrt[3]{\frac{3}{4\pi}} h \ . \end{aligned} \tag{4}$$

4.1.2. Implementation

We want to compute the deformation of each particle at each point in time during the simulation based on Eq. (3). There are multiple options for this. In the following, we first show a direct discretization of the ODE with a semi-implicit update scheme. Then, as an alternative, we show an approximate discretization which does not require to store additional values and thus requires less memory. The results of both methods are compared in Subsection 6.2.

Direct discretization

We discretize Eq. (3) directly by integrating it in time. We first introduce the change of y_i as $v_{y,i}$. Equation (3) reads then

$$\frac{\mathrm{d}v_{y,i}}{\mathrm{d}t} = \frac{C_F}{C_b}\frac{\rho_a}{\rho_l}\frac{\left|\mathbf{v}_{i,\text{rel}}^2\right|}{L^2} - \frac{C_k \sigma}{\rho_l L^3} y_i - \frac{C_d \mu_l}{\rho_l L^2} v_{y,i} \ . \tag{5}$$

We discretize Eq. (5) with a simple forward difference. By employing a semi-implicit update scheme, this results in the following two formulas for computing y_i and $v_{y,i}$:

$$\begin{aligned} v_{y,i}^{t+\Delta t} &= v_{y,i}^t + \Delta t \left(\frac{C_F}{C_b}\frac{\rho_a}{\rho_l}\frac{\left|\mathbf{v}_{i,\text{rel}}^2\right|}{L^2} - \frac{C_k \sigma}{\rho_l L^3} y_i^t - \frac{C_d \mu_l}{\rho_l L^2} v_{y,i}^t \right) , \\ y_i^{t+\Delta t} &= y_i^t + \Delta t v_{y,i}^{t+\Delta t} \ . \end{aligned} \tag{6}$$

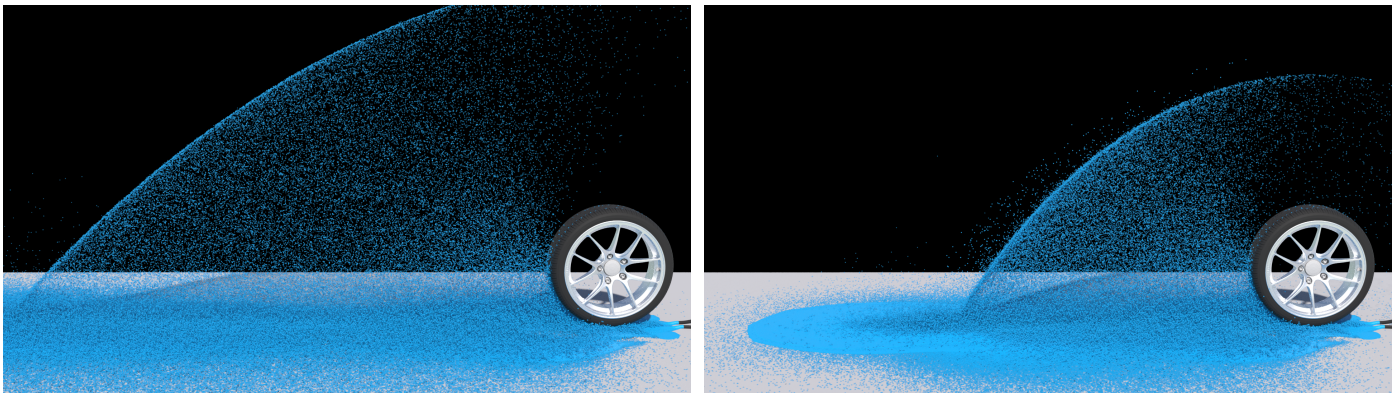


Figure 2: A wheel rotating with a speed corresponding to $20\,\mathrm{km\,h^{-1}}$. When not using our drag force (left), the particles are sprayed very far. Applying our drag force (right), the particles reach a terminal velocity and the spray distance matches real-world experiments.

Using Eq. (6) to compute the deformation of particles accordingly requires to store two additional scalar values per particle.

Approximate discretization

To prevent the memory overhead of the direct discretization shown above and to simplify the implementation, we make the simplifying assumption that the deformation of the droplet does not oscillate. Instead, we assume that the particle is immediately completely deformed based on the current relative velocity. With this assumption, the ODE in Eq. (3) simplifies as follows:

$$\underbrace{\frac{\mathrm{d}^2 y_i}{\mathrm{d}^2 t}}_{=0} = \frac{C_F}{C_b}\frac{\rho_a}{\rho_l}\frac{\left|\mathbf{v}_{i,\mathrm{rel}}^2\right|}{L^2} - \frac{C_k\sigma}{\rho_l L^3} y_i - \underbrace{\frac{C_d\mu_l}{\rho_l L^2}\frac{\mathrm{d}y_i}{\mathrm{d}t}}_{=0} \qquad \text{(3 revisited)}$$

$$\implies 0 = \frac{C_F}{C_b}\frac{\rho_a}{\rho_l}\frac{\left|\mathbf{v}_{i,\mathrm{rel}}^2\right|}{L^2} - \frac{C_k\sigma}{\rho_l L^3} y_i^{\mathrm{approx}} . \qquad (7)$$

In Eq. (7), everything apart from the relative velocity is independent of the specific particle i. We therefore combine these variables:

$$y_i^{\mathrm{approx}} = \left|\mathbf{v}_{i,\mathrm{rel}}^2\right| \underbrace{\frac{C_F}{C_k C_b}\frac{\rho_a L}{\sigma}}_{y_{\mathrm{coeff}}} = \left|\mathbf{v}_{i,\mathrm{rel}}^2\right| y_{\mathrm{coeff}} . \qquad (8)$$

By pre-computing y_{coeff}, the current deformation of each particle is calculated using a simple multiplication. The deformation computed by Eq. (8) corresponds to the converged deformation value of Eq. (3). For a constant relative velocity, Eq. (3) converges to this value due to the viscosity which dampens the oscillation.

In [28], if a droplet reaches a deformation of 1, it is assumed to break up into multiple smaller droplets. Since one particle is the smallest discretization unit in our SPH simulation, we clamp values larger than 1:

$$y_i^{\mathrm{approx}} = \min\left(1, \left|\mathbf{v}_{i,\mathrm{rel}}^2\right| y_{\mathrm{coeff}}\right) \qquad (9)$$

4.2. Drag coefficient

Based on the deformation formula in [28], Liu et al. [32] propose a model to compute the drag coefficient of a droplet. The idea is to use the deformation y to linearly interpolate the drag coefficient between the value of a sphere and the value of a disk. We adapt the formula from [32] to use our calculated approximated deformation y_i^{approx} instead of the current deformation $y_i(t)$. The drag coefficient is then computed as:

$$C_{D,i}^{\mathrm{Liu}} = C_{D,i}^{\mathrm{sphere}}\left(1 + 2.632 y_i^{\mathrm{approx}}\right), \qquad (10)$$

where $C_{D,i}^{\mathrm{sphere}}$ is the drag coefficient of a sphere. It is calculated as:

$$C_{D,i}^{\mathrm{sphere}} = \begin{cases} \frac{24}{Re_i}\left(1 + \frac{1}{6} Re_i^{\frac{2}{3}}\right), & Re_i \le 1000 \\ 0.424, & Re_i > 1000 \end{cases} \qquad (11)$$

Re_i is the Reynolds number for particle i, which is computed as:

$$Re_i = 2\frac{\rho_a \left|\mathbf{v}_{i,\mathrm{rel}}\right| L}{\mu_a} . \qquad (12)$$

We choose the value of the dynamic viscosity of the air as $\mu_a = 0.00001845$.

Finally, as we want a fluid particle which is part of a larger cluster of particles to have a drag coefficient of 1, we linearly interpolate between Eq. (10) and 1 based on the number of fluid neighbors n:

$$C_{D,i} = \left(1 - \frac{\min\left(\frac{2}{3}n^{\mathrm{full}}, n\right)}{\frac{2}{3}n^{\mathrm{full}}}\right) C_{D,i}^{\mathrm{Liu}} + \frac{\min\left(\frac{2}{3}n^{\mathrm{full}}, n\right)}{\frac{2}{3}n^{\mathrm{full}}} . \qquad (13)$$

n^{full} is the number of neighbors a particle with a full neighborhood has. In our simulations, we choose $n^{\mathrm{full}} = 38$. If a particle has $\frac{2}{3}n^{\mathrm{full}}$ or more neighbors, we assume it is part of a larger surface. In this case, its drag coefficient is 1. With no neighbors, the drag coefficient $C_{D,i}^{\mathrm{Liu}}$ from Eq. (10) is used.

4.3. Particle area

In this subsection, we explain how we compute the cross-sectional area of a deformed particle. Together with the occlusion detailed in the next subsection, this results in the area A_i used in the drag equation.

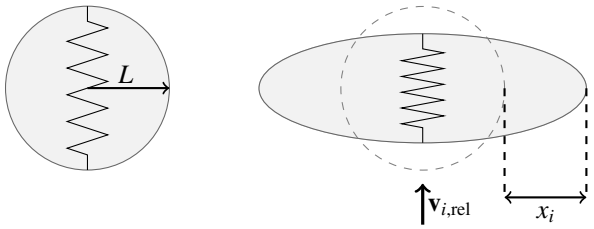


Figure 3: Illustration of the deformation of a single particle. When the relative velocity $\mathbf{v}_{i,\text{rel}}$ between air and particle is zero (left), the particle is assumed to be spherical. When $\mathbf{v}_{i,\text{rel}}$ increases, the particle deforms to a disk (right).

In SPH, the volume of a single particle is normally computed as the cube of the spacing h: $V_i = h^3$. This suggest to use the squared spacing as cross-sectional area of a particle:

$$A_i^{\text{square}} = h^2 \,. \tag{14}$$

This assumption works well for fluid particles that are part of larger, closed surface of the fluid. In this case, the pressure force results in equally-spaced particles on the surface, meaning that Eq. (14) results in an approximately correct surface area. When a fluid particle has no or only a few neighboring particles, this approximation is bad. A single fluid droplet forms a sphere as long as no external forces act on it. This is due to the surface tension. If a surface force like the drag force acts on a droplet, the droplet deforms and more resembles a disk than a sphere (see Fig. 3). To compute the correct cross-sectional area for these particles, we again use the deformation value y_i^{approx} computed in Subsection 4.1.

The cross-sectional area of a spherical droplet with radius L is computed as $A^{\text{sphere}} = \pi L^2$. Following [32], the radius increase of a deformed droplet is calculated as:

$$\begin{aligned} y_i^{\text{approx}} &= \frac{x_i}{C_b L} \\ \implies x_i &= C_b L y_i^{\text{approx}} \,. \end{aligned} \tag{15}$$

The radius of the deformed droplet is then $L + C_b L y_i^{\text{approx}}$. This results in the following cross-sectional area of the disk-like droplet:

$$A_i^{\text{droplet}} = \pi \left(L + C_b L y_i^{\text{approx}} \right)^2 \,. \tag{16}$$

Finally, we linearly interpolate between Eq. (14) and Eq. (16) based on the number of neighbors n. Similar to Eq. (13), with $\frac{2}{3} n^{\text{full}}$ neighbors, the area corresponds to the square area, while with no neighbors we use A_i^{droplet}. This results in the cross-sectional area of a particle:

$$A_i^{\text{unoccluded}} = \left(1 - \frac{\min\left(\frac{2}{3} n^{\text{full}}, n\right)}{\frac{2}{3} n^{\text{full}}} \right) A_i^{\text{droplet}} + \frac{\min\left(\frac{2}{3} n^{\text{full}}, n\right)}{\frac{2}{3} n^{\text{full}}} h^2 \,. \tag{17}$$

We marked the area as unoccluded since we get the final area A_i used in Eq. (2) by combining Eq. (17) with a scaling based on the occlusion of the particle. This is explained in the next subsection.

4.4. Particle occlusion

The drag forces should only act on particles that are exposed to the air, i.e., particles that are on the surface of the liquid. Additionally, only particles with a surface area facing toward $\mathbf{v}_{i,\text{rel}}$ should be influenced. To compute the occluded surface area of a particle, we first calculate the unoccluded area $A_i^{\text{unoccluded}}$ as detailed in Subsection 4.3. This area is then weighted with an occlusion value w_i. The value is between 0 and 1 and states the fraction of the particle area that is occluded,

$$A_i = w_i A_i^{\text{unoccluded}} \,. \tag{18}$$

A surface detection algorithm is needed to compute w_i. One common approach to detect surface particles of an SPH fluid is to compute a color field [33, 10]. Surface particles are detected based on the magnitude of the gradient of this color field, which can be interpreted as the normal of the fluid. We noticed that this approach works well to simply detect surface particles. In our case, we want to additionally compute the ratio of exposed surface area of a single particle in the direction of the relative velocity between the particle and the air. This allows us to partially apply the drag force on these particles. Using the fluid color gradient for this proved to be difficult. For droplets with only a few particles the normal direction tends to vary strongly depending on the particle configuration. Therefore, we use a more geometrically inspired approach to compute the surface particles and their occlusion. Our method is similar to the idea used in [34] and [35].

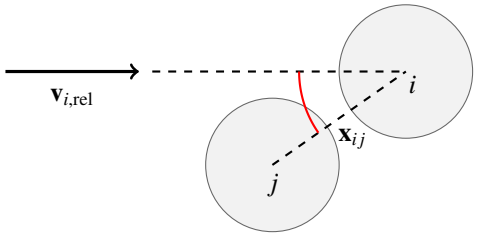


Figure 4: Computation of the occlusion is based on the angle between $\mathbf{v}_{i,\text{rel}}$ and the direction to neighboring particles $\mathbf{x}_{ij}$.

The algorithm in [34] and [35] computes a cover vector $\mathbf{b}_i$ and checks if there is a neighboring particle in a cone in the direction of this vector. If not, the particle is marked as being on the surface. We are interested in the occlusion in the direction of $\mathbf{v}_{i,\text{rel}}$ and therefore use this direction instead of the cover vector.

Furthermore, we do not declare a particle as fully covered or uncovered if it has a neighbor inside the scan cone. Instead, we use the cosine of the angle between $\mathbf{v}_{i,\text{rel}}$ and the neighbor direction $\mathbf{x}_{ij}$ (Fig. 4). For multiple neighbors, we use the maximum of this value. This results in a value between 0 and 1 which indicates the amount of occlusion of the particle. To get a value of 1 if the particles is not occluded at all, the occlusion value w_i for Eq. (18) is computed as shown below. We clamp the value to prevent values larger than 1 or smaller than 0.

$$w_i = \max\left(0, \min\left(1, 1 - \max_j \left(\frac{\mathbf{v}_{i,\text{rel}}}{\left|\mathbf{v}_{i,\text{rel}}\right|} \cdot \frac{\mathbf{x}_{ij}}{\left|\mathbf{x}_{ij}\right|} \right)\right)\right) \,. \tag{19}$$

4.5. Summary

Using the values from the previous subsections allows to compute Eq. (2). Algorithm 1 gives an overview over the order of steps.

The drag force is computed in the beginning of a simulation step since the drag force is added as external force. We require neighbor information for each particle to exist. Since these are necessary anyway for an SPH solver, this imposes no additional computational cost.

Algorithm 1 Drag force computation.

compute y_{coeff} using Eq. (8)
for all particle i **do**
  compute deformation: $y_i^{\text{approx}} = \min\left(1, \left|\mathbf{v}_{i,\text{rel}}^2\right| y_{\text{coeff}}\right)$ (cf. Eq. (9))
  compute drag coefficient $C_{D,i}$ using Eq. (13)
  compute unoccluded area $A_i^{\text{unoccluded}}$ using Eq. (17)
  compute occlusion w_i using Eq. (19)
  compute exposed area: $A_i = w_i A_i^{\text{unoccluded}}$ using Eq. (18)
  compute drag force: $\mathbf{F}_i^{\text{drag}} = \frac{1}{2}\rho_a \mathbf{v}_{i,\text{rel}}^2 C_{D,i} A_i$ using Eq. (2)

The only parameters required by the formulas in the algorithm are the surface tension and viscosity of the liquid as well as the rest density of the air and the liquid. These values can be taken from literature. We use the values shown in Subsection 4.1.1.

5. Rigid-air interactions

In the previous section, we explain how to compute the drag coefficient and exposed area of a deforming fluid particle in order to calculate the acting drag force. In this section, we extend our proposed drag force to also act on rigid bodies. In our SPH framework, interactions between rigid bodies are simulated with the Bullet physics library [2]. For fluid-boundary interactions, the rigid bodies are sampled with a single layer of particles as proposed by Akinci et al. [3]. We use these existing boundary particles to compute the drag force acting on a rigid body. Accordingly, the drag force we apply to a single boundary particle i reads the same as for a fluid particle:

$$\mathbf{F}_i^{\text{drag}} = \frac{1}{2}\rho_a \mathbf{v}_{i,\text{rel}}^2 C_{D,i} A_i \,. \tag{2 revisited}$$

The two unknown variables in Eq. (2) are again the drag coefficient $C_{D,i}$ and the exposed area A_i. Boundary particles do — in contrast to fluid particles — not deform. Therefore, we do not need to compute a deformation. Since a boundary particle is always part of a larger surface area, we set the drag coefficient to 1. This is a simplification that disregards the fact that for example a simple cuboid has a drag coefficient different from 1 because of air turbulence. This is a limitation of our approach which is challenging to fix since we do not compute the air flow but use predefined air velocities. This leaves only the surface area A_i of a single boundary particle i to be computed. Similar as for fluid particles, we compute the area of the particle and its occlusion in direction of the relative velocity $\mathbf{v}_{i,\text{rel}}$. The two steps are explained in the following two sections.

5.1. Particle area

As long as the rigid object is uniformly sampled with boundary particles, the surface area of a particle can be assumed to be the squared spacing $A_i = h^2$. However, one of the benefits of using the boundary handling approach by Akinci et al. [3] is that it also works with non-uniformly sampled meshes. This simplifies the boundary particle generation process and allows for overlapping geometries. When a rigid object is sampled non-uniformly, some areas may be oversampled compared to others. Similar as in [3], we compute a scalar value indicating the oversampling of a boundary region which allows us to scale the surface area of a particle. First, the number density of boundary particle i is computed as

$$\delta_i = \sum_b W_{ib} \,, \tag{20}$$

where b are the boundary neighbors of i. This value is larger for oversampled areas of the boundary. By taking the inverse, we get the scaling factor $\psi_i = \frac{1}{\delta_i}$. Since we used a three-dimensional kernel to compute the scaling factor but the surface of the rigid is only sampled in two dimensions, we additionally multiply this value by 0.7 as motivated in [36]. The reason is that only 70% of the three-dimensional kernel volume is filled by a two dimensional sampling but it should result in a scaling value of 1. Summarized, we compute the unoccluded area of a boundary particle as:

$$A_i^{\text{unoccluded}} = \min\left(1, 0.7\psi_i\right) h^2 \,. \tag{21}$$

Before using this area in Eq. (2), the occlusion of the particle must be computed as detailed in the next section.

5.2. Particle occlusion

Similar as for fluid particles, we estimate how much of the surface area of a boundary particle is affected by the drag force by approximating its occlusion. Additionally to using the angle with respect to its neighbors as done for fluid particles (cf. Subsection 4.4), we use the mesh normal $\mathbf{n}_i$ at the corresponding location. This results in the following formula:

$$w_i = \begin{cases} \max\left(0, \min\left(1, 1 - \max_j\left(\frac{\mathbf{v}_{i,\text{rel}}}{|\mathbf{v}_{i,\text{rel}}|}\frac{\mathbf{x}_{ij}}{|\mathbf{x}_{ij}|}\right)\right)\right), & \mathbf{n}_i \cdot \mathbf{v}_{i,\text{rel}} < 0 \\ 0, & \text{else} \end{cases} \tag{22}$$

The additional condition based on the outwards-pointing mesh normal is needed since we only sample a single layer of boundary particles on the surface of a rigid object. Accordingly, without the additional check, exposed surface would be detected on the inside of a rigid object.

6. Results

In this section, we show the properties of our drag force approach and demonstrate its effects on fluid behavior and on rigid objects. In the presented experiments, we combine our force with different types of SPH solvers. Independent of the type

of solver, we employ a cubic spline kernel [11] with an influence radius of twice the particle spacing. Boundary handling between liquid and rigid objects with one-way and two-way coupling is implemented according to [3]. Surface tension is modeled with [37].

The scenes are simulated on a 16-core Intel Xeon workstation with 3.1 GHz and 128 GB of RAM. Apart from the multiphase comparison, the wiper scene and the pendulum scene which are rendered in our own SPH framework, the images are ray-traced with Houdini [38]. If not noted otherwise, we use the drag force parameters shown in Section 4 and a constant air velocity of zero.

6.1. Comparison with multiphase simulation

Figure 1 shows the scenario we use to compare the visual effects resulting from our approach with a single-phase simulation without our approach and a full multiphase simulation. For all three simulations, IISPH [23] is used as a pressure solver. To improve the multiphase interactions, we employ a number-density-based formulation [13]. We set the density for the gas phase to $50\,\mathrm{kg\,m^{-3}}$ and for the liquid phase to $1000\,\mathrm{kg\,m^{-3}}$. The particle spacing is 1 cm. A circular source with a diameter of 19 cm continuously emits the liquid with an inflow velocity of $4\,\mathrm{m\,s^{-1}}$. During the simulation, up to $100k$ liquid particles are created. For the multiphase simulation, we filled a box of size $(1, 2, 1.4)$ m with air particles surrounding the source. We chose the box extents as small as possible such that the boundary of the box did not directly influence the liquid behavior. Filling the box required $2.7M$ air particles.

Comparing the simulations illustrates that with our approach a behavior similar to the multiphase simulation is achieved. The front of the liquid buckles outwards in the drag-force-based and the multiphase simulation. In contrast to the single-phase simulation, the front of the liquid is slowed down resulting in a similar location of the liquid front in both simulations. Still, there are differences between the simulation using our drag force and the multiphase simulation. In the multiphase simulation, the air surrounding the liquid starts to spin. This turbulence in the air also affects the liquid leading to a more irregular sampling of the particles. Furthermore, larger air vortices form behind the front of the liquid. The width of the liquid column shrinks as a result of this. In contrast, our drag force does not simulate the air phase or its turbulence. The sampling of the particles of the liquid phase therefore looks more regular. The computation of the simulation with our drag force is significantly cheaper than the multiphase simulation since no air particles are needed. Simulating 350 frames with one frame corresponding to 0.0001 s took 30 min for the multiphase simulation. Both single-phase simulations took 22 s.

6.2. Reaching of terminal velocity

Single particle drop

For comparing the direct and approximate discretization of the deformation ODE (Eq. (3)) and to verify that a droplet reaches a correct terminal velocity, we simulate a single liquid particle. The diameter of the liquid particle is 5 mm. It is accelerating downward due to a gravity of $9.81\,\mathrm{m\,s^{-2}}$. Pressure, viscosity and cohesion forces are not acting on the particle since it has no neighboring particles.

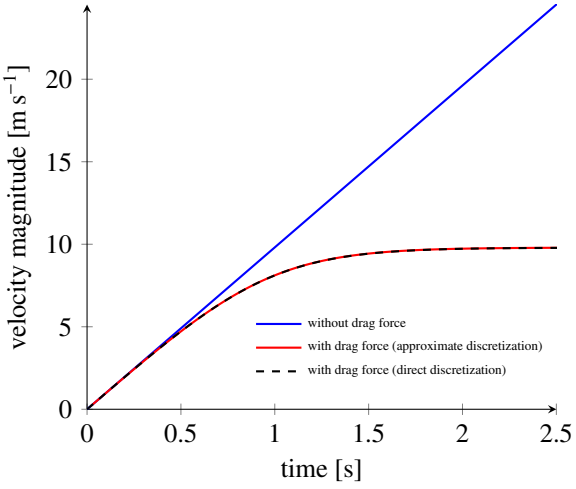


Figure 5: Velocity profile of a single particle with a diameter of 5 mm being accelerated due to gravity. When employing our drag force, the particle reaches a terminal velocity which matches the expected one.

Figure 5 shows a graph of the velocity profile of the particle. When only affected by the gravity, the particle accelerates indefinitely. Applying our proposed drag force, the particle reaches a terminal velocity when the gravity and drag force cancel each other out. The terminal velocity of the particle affected by our drag force matches the values measured for falling raindrops of the same size [39, 40]. Fig. 6 shows the deformation of the liquid particle over time. It shows that our approximate computation of the deformation results in the same values as the direct discretization of Eq. (3). Accordingly, it is also clear that the velocities in Fig. 5 for the approximate discretization and direct discretization are the same.

Note, that both discretizations only result in the same deformation value as long as the relative velocity $\mathbf{v}_{i,\mathrm{rel}}$ is only changing slowly. In contrast to the approximate discretization, the direct discretization is able to capture the oscillation of the deformation of a particle. This means, that if the relative velocity for a single particle suddenly and rapidly changes, the computed deformation values may vary initially.

Tire testing

The results of simulations are closer to reality when particles reach their correct terminal velocity. Figure 2 shows a tire testing setup: A wheel rotates on a large cylinder and water is flowing in with $33.36\,\mathrm{L\,min^{-1}}$ from the right side. The rotation speed of the wheel corresponds to $20\,\mathrm{km\,h^{-1}}$. Due to friction and adhesion between the water and the tire, the water is sprayed away from the tire. In this scene, the water is simulated with WCSPH [20] and the particle spacing is set to 1 mm. During the 8 s of simulated time, up to $3.2M$ particles are generated. Without our drag force, the particles do not reach a terminal velocity. In contrast, using our proposed drag force, the particles reach a terminal velocity and the spray distance is more plausible.

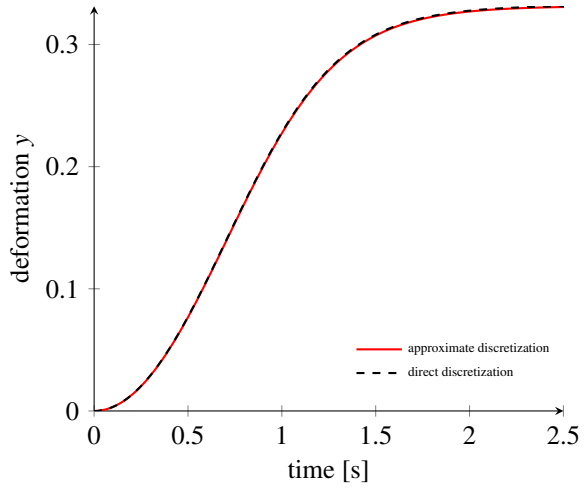


Figure 6: Deformation value of a single falling particle over time.

6.3. Combination with Different SPH Solvers

Since our force is added as an external force, it can be combined with different types of SPH solvers. Regarding fluids, we integrated our force into WCSPH [20] for the simulation of Fig. 2. The simulations in Figs. 1 and 9 use IISPH [23]. We also added our drag force to a viscous fluid SPH solver and an SPH solver for deformable objects as described in the following.

Viscous SPH

For the simulation of the viscous fluid, we employ the implementation described by Peer et al. [26]. The scene is shown in Fig. 7. A bunny is shot with a cannon ball made out of viscous fluid. The cannon ball consists of 800k particles with a particle spacing of 1 mm. It is shot with a speed of 10 m s^{-1}. The viscosity parameter is set to $\xi = 0.9$ and the time step is 0.25 ms resulting in 8 simulation steps per frame for a frame rate of 500 frames/s. The cannon ball deforms due to the drag forces acting on it. Since we compute an occlusion for each particle, the drag force only acts on the front of the cannon ball.

Deformable SPH

We also combined our drag force with an SPH solver capable of simulating deformable objects. For the solver implementation, we followed the method by Ganzenmüller [41]. The approaches by Becker et al. [42] or Solenthaler et al. [43] would also be possible choices. We simulated a deformable Armadillo. You can see the scene in Fig. 8. The Armadillo is volumetrically sampled with particles which have a diameter of 1 cm. Wind is blowing upward with a speed of 80 m s^{-1}. To prevent the Armadillo from leaving the plane, we fixed the lowest particle layer. We used a time step of 0.2 ms and deformation parameters $\mu = 5000$ and $\lambda = 75000$.

As shown in Fig. 8, it is possible to produce interesting effects, although it is hard to verify the realism of the results.

6.4. Coupling with precomputed air flow

Figure 9 shows a wiper simulation. The scene contains a lot of animated, complex-shaped geometry. The liquid is simulated with IISPH [23] with a particle spacing of 1.25 mm. Up to 1.9M particles are simulated. Instead of assuming a constant air velocity everywhere, a grid of air velocities is imported from a

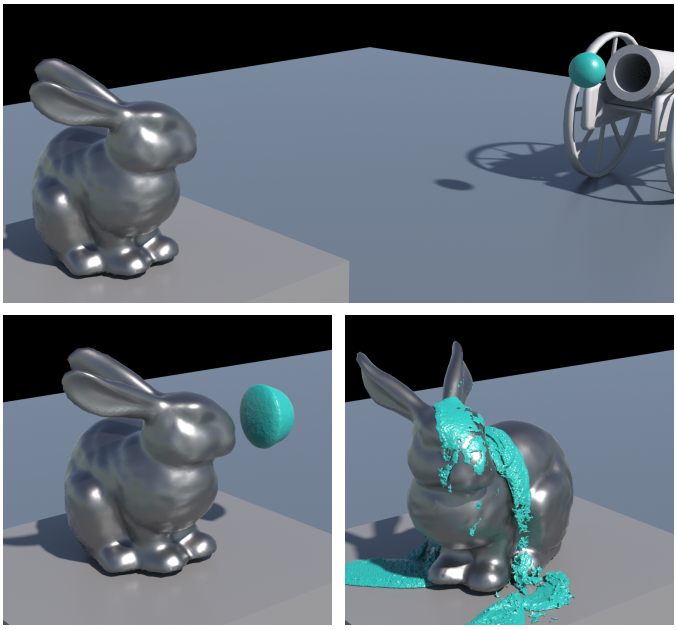


Figure 7: The Stanford Bunny being shot with a viscous cannon ball. The cannon ball consists of 800k fluid particles and deforms due to the drag force. The cannon model is courtesy of dberube4 on www.blendswap.com.

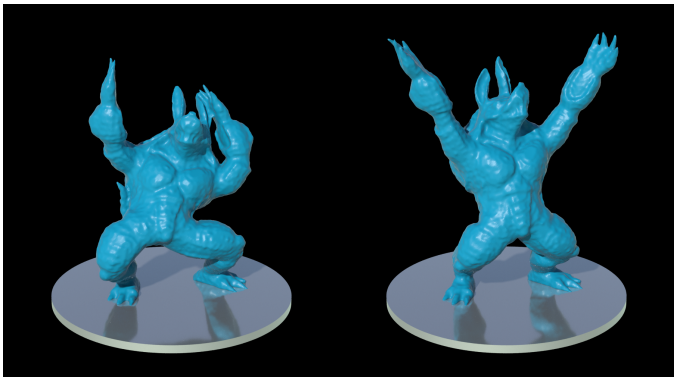


Figure 8: Armadillo dancing in the wind. The Armadillo is simulated as deformable object and consists of 230k SPH particles. The air velocity points upward.

previous aerodynamic simulation done with [44]. This allows to take the air flow at the windshield apron and inside the motor compartment into account, which also influences the liquid flow. A comparison between the simulation with imported air velocities and a constant air velocity is shown in the accompanying video. It demonstrates that the air has a strong influence on the overall liquid behavior.

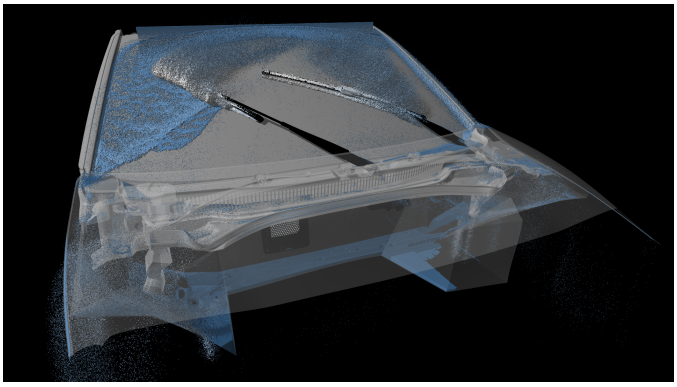


Figure 9: Wiper simulation. The air velocities used for the drag force have been precomputed with an aerodynamic simulation. This allows to consider complex air flows.

6.5. Basic air-rigid interaction

As a simple example for the air-rigid interaction forces, we let a pendulum fall due to a gravity of $9.81\,\mathrm{m\,s^{-2}}$. The pendulum has a length of 12 cm, a density of $50\,\mathrm{kg\,m^{-3}}$ and is falling from a height of 8 m. The spacing of the boundary particles is 1 mm. We use a frame rate of 500 frames/s and do one simulation step per frame resulting in a time step of 2 ms. As seen in the accompanying video and Fig. 10, our drag force results in the initially sideways lying pendulum swinging back and forth.

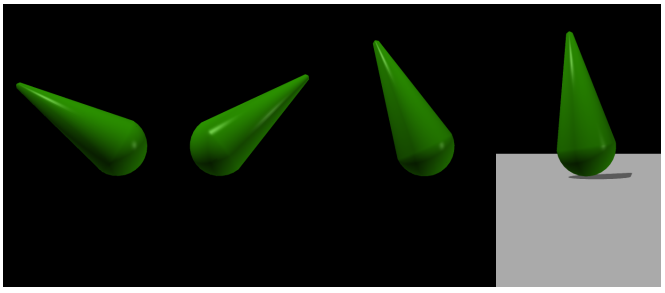


Figure 10: Falling pendulum. Due to the drag force acting on the sides of the pendulum it starts swinging back and forth.

6.6. Two-way coupled rigid object

To demonstrate the possibility to simulate a two-way coupled rigid object that is influenced by the surrounding wind as well as by a liquid, we simulated a sailing ship. In this scene, the drag force acts on the fluid as well as on the ship. The ship sails on a wavy sea which we generated with a moving plane on the right. The particle size is 2 cm and the wind blows with $10\,\mathrm{m\,s^{-1}}$ to the right. The ship as well as the water have a density of $1000\,\mathrm{kg\,m^{-3}}$. All other settings are default as described at the start of this section. For comparison, we also simulated the scene with no drag force. In Fig. 11 the wind blows the ship to the right. Additionally to moving the sailing ship, the drag force also influences the waves. Since it acts on the front of the waves, the wave breaks later as seen in Fig. 12.

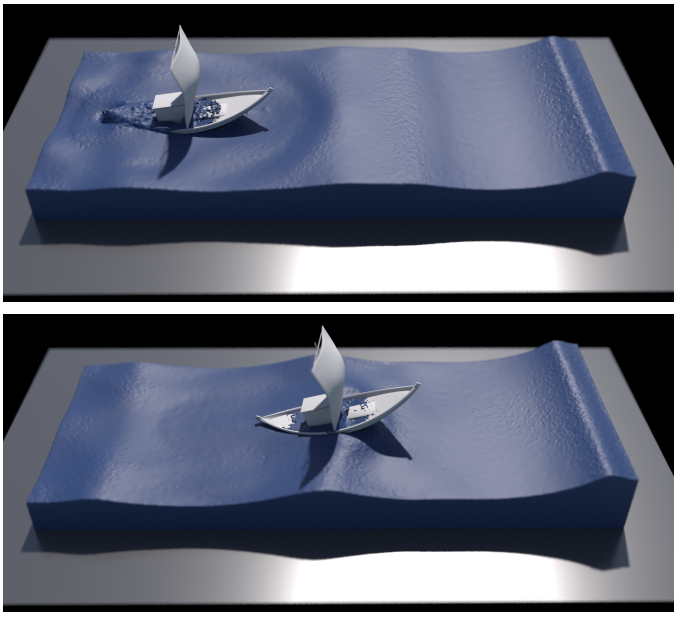


Figure 11: In the scene with no drag force (top) the sailing ship stays on the left. When the wind is blowing into its sail the ship slowly moves to the right (bottom).

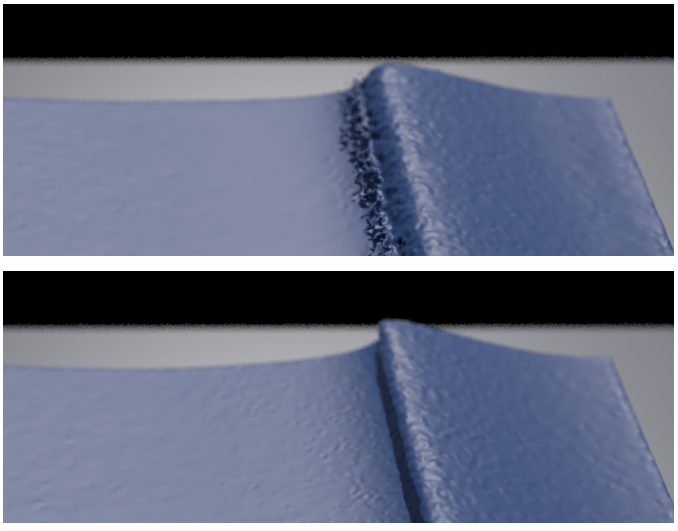


Figure 12: The drag force blowing against the waves results in them breaking later (bottom) compared to the scene with no drag forces (top).

6.7. Air-rigid interaction: comparison to SPH simulation

We want to compare the behavior of rigids influenced by our drag force to a simulation where the air is simulated with SPH. For this, we followed an experiment detailed in [45]. A simple channel is set up with a height and depth of 0.3 m and a length of 1 m. The air is flowing from left to right with an inflow velocity

of $1\,\mathrm{m\,s^{-1}}$. A small block is located 0.32 m to the right of the inflow with a height of 0.04 m and a length of 0.01 m. Three boxes with a density of $10\,\mathrm{kg\,m^{-3}}$ are placed on the left side. The gravity is set to $0\,\mathrm{m\,s^{-2}}$. Figure 13 shows the scene. We simulated this setup with our drag force approach as well as with an IISPH simulation used for the air. We set the frame rate to 500 frames per second and simulated 650 frames resulting in a simulated time of 1.3 s. The time step is 0.4 ms. For the IISPH simulation, we additionally simulated 500 frames before releasing the three cuboids in order to let the channel fully fill up with particles. The particle spacing is set to 2 mm. For our drag force, we used a precomputed air velocity field which we calculated using IISPH. Although the behavior of the cuboids differs when using the drag force instead of the full simulation of the air phase, it looks plausible. In the accompanying video you can see both movement sequences side by side. The drag force simulation took only 40 s in comparison to 13.5 min for the SPH simulation.

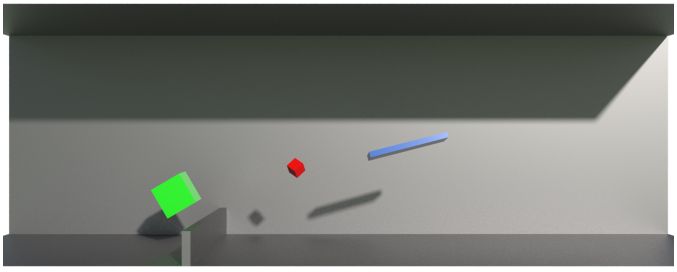


Figure 13: Our proposed drag force acts on the colored boxes and moves them from left to right.

7. Conclusion

We presented a drag force formulation and discretization to model the effects of air onto a particle-based simulation. We combined the drag force with SPH-based free-surface liquids and a rigid body simulation. Our proposed force takes the exposed surface area of each particle into account. For liquid particles, it additionally approximates their deformation to improve the parameter estimation.

We have shown that our force allows to capture effects acting from a second phase onto a liquid. We compared a simulation with our drag force to a multiphase simulation and showed matching results. With our approach the surrounding air does not need to be sampled with particles and is computationally cheaper than a multiphase simulation. Since we add our drag force as an explicit force, it can be combined with different types of SPH solvers. We combine our force with different SPH pressure solvers and also with solvers for different materials. Multiple scenes have been simulated to show the effects of our drag force on liquids. We also demonstrated that our proposed drag force results in plausible effects when applied to rigid objects. It allows to model interesting effects like for example a sailing ship. Summarized, our proposed drag force is a simple and computationally cheap method to improve the plausibility of particle-based simulations.

7.1. Limitations and future work

Depending on the resolution of the fluid, i.e., the size of a single particle, the terminal velocity of the particle varies. This can be problematic when the particle spacing is larger than a fluid droplet would be in reality. Furthermore, the radius used for the neighborhood search influences which neighboring particles contribute to the occlusion of a particle. Since we set the neighborhood search radius depending on the particle spacing, this also varies with simulation resolution. This problem could be fixed by decoupling the neighborhood search radius for the occlusion computation from the SPH neighborhood. However, this may require more memory and computational effort.

With our approach, we cannot simulate entrapped air inside a liquid since we do not represent the air with particles. It would probably be feasible to combine it with approaches that model these effects. Furthermore, since we do not simulate the dynamics of the air phase, it is hard to produce effects that come from this fluid behavior of the air. This includes simulating falling leaves to achieve results similar to the ones shown in [46]. An interesting future topic could be to try simulating the air phase with a different method than SPH, e.g., with an Eulerian approach. The grid cells could probably be chosen coarser than the SPH resolution which would result in a computationally cheaper simulation compared to a full SPH multiphase simulation. Both simulations could then be coupled with our proposed drag force.

Acknowledgments

We would like to thank the reviewers for their helpful comments. This project is partially supported by the German Research Foundation (DFG) under contract number TE 632-1/2. The Armadillo and Bunny models are courtesy of the Stanford Computer Graphics Laboratory. We thank BBS Motorsport GmbH for providing the rim model for the tire testing scene. Further, we thank FIFTY2 Technology's team members for their support. We also want to give special thanks to Michael Ehlen from qpunkt GmbH for providing feedback and ideas.

References

[1] Ihmsen, M, Orthmann, J, Solenthaler, B, Kolb, A, Teschner, M. SPH Fluids in Computer Graphics. In: Lefebvre, S, Spagnuolo, M, editors. Eurographics 2014 - State of the Art Reports. The Eurographics Association; 2014,doi:10.2312/egst.20141034.

[2] Coumans, E. Bullet physics library. www.bulletphysics.org; 2017.

[3] Akinci, N, Ihmsen, M, Akinci, G, Solenthaler, B, Teschner, M. Versatile Rigid-fluid Coupling for Incompressible SPH. ACM Trans Graph 2012;31(4):62:1–62:8. doi:10.1145/2185520.2185558.

[4] Schechter, H, Bridson, R. Ghost SPH for Animating Water. ACM Trans Graph 2012;31(4):61:1–61:8. doi:10.1145/2185520.2185557.

[5] Müller, M, Solenthaler, B, Keiser, R, Gross, M. Particle-based Fluid-fluid Interaction. In: Proceedings of the 2005 ACM SIGGRAPH/Eurographics Symposium on Computer Animation. SCA '05; New York, NY, USA: ACM. ISBN 1-59593-198-8; 2005, p. 237–244. doi:10.1145/1073368.1073402.

[6] Ihmsen, M, Bader, J, Akinci, G, Teschner, M. Animation of Air Bubbles with SPH. In: GRAPP 2011 - Proceedings of the International Conference on Computer Graphics Theory and Applications, Vilamoura, Algarve, Portugal, March 5-7, 2011. 2011, p. 225–234.

[7] Ihmsen, M, Akinci, N, Akinci, G, Teschner, M. Unified Spray, Foam and Air Bubbles for Particle-based Fluids. Vis Comput 2012;28(6-8):669–677. doi:10.1007/s00371-012-0697-9.

[8] Macklin, M, Müller, M, Chentanez, N, Kim, TY. Unified Particle Physics for Real-time Applications. ACM Trans Graph 2014;33(4):153:1–153:12. doi:10.1145/2601097.2601152.

[9] Gissler, C, Band, S, Peer, A, Ihmsen, M, Teschner, M. Approximate Air-Fluid Interactions for SPH. In: Jaillet, F, Zara, F, editors. Workshop on Virtual Reality Interaction and Physical Simulation. The Eurographics Association. ISBN 978-3-03868-032-1; 2017,doi:10.2312/vriphys.20171081.

[10] Müller, M, Charypar, D, Gross, M. Particle-based Fluid Simulation for Interactive Applications. In: Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation. SCA '03; Aire-la-Ville, Switzerland, Switzerland: Eurographics Association. ISBN 1-58113-659-5; 2003, p. 154–159.

[11] Monaghan, JJ. Smoothed particle hydrodynamics. Reports on Progress in Physics 2005;68(8):1703.

[12] Tartakovsky, AM, Meakin, P. A Smoothed Particle Hydrodynamics Model for Miscible Flow in Three-dimensional Fractures and the Two-dimensional Rayleigh-Taylor Instability. J Comput Phys 2005;207(2):610–624. doi:10.1016/j.jcp.2005.02.001.

[13] Solenthaler, B, Pajarola, R. Density Contrast SPH Interfaces. In: Proceedings of the 2008 ACM SIGGRAPH/Eurographics Symposium on Computer Animation. SCA '08; Aire-la-Ville, Switzerland, Switzerland: Eurographics Association. ISBN 978-3-905674-10-1; 2008, p. 211–218.

[14] Lombardi, JC, Sills, A, Rasio, FA, Shapiro, SL. Tests of Spurious Transport in Smoothed Particle Hydrodynamics. Journal of Computational Physics 1999;152(2):687–735.

[15] Cleary, PW, Pyo, SH, Prakash, M, Koo, BK. Bubbling and Frothing Liquids. ACM Trans Graph 2007;26(3). doi:10.1145/1276377.1276499.

[16] Hong, JM, Lee, HY, Yoon, JC, Kim, CH. Bubbles Alive. ACM Trans Graph 2008;27(3):48:1–48:4. doi:10.1145/1360612.1360647.

[17] Bhat, KS, Twigg, CD, Hodgins, JK, Khosla, PK, Popović, Z, Seitz, SM. Estimating Cloth Simulation Parameters from Video. In: Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation. SCA '03; Aire-la-Ville, Switzerland, Switzerland: Eurographics Association. ISBN 1-58113-659-5; 2003, p. 37–51.

[18] Wilson, K, McAdams, A, Leo, H, Simmons, M. Simulating Wind Effects on Cloth and Hair in Disney's Frozen. In: ACM SIGGRAPH 2014 Talks. SIGGRAPH '14; New York, NY, USA: ACM. ISBN 978-1-4503-2960-6; 2014, p. 48:1–48:1. doi:10.1145/2614106.2614120.

[19] Keckeisen, M, Kimmerle, S, Thomaszewski, B, Wacker, M. Modelling Effects of Wind Fields in Cloth Animations. Journal of WSCG 2004;12(1–3):205–212.

[20] Becker, M, Teschner, M. Weakly Compressible SPH for Free Surface Flows. In: Proceedings of the 2007 ACM SIGGRAPH/Eurographics Symposium on Computer Animation. SCA '07; Aire-la-Ville, Switzerland, Switzerland: Eurographics Association. ISBN 978-1-59593-624-0; 2007, p. 209–217.

[21] Solenthaler, B, Pajarola, R. Predictive-corrective Incompressible SPH. ACM Trans Graph 2009;28(3):40:1–40:6. doi:10.1145/1531326.1531346.

[22] Bender, J, Koschier, D. Divergence-Free SPH for Incompressible and Viscous Fluids. IEEE Transactions on Visualization and Computer Graphics 2017;23(3):1193–1206. doi:10.1109/TVCG.2016.2578335.

[23] Ihmsen, M, Cornelis, J, Solenthaler, B, Horvath, C, Teschner, M. Implicit Incompressible SPH. IEEE Transactions on Visualization and Computer Graphics 2014;20(3):426–435. doi:10.1109/TVCG.2013.105.

[24] Morris, JP, Fox, PJ, Zhu, Y. Modeling Low Reynolds Number Incompressible Flows Using SPH. J Comput Phys 1997;136(1):214–226. doi:10.1006/jcph.1997.5776.

[25] Takahashi, T, Dobashi, Y, Fujishiro, I, Nishita, T, Lin, MC. Implicit Formulation for SPH-based Viscous Fluids. Comput Graph Forum 2015;34(2):493–502. doi:10.1111/cgf.12578.

[26] Peer, A, Ihmsen, M, Cornelis, J, Teschner, M. An Implicit Viscosity Formulation for SPH Fluids. ACM Trans Graph 2015;34(4):114:1–114:10. doi:10.1145/2766925.

[27] Anderson Jr., JD. Fundamentals of Aerodynamics. Third ed.; McGraw-Hill; 2001. ISBN 9780072373350.

[28] O'Rourke, PJ, Amsden, AA. The Tab Method for Numerical Calculation of Spray Droplet Breakup. In: SAE Technical Paper. SAE International; 1987,doi:10.4271/872089.

[29] Nelson, AR, Gokhale, NR. Oscillation Frequencies of Freely Suspended Water Drops. Journal of Geophysical Research 1972;77(15):2724–2727. doi:10.1029/JC077i015p02724.

[30] Vollmer, M, Möllmann, KP. Oscillating droplets and incompressible liquids: slow-motion visualization of experiments with fluids. Physics Education 2012;47(6):664.

[31] Taylor, GI. The Shape and Acceleration of a Drop in a High Speed Air Stream. The Scientific Papers of Geoffrey Ingram Taylor 1963;3:457–464.

[32] Liu, AB, Mather, D, Reitz, RD. Modeling the Effects of Drop Drag and Breakup on Fuel Sprays. In: SAE Technical Paper. SAE International; 1993,doi:10.4271/930072.

[33] Morris, JP. Simulating surface tension with smoothed particle hydrodynamics. International Journal for Numerical Methods in Fluids 2000;33(3):333–353. doi:10.1002/1097-0363(20000615)33:3<333::AID-FLD11>3.0.CO;2-7.

[34] Barecasco, A, Terissa, H, Naa, CF. Simple free-surface detection in two and three-dimensional SPH solver. Selected Papers from the International Symposium on Computational Science Kanazawa University, Japan 2013;4:1–10.

[35] Yang, T, Lin, MC, Martin, RR, Chang, J, Hu, SM. Versatile Interactions at Interfaces for SPH-based Simulations. In: Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation. SCA '16; Aire-la-Ville, Switzerland, Switzerland: Eurographics Association. ISBN 978-3-905674-61-3; 2016, p. 57–66.

[36] Ihmsen, M. Particle-based Simulation of Large Bodies of Water with Bubbles, Spray and Foam. Ph.D. thesis; University of Freiburg; 2013.

[37] Akinci, N, Akinci, G, Teschner, M. Versatile Surface Tension and Adhesion for SPH Fluids. ACM Trans Graph 2013;32(6):182:1–182:8. doi:10.1145/2508363.2508395.

[38] Side Effects Software, . Houdini. www.sidefx.com; 2017.

[39] Gunn, R, Kinzer, GD. The Terminal Velocity of Fall for Water Droplets in Stagnant Air. Journal of Meteorology 1949;6(4):243–248. doi:10.1175/1520-0469(1949)006<0243:TTVOFF>2.0.CO;2.

[40] Foote, GB, Du Toit, PS. Terminal Velocity of Raindrops Aloft. Journal of Applied Meteorology 1969;8(2):249–253. doi:10.1175/1520-0450(1969)008<0249:TVORA>2.0.CO;2.

[41] Ganzenmüller, GC. An hourglass control algorithm for Lagrangian Smooth Particle Hydrodynamics. Computer Methods in Applied Mechanics and Engineering 2015;286:87–106. doi:10.1016/j.cma.2014.12.005.

[42] Becker, M, Ihmsen, M, Teschner, M. Corotated SPH for Deformable Solids. In: Proceedings of the Fifth Eurographics Conference on Natural Phenomena. NPH'09; Aire-la-Ville, Switzerland, Switzerland: Eurographics Association. ISBN 978-3-905674-11-8; 2009, p. 27–34. doi:10.2312EG/DL/conf/EG2009/nph/027-034.

[43] Solenthaler, B, Schläfli, J, Pajarola, R. A unified particle model for fluid-solid interactions. Computer Animation and Virtual Worlds 2007;18(1):69–82. doi:10.1002/cav.162.

[44] Dassault Systèmes, . XFlow. www.xflowcfd.com; 2017.

[45] Hess, S, Lin, L, Sampath, R, Prescott, S, Smith, C. Smoothed-particle Hydrodynamics based Wind Representation. Tech. Rep.; Idaho National Laboratory; 2016.

[46] Martin, T, Umetani, N, Bickel, B. OmniAD: Data-driven Omnidirectional Aerodynamics. ACM Trans Graph 2015;34(4):113:1–113:12. doi:10.1145/2766919.