GRAPP (2011)
Algarve (Portugal)

Animation of air bubbles with SPH

Markus Ihmsen Julian Bader Gizem Akinci Matthias Teschner

University of Freiburg

Abstract

We present a physically-based multiphase model for simulating water and air bubbles with Smoothed Particle Hydrodynamics (SPH). Since the high density ratio of air and water is problematic for existing SPH solvers, we compute the density and pressure forces of both phases separately. The two-way coupling is computed according to the velocity field. The proposed model is capable of simulating the complex bubble flow, e. g. path instability, deformation and merging of bubbles and volume-dependent buoyancy. Furthermore, we present a velocity-based heuristic for generating bubbles in regions where air is likely trapped. Thereby, bubbles are generated on the fly, without explicitly simulating the air phase surrounding the liquid. Instead of deleting the bubbles when they reach the surface, we employ a simple foam model. By incorporating our model into the predictive-corrective SPH method, large time steps can be used. Thus, we can simulate scenarios of high resolution where the size of the bubbles is small in comparison to the liquid volume.

Categories and Subject Descriptors (according to ACM CCS): Computer Graphics [I.3.7]: Three-Dimensional Graphics and Realism: —Animation

1. INTRODUCTION

Air bubbles are a natural phenomenon that occurs in everyday life. Whenever a liquid is poured into a glass, bubbles are created by trapped air. Accordingly, the visual realism of fluid animations is significantly enhanced by modeling the creation and flow of bubbles. They might be even used to synthesize the sound generated by the fluid [MYH*10]. However, the realistic simulation of air bubbles poses some challenges.

In Computer Graphics, two major approaches are employed for animating fluids, the Eulerian grid-based method and the Lagrangian particle method. While the Eulerian approach is particularly suited to simulate large volumes of water, its performance is limited by the grid spacing. In order to simulate small scale features, a very fine resolution is required which restricts the time step and increases the computing time. In contrast, particle methods like Smoothed Particle Hydrodynamics (SPH) are suitable for capturing small scale effects like the flow of tiny air bubbles.

In order to capture the creation of bubbles, the computation of the air phase is required. However, this invokes a significant computational overhead. For grid-based methods, a second grid with fine resolution is required which computes the air flow. In particle methods, the air phase has to be represented explicitly.

In reality, air and water are interacting in a two-way manner. Unlike water droplets, air bubbles are under strong velocity diffusion because they are coupled to the surrounding fluid by drag and lift forces. According to the very large density ratio of air to water (≈ 1000), water exerts a high pressure on air bubbles which makes them merge rapidly. As the bubbles grow, they rise faster due to the rapid increase in buoyancy. In turn, those large and fast rising bubbles significantly influence the liquid flow. As stated in [SP08], modeling high density ratios with SPH is problematic and may lead to numerical instabilities due to large forces. Thus, the simulation of air bubbles is not possible by directly employing the standard SPH method [MSKG05, SP08].

Contribution. We present a new SPH model for simulating air bubbles and foam. In order to handle the high density ratio of air and water, we treat the two phases separately. We account for the interaction of the two phases by employing a drag force. As we show, the proposed drag force is sufficient to capture the two-way interaction realistically, while the numerical stability is not affected.

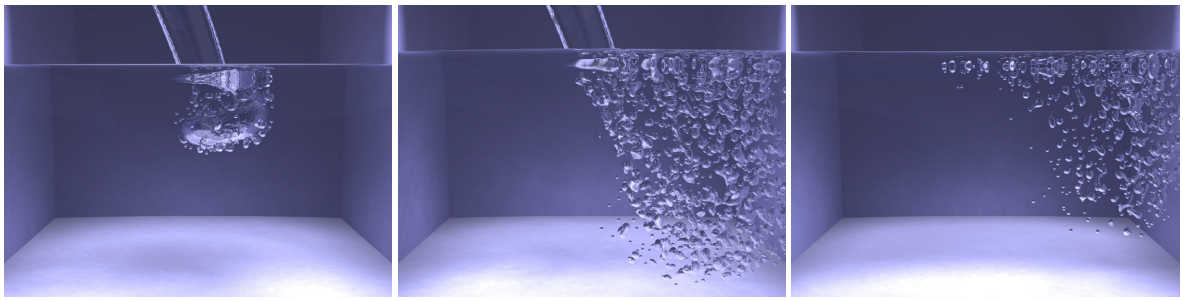

Figure 1: Trapped air. Air bubbles are generated on the fly in regions of high velocity differences. The bubble flow is significantly influenced by the liquid. Bubbles are merging and deforming. This simulation contains 1.4 million liquid particles and up to 6000 air particles.

We model the buoyancy of the air bubbles by a saturated function that accounts for the volume. Thereby, large bubbles rise faster than small bubbles. Furthermore, a cohesion force is employed that minimizes the surface and makes rising bubbles merge.

In order to simulate trapped air without explicitly modeling the air surrounding the fluid, we generate them on the fly in surface regions with high velocity differences. When air particles have reached the surface, they are treated as foam and finally deleted after a user defined time.

The presented bubble model can be easily incorporated into any existing SPH solver with negligible computational overhead. We suggest to use the predictive-corrective SPH (PCISPH) algorithm [SP09] since it can handle large time steps and is efficient to compute. Therefore, high resolution scenes can be simulated where the size of the bubbles is small in comparison to the liquid volume. A first example is illustrated in Fig. 1.

2. RELATED WORK

In this work, we focus on an SPH based fluid simulation for animating air bubbles and their interaction with water. In Computer Graphics, SPH is applied to model different materials like gas [SF95], deformable objects [DC96], [SSP07], hair [HMT01] and liquids [MCG03]. However, the interaction of air and water is hardly covered since the high density ratio poses severe problems to the SPH algorithm [Mon02].

In [MSKG05], the standard SPH model for single-phase fluid simulations [MCG03] is extended to handle multiple fluids. This approach can handle density ratios of up to 10. In order to simulate air bubbles, an artificial buoyancy force is applied. However, as stated in [SP08], this approach suffers from falsified density estimations at the interface which induce wrong pressure values. This in turn, limits the time step and might result in numerical instabilities for fast rising air particles due to large pressure forces. [SP08] overcomes this problem by ignoring the mass in the computation of the particle density. Thereby, sharp density changes at the fluid interface can be reproduced. Although this method can handle density ratios of up to 100, the flow of small, light volumes like air bubbles can not be realistically handled. According to Solenthaler et al., the buoyant volumes can not break up the crystallized particle configuration formed by the pressure forces. In order to circumvent these problems, we ignore particle neighbors of other phases when computing the density. Thus, we treat each phase separately. The interaction of both phases is modeled via a drag force.

A similar idea is presented in [CPPK07] for simulating dynamic gas bubbles generated from gas dissolution. In this approach, each phase is computed separately, where the air bubbles are modeled by discrete entities with fixed shape and the liquid is computed with SPH. The bubbles are coupled to the liquid via a drag force while the influence of the bubbles onto the liquid is neglected. In contrast, our bubble model is based on SPH and the governing forces are computed differently. Furthermore, we couple the liquid and air phase in a two-way manner using a different formulation of the drag force.

Capturing the fine scale flow of bubbles with Eulerian methods, requires very fine grid resolutions. As stated in [HLYK08], for numerical reasons, each bubble should at least occupy 3 nodes in each dimension. Although, the computational overhead can be minimized by adaptively refining the grid using an octree [LGF04], the required grid spacing significantly restricts the time step. Consequently, pure grid-based methods like the regional level set method [ZYP06] are only suited to handle relatively large bubbles in comparison to the fluid volume. Alternatively, hybrid methods have been proposed [HK03, GH04], in which the bubbles are simulated by passive air particles that are advected according to the underlying grid. Similar to [KVG02], the particles are modeled as spheres which do no not change their shape. By coupling the bubble particles to a low resolution grid, millions of air particles can be simulated efficiently as shown in [KSK10]. However, in these models the interaction of particles is often neglected. Therefore, the size and shape of air bubbles is not varying over time. In contrast, in the proposed model, the air bubbles can consist of many particles. The

employed cohesion force minimizes the bubble surface and makes bubbles merge. Since we reconstruct the bubble surface from the particle positions as described in [SP08], the bubble shape is deformable.

A hybrid solver is also proposed in [HLYK08], where bubbles are simulated with SPH and coupled to a grid-based fluid solver. The two phases are coupled via the velocity field. However, due to the insufficient resolution of the underlying grid, the path instability of air bubbles can not be simulated by this coupling. This is achieved by adapting the vorticity confinement method [FSJ01, SRF05] and artificial velocity disturbance based on random numbers. In contrast, we model the liquid phase with SPH. Thus, fine scale turbulences in the liquid can be simulated. The proposed two-way coupling is velocity based. Thereby, the bubble flow is significantly influenced by the velocity field of the liquid. Consequently, the path instability of bubbles can be realistically simulated without adding artificial disturbance.

In [TSS*07], a two-dimensional shallow water model is coupled to a particle-based bubble simulation. In order to capture three-dimensional effects, the shallow water model makes a number of simplifying assumptions. In particular, the fluid flow is only modeled around bubbles and only if bubbles are in the fluid. Thereby, the model is very efficient to compute, but some effects can not be modeled, e. g. inertia effects of the fluid. In contrast, the primary focus of the proposed model is not interactivity, but the realism of the animation. However, we integrate our model into the PCISPH algorithm presented in [SP09]. Thereby, large time steps can be used which allows us to simulate millions of particles in reasonable time.

The remainder of the paper is organized as follows. First, we describe the basics of SPH and discuss its problems in handling high-density ratios. In Sec. 4, the proposed model for simulating air bubbles with SPH is explained in detail. Finally, we discuss implementation issues and demonstrate the capability of the presented method.

3. SPH

In this section, we briefly explain the basics of the SPH method for single-phase and multiphase fluids.

3.1. Single-phase SPH

In SPH, the fluid is discretized into a finite set of particles i with position $\mathbf{x}_i$ and velocity $\mathbf{v}_i$. Generally, a particle quantity A_i is approximated by a smooth function which interpolates A_i using a finite set of sampling points j located within a distance h. This set of sampling points is called particle neighborhood. The smooth function is defined as

$$A_i = \sum_j \frac{m_j}{\rho_j} A_j W(\mathbf{x}_i - \mathbf{x}_j, h), \tag{1}$$

where m_j is the mass of j, ρ_j its density and $W(\mathbf{x}_i - \mathbf{x}_j, h)$ is a kernel function with support radius h.

The particle positions and velocities are integrated according to internal and external forces. Internal forces are viscosity, surface tension and pressure forces, where the pressure force mainly governs the macroscopic flow.

In Computer Graphics, two different algorithms are generally used for computing the pressure of SPH fluids, namely the state equation based algorithm (SESPH) and the predictive-corrective SPH algorithm (PCISPH), see Alg. 1 and Sec. 4.4, respectively.

In SESPH, the pressure is related with the density. Commonly, for compressible fluids, the ideal gas equation (2) [MCG03] and for weakly-compressible fluids, the Tait equation (3) [Mon92, BT07] are used

$$p_i = c_s^2(\rho_i - \rho_0) \tag{2}$$

$$p_i = \frac{c_s^2 \rho_0}{7}\left(\left(\frac{\rho_i}{\rho_0}\right)^7 - 1\right) \tag{3}$$

where c_s denotes the speed of sound and ρ_0 the rest density of the fluid. The density can be computed with (1) as

$$\rho_i = \sum_j m_j W(\mathbf{x}_{ij}, h) \tag{4}$$

where $\mathbf{x}_{ij} = \mathbf{x}_i - \mathbf{x}_j$. The pressure force is directly derived from the Navier-Stokes equations as

$$\mathbf{F}_i^{pressure} = -m_i \sum_j m_j \left(\frac{p_i}{\rho_i^2} + \frac{p_j}{\rho_j^2}\right) \nabla W(\mathbf{x}_{ij}, h). \tag{5}$$

According to [Mon05], the viscosity force for particle pairs with $\mathbf{v}_{ij} \cdot \mathbf{x}_{ij} < 0$ is computed as

$$\mathbf{F}_i^{viscosity} = m_i \sum_j m_j \nu \left(\frac{\mathbf{v}_{ij} \cdot \mathbf{x}_{ij}}{\|\mathbf{x}_{ij}\|^2 + \varepsilon h^2}\right) \nabla_i W(\mathbf{x}_{ij}, h) \tag{6}$$

with the viscosity term $\nu = \mu \frac{2hc_s}{\rho_i + \rho_j}$, where μ is the viscosity constant.

The surface tension force can be elegantly computed as proposed in [BT07] as

$$\mathbf{F}_i^{surface} = -k_s \sum_j m_j \mathbf{x}_{ij} W(\mathbf{x}_{ij}, h) \tag{7}$$

where k_s is a user-defined surface tension coefficient.

Finally, the acceleration $\mathbf{a}_i$ of a particle i is computed as

$$\mathbf{a}_i = m_i^{-1}\left(\mathbf{F}_i^{pressure} + \mathbf{F}_i^{viscosity} + \mathbf{F}_i^{surface}\right) + \mathbf{g} \tag{8}$$

where $\mathbf{g}$ denotes the gravity. Similar to recent work in the field of SPH, we use the above force equations in conjunction with the cubic spline kernel of [Mon92]. Alternative force equations and kernels can be found in [MCG03].

Algorithm 1: SESPH

foreach *particle i* **do**
  compute density (4) ;
  compute pressure (3) or (2);
foreach *particle i* **do**
  compute acceleration (8);
  integrate position, velocity;

3.2. Multiphase SPH

The SESPH algorithm can be easily extended in order to handle multiple fluids with different rest densities [MSKG05]. However, as shown in [SP08], miscible fluids with a density ratio larger than 10 can not be realistically simulated if the standard SPH density summation (4) is used. The reason is that in SPH, the macroscopic flow is mainly governed by the density computation. Over- or underestimating the density leads to erroneous pressure values, which might result in unnatural acceleration caused by erroneously introduced pressure ratios. In [SP08], a different density model is proposed which treats all particle neighbors as if they belong to the same phase, i. e. have the same mass and rest density. This model computes the density as $\rho_i = m_i \sum_j W(\mathbf{x}_{ij}, h)$. Thereby, the densities at the interface are computed correctly.

Although this method can represent sharp density changes at the interface, it suffers from severe limitations when dealing with large density ratios. The buoyancy of small, light volumes is significantly damped. As stated in [SP08], this is due to the pressure force which compels the particles to arrange in a stable lattice structure. This structure can not be broken by small volumes like e. g. air bubbles.

In the following section, we present a new SPH model for simulating the flow of air bubbles and the two-way coupled interaction of air and water. The presented model avoids the above mentioned problems.

4. BUBBLES

In order to simulate bubbles with SPH, we propose to simulate the air and liquid phase separately, i. e. only particle neighbors of the same phase contribute to the density, pressure and general force computations. Accordingly, problems like e. g. high pressure ratios or buoyancy dampening do not occur. We account for the interaction of both phases by employing a new velocity based coupling.

In the following, we present force equations for controlling the bubble flow and the interaction of both phases. Subsequently, we propose an efficient method for generating air bubbles caused by trapped air and discuss a simple model for transforming air bubbles into foam. Finally, we show how this model can be integrated into the PCISPH algorithm.

4.1. Bubble Physics

In order to simulate bubbles realistically, we have to capture the prominent effects of their complex behavior. In general, bubbles have a sphere like shape according to cohesion forces (surface and interface tension). However, since bubbles are heavily influenced by the velocity field of the surrounding fluid, the bubble shape is deforming. Furthermore, bubbles are different in size, where larger bubbles rise faster and attract smaller bubbles.

For modeling these effects, we employ two forces that are computed for the air phase only, namely the buoyancy force $\mathbf{F}^{buyoancy}$ and the cohesion force $\mathbf{F}^{cohesion}$. The coupling of the two phases is realized by the drag force $\mathbf{F}^{drag}$.

Buoyancy. The buoyancy force accelerates the air bubble in direction of the liquid surface. In order to make larger bubbles rise faster than smaller ones, the buoyancy force should account for the volume of the air bubble V_{bub}. This leads to

$$\mathbf{F}_i^{buoyancy} = -k_b \cdot V_{bub} \cdot \mathbf{g} \tag{9}$$

where k_b controls the buoyancy. However, computing V_{bub} is not straight forward without knowing which particle belongs to which bubble. The determination thereof invokes a significant computational overhead which can be avoided by using a heuristic formulation.

Replacing V_{bub} in (9) with $V_i = \frac{m_i}{\rho_i} = \frac{1}{\sum_j W(\mathbf{x}_{ij},h)}$ is not suitable since V_i gets smaller as the number of neighbors grows, i. e. an isolated particle would rise faster than a large bubble. On the other hand, correlating the buoyancy with the density ρ_i is also not optimal since thereby, a compressed bubble (smaller volume) would rise faster than an expanded one.

We therefore propose to relate the magnitude of the buoyancy force to the number of particle neighbors n with

$$\mathbf{F}_i^{buoyancy} = -m_i k_b \cdot \left(k_{max} - (k_{max} - 1) \cdot e^{-0.1 n_i} \right) \cdot \mathbf{g} \tag{10}$$

where k_b controls the minimum buoyancy and k_{max} the maximum buoyancy. The mass is added to the force formulation, in order to make the resulting acceleration independent of the simulation resolution. Thus, the resulting acceleration caused by the buoyancy force can be perfectly controlled since $\|k_b \mathbf{g}\| \leq \left\| \mathbf{a}_i^{buoyancy} \right\| \leq \|k_{max} \cdot k_b \cdot \mathbf{g}\|$. This is also illustrated in Fig. 2.

Cohesion. The coalescence of air bubbles is an important effect. Smaller air bubbles are attracted by surrounding bubbles due to interface and surface tension forces. We model this behavior by employing an artificial cohesion force

$$\mathbf{F}_i^{cohesion} = -k_c m_i \sum_j \rho_j \mathbf{x}_{ij} \tag{11}$$

where k_c controls the strength of the cohesion force. According to (11), air particles are attracted by neighboring

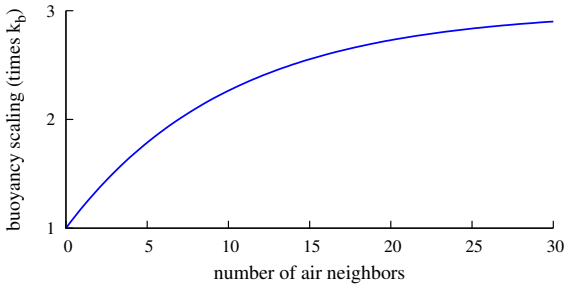

Figure 2: Scaling of the proposed volume-dependent buoyancy for $k_{max} = 3$. In order to account for the unknown total bubble volume, the buoyancy of an air particle is non-linearly related to its number of air neighbors.

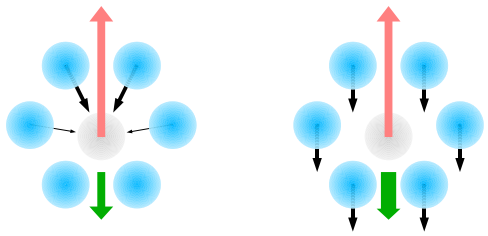

Figure 3: Comparison of the proposed drag force (left) and the drag force used in [CPPK07] (right). The influence of the liquid particles (blue) onto the air particle (grey) is shown. The velocity of the air particle is indicated by the red arrow, the liquid is at rest. Partial forces are illustrated by black arrows, while the resultant force is given by the green arrow. The thickness denotes the magnitude.

air particles with higher density. Thereby, spatially close air bubbles do merge. Note that the pressure force (5) counteracts the attraction, when the density ρ_i becomes too high. As a result, the surface of the bubble is minimized while the forces converge to an equilibrium.

Two-way coupling. In [CPPK07] and [HLYK08], the air phase is coupled to the liquid phase via an empirical drag force. The effect of the bubble momentum on the liquid phase is neglected. With respect to the generally high Reynolds number for air bubbles, the drag force $\mathbf{F}_i^{drag}$ acting on an air particle i is computed as

$$\mathbf{F}_i^{drag} = -k_d A_{bub} \sum_{liq} (\mathbf{v}_i - \mathbf{v}_{liq}) \left\| \mathbf{v}_{air} - \mathbf{v}_{liq} \right\| \tag{12}$$

where k_d is a constant drag coefficient, liq the liquid neighbors of i and A_{bub} is the surface area of the bubble. Note that in [CPPK07], air bubbles are simulated as discrete spheres. Thus, A_{bub} is easy to compute. For the presented model, it is hard to compute A_{bub} since bubbles may have arbitrary shapes and can consist of many particles.

In (12), the forces exerted by neighboring liquid particles are related to the velocity difference, but the flow direction and the distance of the particles are neglected. Thus, a liquid neighbor j, that is very close $\left\| \mathbf{x}_{ij} \leq \frac{h}{2} \right\|$, influences the velocity of the air particle by the same amount as a particle with distance h. Furthermore, the partial drag force $\mathbf{F}_{ij}^{drag}$ is non-zero as long as $\left\| \mathbf{v}_{ij} \right\| > 0$, whether the particles move towards each other or not.

In contrast, we account for the distance and flow direction (see Fig. 3). Therefore, we propose a drag force that is motivated by the viscosity force (6) and couples both phases in a two-way manner. Consequently, the movement of the bubbles influences the velocity field of the fluid and vice versa. The drag force is defined as

$$\mathbf{F}_i^{drag} = m_i \sum_j m_j \frac{k_d h c_s}{\rho_i + \rho_j} \Pi_{ij} \nabla_i W(\mathbf{x}_{ij}, h) \tag{13}$$

where k_d denotes the drag constant. Π_{ij} is zero if either i and j belong to the same phase or if $\mathbf{v}_{ij} \cdot \mathbf{x}_{ij} \leq 0$, otherwise $\Pi_{ij} = \left(\frac{\mathbf{v}_{ij} \cdot \mathbf{x}_{ij}}{\|\mathbf{x}_{ij} + \varepsilon h^2\|} \right)$.

Acceleration. The acceleration of an air particle is finally computed as

$$\mathbf{a}_{air} = m_{air}^{-1} \left(\mathbf{F}_{air}^{pressure} + \mathbf{F}_{air}^{cohesion} + \mathbf{F}_{air}^{buoyancy} + \mathbf{F}_{air}^{drag} \right) + \mathbf{g}, \tag{14}$$

while for a liquid particle liq, the acceleration computes to

$$\mathbf{a}_{liq} = m_{liq}^{-1} \left(\mathbf{F}_{liq}^{pressure} + \mathbf{F}_{liq}^{viscosity} + \mathbf{F}_{liq}^{surface} + \mathbf{F}_{liq}^{drag} \right) + \mathbf{g}. \tag{15}$$

Note that the proposed bubble model can be easily integrated into any existing SPH solver. A discussion is provided in Sec. 4.4, but first we suggest an heuristic for generating and deleting air particles on the fly. Thereby, trapped air can be simulated efficiently.

4.2. Trapped Air

Air bubbles are generated when air is trapped inside the liquid. One possibility to capture this effect is to simulate the air phase surrounding the liquid explicitly. In order to avoid the implied computational overhead, we use a heuristic formulation in order to detect regions where air is trapped.

The proposed heuristic is based on the following observations:

- Air molecules are pulled inside the liquid by inflows
- The amount of trapped air grows with the velocity of the inflow.

Accordingly, we generate air particles in surface regions of high velocity differences. Therefore, for each liquid particle on the surface, the magnitude of the velocity difference

$\mathbf{v}_i^{diff} = \sum_j \frac{m_j}{\rho_j}(\mathbf{v}_i - \mathbf{v}_j)W(\mathbf{x}_{ij}, h)$ is compared with a user defined threshold v_t. If $\left\|\mathbf{v}_i^{diff}\right\| > v_t$, air is likely trapped at the position $\mathbf{x}_i$.

In order to relate the velocity of the inflow with the volume of the generated air, two further conditions must be met. First, the magnitude of the velocity must be greater than a predefined threshold $\|\mathbf{v}_i\| > v_{min}$. Second, the number of air particle neighbors n_{air} of particle i should not be greater than $\left\|\mathbf{v}^{diff}\right\| / v_t$. Consequently, the number of generated particles grows with the velocity of the inflow.

In summary, a liquid particle i generates an air particle if $\left\|\mathbf{v}_i^{diff}\right\| > v_t$, $\|\mathbf{v}_i\| > v_{min}$ and $n_{air} < \left\|\mathbf{v}^{diff}\right\| / v_t$. The position and velocity of the air particle are chosen to be the same as for the liquid particle i.

Discussion. Our model computes the density and pressure of the air and liquid phase separately. Therefore, an air particle can be generated at the position of a liquid particle without introducing high pressure forces causing unnatural acceleration. This is in contrast to the multiphase methods presented in [MSKG05, SP08] where high pressures are introduced when the distance of air and liquid particles is too small. Thus, in these models, determining an appropriate position for a generated air particle without causing numerical instabilities is not straightforward.

4.3. Foam

When air bubbles reach the liquid surface they do not rise anymore, but float on the surface until they burst. In reality, an air (foam) bubble disperses according to film rupture in a two step process that can create smaller bubbles. Note that the physics that govern this complex behavior is not fully discovered yet [BdRCS10]. However, for the purpose of animation, we use a simplified model to simulate the floating and bursting of foam bubbles.

First of all, we differentiate between particles that are inside the liquid (rising bubbles) and particles that are on the surface (foam). The liquid surface can be determined either via a smoothed color field [MCG03, KAG*05] or by comparing the number of particle neighbors with a given threshold. We treat an air particle i as on the surface (foam) either if its density ρ_i gets below a threshold t_ρ or there is no liquid neighbor j with $(\mathbf{x}_j - \mathbf{x}_i) \cdot \mathbf{g} > 0$.

The buoyancy force of foam particles is computed as

$$\mathbf{F}_i^{buoyancy} = -m_i \mathbf{g} \tag{16}$$

This force cancels out the acceleration due to gravity $\mathbf{g}$. Like rising bubble particles, foam particles are coupled to the liquid by the drag force (13). Thereby, the resulting velocity of foam particles is governed by the surrounding liquid. Thus, the foam floats on the liquid surface. Due to the cohesion force, foam particles also cluster on the surface, trying to minimize the surface. Consequently, foam bubbles are different in size and shape (see Fig. 4).

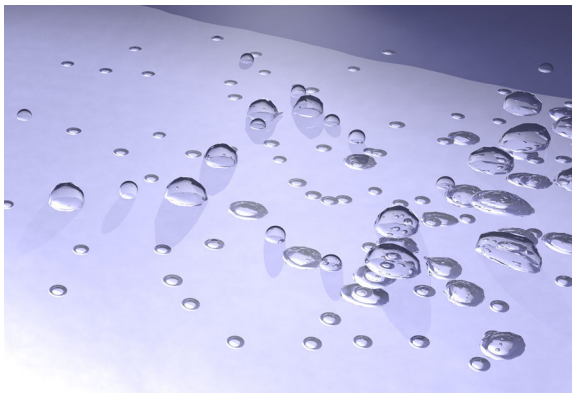

Figure 4: Foam bubbles. Air particles are not directly deleted when they reach the surface, but treated as foam. They float on the surface, according to the buoyancy and drag force. Due to the cohesion force, foam bubbles are different in size and shape.

Deletion of foam particles. When an air particle reaches the surface it is given a *floating time* t_f, i. e. time until the particle is deleted. In order to improve the realism, we vary t_f for each particle randomly using a uniform distribution. However, if two foam particles merge, i. e. are neighbors, the minimum of their floating times is assigned to both particles. Thereby, foam bubbles consisting of more than one particle disperse at once.

4.4. Algorithm

In order to simulate water realistically, the compressibility should be set very low. In SESPH, the compressibility is controlled by the speed of sound (see (2), (3)). Setting c_s higher reduces the compressibility, but also restricts the time step. In [SP09], an alternative SPH algorithm for incompressible fluids is suggested, called PCISPH. In this approach, the compression error is predicted and corrected iteratively. Thereby, significantly larger time steps can be used, while the computational overhead is small compared to SESPH. Accordingly, high-resolution fluids can be simulated within reasonable time.

The proposed bubble model can be directly incorporated into the PCISPH algorithm (see Alg. 2) with negligible computational overhead. Furthermore, the time step is not affected by the high density ratio of water and air. This is due to the velocity based coupling which avoids the computation of the pressure forces at the water-air interface.

Explaining the prediction and correction loop of the PCISPH algorithm is beyond the scope of this paper. Detailed explanations can be found in [SP09, IAGT10].

Algorithm 2: PCISPH with bubbles

```
foreach air particle do
    compute forces (11), (10), (13), (16);
foreach liquid particle do
    compute forces (6), (7), (13) ;
k = 0 ;
while (max(ρ*_err) > η or k < 3) do
    forall the particles do
        predict velocity ;
        predict position ;
    forall the particles do
        update distances to neighbors;
        predict density variation;
        update pressure ;
    forall the particles do
        compute pressure force;
    k+ = 1;
forall the particles do
    compute acceleration (14) or (15);
forall the particles i do
    integrate velocity and position;
    if i is liquid then
        test for air generation
    else
        test for deletion
```

5. RESULTS

In this section, we briefly cover implementation aspects and parameter settings. Then, we demonstrate the capability of the proposed model to animate the air bubbles and their interaction with water.

Implementation. In order to correct the density at the fluid surface, we use the constant correction technique as described in [BK02]. Positions and velocities are updated using the Euler-Cromer scheme. The neighborhood search is computed using the parallelized compact hashing method [IABT11].

The surfaces are extracted by first mapping the particle positions to a scalar field as described in [SSP07] and then using the Marching Cubes algorithm [LC87]. The resulting meshes are rendered with POV-Ray.

Parameter setting. In all our 3D scenes, we set the reference density of water to 1000 kg/m^3 and for air particles we set it to 1 kg/m^3. The particle mass is computed as $\rho_0/(0.5h)^3$, where $0.5h$ is the initial particle spacing, which is 0.02 in most of our scenes. The gravity is set to $(0,-9.81,0)^T$. Velocities and accelerations are given in m/s, m/s^2 respectively.

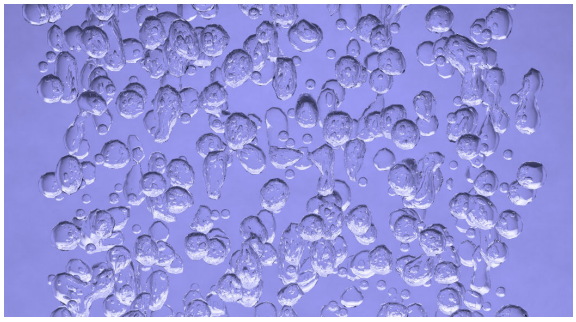

Figure 5: Rising bubbles in calm water. Bubbles are different in shape and size due to the cohesion force.

For the bubbles, we empirically found the following setting to obtain good results: the buoyancy coefficients $k_b = 14$ and $k_{max} = 6$, cohesion $k_c = 12$ and the drag coefficient k_d is set to 8 for air particles and to 3 for water. We have varied the parameters for generating air bubbles in order to show their effect on the simulation (see Sec. 5.2).

Note that the coefficients introduced for the bubble model are independent of the resolution h. By increasing the parameters k_d, k_c and k_b, the effect of the corresponding force is amplified.

5.1. Bubble flow

The basic capability of the presented bubble model is demonstrated by simulating rising bubbles in calm water (see Fig. 5). In this example scene, air particles are randomly seeded at the bottom. The initial fluid velocity is zero. According to the proposed two-way coupling, the velocity field of the fluid is influenced by the air phase and vice versa. Thereby, small scale turbulences are generated which results in a natural zig-zag flow of the bubbles without adding random disturbance. As the example shows, the prominent effects of air bubble flow can be successfully captured, e. g. merging, path instability, volume dependent buoyancy.

In this scene, 2.4 million water particles and up to $10k$ air particles are simulated. The time step is set to $0.0015s$ which guarantees a compressibility of less than 1.5%.

5.2. Bubble generation

A major contribution of the presented model is the generation of air bubbles. In contrast to existing work, air is generated in surface regions with high local velocity differences. According to the velocity based coupling, bubbles can be generated on the fly without causing numerical instabilities. The generation of air bubbles is demonstrated in two example scenes.

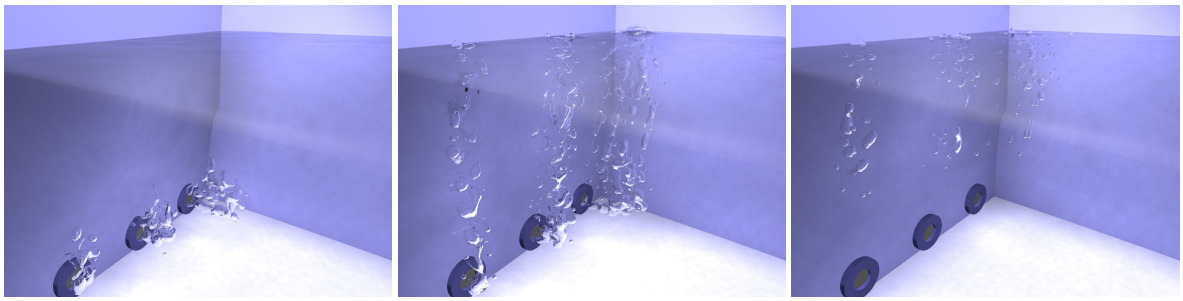

Figure 6: Underwater inflows. Water streams out of the three pipes with different velocities. The inflow in front has the lowest velocity and the inflow in the back the highest. Note that the amount of generated air scales with the velocity of the inflow. Larger air bubbles influence the liquid significantly. The simulation uses 650k water particles and up to 2k air particles.

In the first example, an inflow is simulated that is poured into a volume of water (see Fig. 1). High velocity differences occur around the inflow. Accordingly, air particles are generated. Although, the bubble flow is mainly influenced by the turbulent velocity field of the liquid, the individual flow is quite different. Bubbles are varying in size and shape according to the cohesion force. Due to the volume dependent buoyancy, larger bubbles rise faster and more straight, while smaller bubbles are mainly influenced by the liquid. According to the proposed foam model, the air bubbles float realistically on the surface before they burst. For this example, we set the average floating time t_f to $0.7s$. In this example, a liquid surface particle only generates an air particle if the velocity difference is larger than $v_t = 0.3$ and the magnitude of its velocity is larger than $v_{min} = 3$. Note that less air would be generated if v_t or v_{min} are set higher.

The second example demonstrates that the amount of generated air particles scales with the magnitude of the velocity difference. Therefore, three underwater inflows are simulated (see Fig. 6). The velocities $v_n = (x_n, 0, 0)^T$ of the water inflows are set differently, where $x_1 = 5.5$, $x_2 = 7.0$ and $x_3 = 9.0$. v_t is set to 0.75 and v_{min} to 3.5. Thereby, the velocity differences around the inflow do vary. According to the proposed heuristic (see Sec. 5.2), most of the air particles are generated by the fastest inflow, while the slowest inflow generates significantly less bubbles. Furthermore, higher inflow velocities result in larger turbulences. Since these vorticities are mapped onto the bubble flow, the bubbles indicate the liquid flow. Note that without the animation of bubbles, the liquid inflow would not be visible in this example.

Again, spatially close air particles merge. Consequently, some air bubbles get quite large, particularly for the fast inflow. These large bubbles rise very fast due to the volume dependent buoyancy force. According to the drag force, they significantly influence the liquid, as is clearly visible at the liquid surface. In both scenes, we set the time step to $0.002s$.

6. CONCLUSION

We have presented a new air bubble model for SPH. Numerical instabilities, invoked by the high density ratio at the bubbles interface, are avoided by coupling the two phases via the velocity field. Since the coupling is in both directions, small scale turbulences are naturally captured, e. g. path instability. In contrast to existing models, the bubbles are simulated with SPH and not just by solid spheres. According to the proposed cohesion force, effects like merging and deformation of bubbles can be successfully simulated. Furthermore, we have proposed a velocity based heuristic that generates air bubbles for inflows. Thereby, trapped air is animated efficiently, i.e. without explicitly simulating the air phase surrounding the liquid. We also employ a simple foam model in order to simulate floating air bubbles. By incorporating the bubble model into the PCISPH method, large time steps can be used. This allows to simulate high-resolution animations where the bubbles are small in size in comparison to the liquid volume. This is demonstrated in the example scenes.

Future work. In this work, we do not cover the interaction of air bubbles with rigid or deformable bodies. Effects like natural surface attraction of bubbles to solid surfaces would certainly further improve the realism of the simulation.

Moreover, we think that the performance can be further improved by using smaller support radii for air particles than for liquid particles. This is proposed for single-phase fluids in [DC96], [APKG07]. In scenarios, where the influence of air bubbles onto water can be neglected, the air bubble model can be applied as a post-processing step. The generation of bubbles and their flow can be sufficiently computed using the already simulated velocity field of the liquid. This could be interesting for highly turbulent fluid simulations like rivers or waves.

ACKNOWLEDGEMENTS

This project is supported by the German Research Foundation (DFG) under contract number TE 632/1- 1.

References

[APKG07] ADAMS B., PAULY M., KEISER R., GUIBAS L.: Adaptively sampled particle fluids. In *SIGGRAPH '07: ACM SIGGRAPH 2007 papers* (New York, USA, 2007), ACM Press, p. 48. 8

[BdRCS10] BIRD J. C., DE RUITER R., CORBIN L., STONE H. A.: Daughter bubble cascades produced by folding of ruptured thin films. *Nature 465* (2010), 759–762. 6

[BK02] BONET J., KULASEGARAM S.: A simplified approach to enhance the performance of smooth particle hydrodynamics methods. *Applied Mathematics and Computation 126*, 2-3 (2002), 133–155. 7

[BT07] BECKER M., TESCHNER M.: Weakly compressible SPH for free surface flows. In *SCA '07: Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation* (Aire-la-Ville, Switzerland, 2007), Eurographics Association, pp. 209–217. 3

[CPPK07] CLEARY P., PYO S., PRAKASH M., KOO B.: Bubbling and frothing liquids. *ACM Transaction on Graphics 26*, 3 (2007), 97. 2, 5

[DC96] DESBRUN M., CANI M.-P.: Smoothed Particles: A new paradigm for animating highly deformable bodies. In *Eurographics Workshop on Computer Animation and Simulation (EGCAS)* (1996), Springer-Verlag, pp. 61–76. 2, 8

[FSJ01] FEDKIW R., STAM J., JENSEN H.: Visual simulation of smoke. In *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques* (New York, USA, 2001), ACM, pp. 15–22. 3

[GH04] GREENWOOD S. T., HOUSE D. H.: Better with bubbles: enhancing the visual realism of simulated fluid. In *SCA '04: Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation* (Aire-la-Ville, Switzerland, 2004), Eurographics Association, pp. 287–296. 2

[HK03] HONG J.-M., KIM C.-H.: Animation of Bubbles in Liquid. *Computer Graphics Forum 22* (2003), 253–262. 2

[HLYK08] HONG J.-M., LEE H.-Y., YOON J.-C., KIM C.-H.: Bubbles Alive. In *SIGGRAPH '08: ACM SIGGRAPH 2008 papers* (New York, USA, 2008), ACM, pp. 1–4. 2, 3, 5

[HMT01] HADAP S., MAGNENAT-THALMANN N.: Modeling Dynamic Hair as a Continuum. *Computer Graphics Forum 20*, 3 (2001), 329–338. 2

[IABT11] IHMSEN M., AKINCI N., BECKER M., TESCHNER M.: A Parallel SPH Implementation on Multi-core CPUs. *Computer Graphics Forum* (2011). to appear. 7

[IAGT10] IHMSEN M., AKINCI N., GISSLER M., TESCHNER M.: Boundary Handling and Adaptive Time-stepping for PCISPH. In *Proc. VRIPHYS* (2010), pp. 79–88. 6

[KAG*05] KEISER R., ADAMS B., GASSER D., BAZZI P., DUTRÉ P., GROSS M.: A Unified Lagrangian Approach to Solid-Fluid Animation. In *Proceedings of the Eurographics Symposium on Point-Based Graphics* (2005), pp. 125–134. 6

[KSK10] KIM D., SONG O.-Y., KO H.-S.: A practical simulation of dispersed bubble flow. In *ACM SIGGRAPH 2010 papers* (New York, USA, 2010), SIGGRAPH '10, ACM, pp. 70:1–70:5. 2

[KVG02] KÜCK H., VOGELGSANG C., GREINER G.: Simulation and rendering of liquid foams. In *In Proc. Graphics Interface Š02 (2002)* (2002), pp. 81–88. 2

[LC87] LORENSEN W., CLINE H.: Marching cubes: A high resolution 3D surface construction algorithm. In *SIGGRAPH '87: Proceedings of the 14th annual conference on Computer graphics and interactive techniques* (New York, USA, 1987), ACM Press, pp. 163–169. 7

[LGF04] LOSASSO F., GIBOU F., FEDKIW R.: Simulating water and smoke with an octree data structure. In *SIGGRAPH '04: ACM SIGGRAPH 2004 Papers* (New York, USA, 2004), ACM, pp. 457–462. 2

[MCG03] MÜLLER M., CHARYPAR D., GROSS M.: Particle-based fluid simulation for interactive applications. In *SCA '03: Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation* (Aire-la-Ville, Switzerland, 2003), Eurographics Association, pp. 154–159. 2, 3, 6

[Mon92] MONAGHAN J.: Smoothed particle hydrodynamics. *Ann. Rev. Astron. Astrophys. 30* (1992), 543–574. 3

[Mon02] MONAGHAN J.: SPH compressible turbulence. *Monthly Notices of the Royal Astronomical Society 335*, 3 (2002), 843–852. 2

[Mon05] MONAGHAN J.: Smoothed particle hydrodynamics. *Reports on Progress in Physics 68*, 8 (2005), 1703–1759. 3

[MSKG05] MÜLLER M., SOLENTHALER B., KEISER R., GROSS M.: Particle-based fluid-fluid interaction. In *SCA '05: Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation* (New York, USA, 2005), ACM, pp. 237–244. 1, 2, 4, 6

[MYH*10] MOSS W., YEH H., HONG J.-M., LIN M. C., MANOCHA D.: Sounding liquids: Automatic sound synthesis from fluid simulation. *ACM Trans. Graph. 29*, 3 (2010), 1–13. 1

[SF95] STAM J., FIUME E.: Depicting fire and other gaseous phenomena using diffusion processes. In *SIGGRAPH '95: Proceedings of the 22nd annual conference on Computer graphics and interactive techniques* (New York, USA, 1995), ACM Press, pp. 129–136. 2

[SP08] SOLENTHALER B., PAJAROLA R.: Density Contrast SPH Interfaces. In *SCA '08: Proceedings of the 2008 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2008), pp. 211–218. 1, 2, 3, 4, 6

[SP09] SOLENTHALER B., PAJAROLA R.: Predictive-corrective incompressible SPH. In *SIGGRAPH '09: ACM SIGGRAPH 2009 Papers* (New York, USA, 2009), ACM, pp. 1–6. 2, 3, 6

[SRF05] SELLE A., RASMUSSEN N., FEDKIW R.: A vortex particle method for smoke, water and explosions. In *SIGGRAPH '05: ACM SIGGRAPH 2005 Papers* (New York, NY, USA, 2005), ACM, pp. 910–914. 3

[SSP07] SOLENTHALER B., SCHLÄFLI J., PAJAROLA R.: A unified particle model for fluid-solid interactions. *Computer Animation and Virtual Worlds 18*, 1 (2007), 69–82. 2, 7

[TSS*07] THÜREY N., SADLO F., SCHIRM S., MÜLLER-FISCHER M., GROSS M.: Real-time simulations of bubbles and foam within a shallow water framework. In *SCA '07: Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation* (Aire-la-Ville, Switzerland, 2007), Eurographics Association, pp. 191–198. 3

[ZYP06] ZHENG W., YONG J.-H., PAUL J.-C.: Simulation of bubbles. In *SCA '06: Proceedings of the 2006 ACM SIGGRAPH/Eurographics symposium on Computer animation* (Aire-la-Ville, Switzerland, 2006), Eurographics Association, pp. 325–333. 2