# Evaporation and Condensation of SPH-based Fluids

Hendrik Hochstetter
University of Siegen
Germany

Andreas Kolb
University of Siegen
Germany

## ABSTRACT

In this paper we present a method to simulate evaporation and condensation of liquids. Therefore, both the air and liquid phases have to be simulated. We use, as a carrier of vapor, a coarse grid for the air phase and mass-preservingly couple it to an SPH-based liquid and rigid body simulation. Since condensation only takes place on rigid surfaces, it is captured using textures that carry water to achieve high surface detail. The textures can exchange water with the air phase and are used to generate new particles due to condensation effects yielding a full two-way coupling of air phase and liquid. In order to allow gradual evaporation and condensation processes, liquid particles can take on variable sizes. Our proposed improved implicit surface definition is able to render dynamic contact angles for moving droplets yielding highly detailed fluid rendering.

## CCS CONCEPTS

• **Computing methodologies** → **Physical simulation**; *Shape modeling*;

## KEYWORDS

Smoothed particle hydrodynamics, fluid simulation, condensation, evaporation, implicit surface, dynamic contact angle, fluid rendering

**ACM Reference format:**
Hendrik Hochstetter and Andreas Kolb. 2017. Evaporation and Condensation of SPH-based Fluids. In *Proceedings of SCA '17, Los Angeles, CA, USA, July 28-30, 2017,* 9 pages.
https://doi.org/10.1145/3099564.3099580

## 1 INTRODUCTION

Fluid simulation has long been an area of interest in computer graphics. There are two main approaches to simulate fluids: Either using Lagrangian particles like SPH [Monaghan 2005; Müller et al. 2003] or Eulerian grids [Stam 1999] to discretize the Navier-Stokes equations. While SPH-based Lagrangian simulations are inherently mass-preserving and robustly handle free surfaces, grid-based approaches can very efficiently simulate volumetric effects like steam and smoke [Fedkiw et al. 2001], however, they have difficulties in preserving mass or volume at free surfaces.

In SPH, a wide range of phenomena has been modeled including melting and solidification [Solenthaler et al. 2007], coupling of liquids with rigid objects [Akinci et al. 2012] and gases [Macklin et al. 2014], surface tension and adhesion on surfaces [Akinci et al. 2013] and dynamics of surface active substances [Orthmann et al. 2013]. Adaptive particle sizes have been used to increase details while keeping the simulation efficient [Adams et al. 2007; Orthmann and Kolb 2012; Winchenbach et al. 2017].

Although evaporation and condensation are ubiquitous phenomena we encounter in everyday life, they have not yet been modeled and simulated in a comprehensive way. While there are some works in which liquid and gaseous fluids are coupled, e.g., to simulate burning oil [Losasso et al. 2006; Macklin et al. 2014], no two-way coupling of water and its vapor has been proposed so far. Evaporation has only been simulated by randomly generating air particles in the SPH context [Müller et al. 2005] while Tillmann and Bohn [2015] simulated condensation on rigid surfaces by generating explicitly meshed droplets from vapor transported in a grid.

In this paper, we propose the simulation of evaporation and condensation using a coarse Eulerian grid to simulate the air phase while the liquid phase is simulated using SPH particles. Condensation only takes place on rigid surfaces and is realized in sub-particle detail using textures to deposit mass into, and to generate new particles from. In order to allow for gradual evaporation, particles use variable sizes. In reality, fluids in close contact with rigid surfaces take on distinct contact angles. We propose a modified implicit surface description that allows rendering of SPH fluids with dynamic contact angles.

In summary our contributions are

- a physically plausible evaporation and condensation model that allows for
- mass-preserving coupling of grid, and particle system to simulate evaporation and condensation,
- sub-particle detail of surface wetting using textures to contain mass and
- rendering of dynamic contact angles using an improved implicit surface representation for SPH fluids.

The remainder of this paper is structured as follows. Sec. 2 discusses related work and Sec. 3 introduces necessary foundations for our algorithm which is outlined in Sec. 4. We describe the heat coupling between different systems in Sec. 5, and Sec. 6 presents our method to simulate evaporation and condensation. In Sec. 7 we introduce our improved implicit surface definition. Results are presented in Sec. 8, before Sec. 9 draws final conclusions.

## 2 RELATED WORK

Although SPH has initially been developed to simulate astrophysical problems [Gingold and Monaghan 1977; Lucy 1977], it has since found its way into many areas of research including computer graphics [Ihmsen et al. 2014b; Monaghan 2005]. In order to simulate incompressible fluids like water, the pressure can be solved using predictive-corrective [Solenthaler and Pajarola 2009] or iterative methods [Ihmsen et al. 2014a; Macklin and Müller 2013]. Two-way coupling with rigid boundaries has been realized using boundary particles [Akinci et al. 2012; Becker et al. 2009] which

*SCA '17, July 28-30, 2017, Los Angeles, CA, USA*

can also be used to take part in processes like washing off substances off rigid surfaces [Orthmann et al. 2013]. Liquids have also been dynamically coupled with gases [Macklin et al. 2014; Müller et al. 2005]. Fluid behavior on small scales is strongly influenced by surface tension. While early curvature-based [Müller et al. 2003] and inter-particle attraction forces [Becker and Teschner 2007] were not able to stably describe strong tension effects, a combination of both approaches using special kernel functions achieved the desired effects [Akinci et al. 2013]. Simulation of fluids with very high detail at moderate computational cost, can be achieved using adaptive particles. The first adaptive approach to work mass-preservingly with incompressible fluids used a temporal blending between discrete particle sizes [Orthmann and Kolb 2012]. Recently, Winchenbach et al. [2017] took the idea of adaptivity a step further by continuously adjusting particles to desired sizes by splitting, merging and redistributing mass between particles.

While there is a wide array of other phenomena that have successfully been described in SPH, like the thermodynamic processes of melting and solidification [Solenthaler et al. 2007], evaporation has, so far, only been modeled using random generation of air particles after reaching a critical temperature [Müller et al. 2005].

Foster and Metaxas [1996] first introduced grid-based simulation of the Navier-Stokes equations to computer graphics. Stam improved the simulation using unconditionally stable semi-Lagrangian advection [Stam 1999] and vorticity confinement was introduced to retain fine flow details [Fedkiw et al. 2001]. Versatile interaction of liquids, solids and gases has been realized by Losasso et al. [2006] and very detailed interaction of water with surfaces has been simulated by Wang et al. [2005]. Tillmann and Bohn [2015] used a grid to simulate an air phase that transports vapor that can be condensed on rigid surfaces using explicit meshes. As we use the grid in a similar fashion as a means to transport vapor, we will not go into a more detailed discussion of grid-based simulations.

Except for an unphysical SPH-based simulation of evaporation by Müller et al. [2005] and a grid-based condensation simulation that is only able to generate static droplets [Tillmann and Bohn 2015], the coupled processes of evaporation and condensation have not been simulated so far.

High-quality rendering of SPH fluids is mainly achieved using implicit surface definitions. These can be based on density thresholding [Müller et al. 2003] which causes a blobby appearance or on a more geometric definition using the distance from the center of mass [Zhu and Bridson 2005] which, however, can cause artefacts between neighboring particles that do not touch. These artefacts can be corrected using PCA [Solenthaler et al. 2007] or by using anisotropic kernels [Yu and Turk 2010]. Only recently, the interaction of fluids with rigid surfaces has been taken into account by Morgenroth et al. [2016] who apply a correction in order to enforce a prescribed contact angle between liquid and solid surfaces which greatly enhances optical realism.

# 3 FOUNDATIONS

## 3.1 Equations of fluid flow

For grid simulations, usually an incompressible, inviscid fluid is assumed which can be described using the Navier-Stokes equations as [Stam 1999]

$$\nabla \cdot \vec{\boldsymbol{v}} = 0 \tag{1}$$

$$\frac{\partial \vec{\boldsymbol{v}}}{\partial \tau} = -\left(\vec{\boldsymbol{v}} \cdot \nabla\right) \vec{\boldsymbol{v}} - \nabla p + \vec{\boldsymbol{f}}, \tag{2}$$

where $\vec{\boldsymbol{v}}$ denotes velocity, $p$ pressure and $\vec{\boldsymbol{f}}$ external forces. Throughout the text $\tau$ denotes the time. Temperature $T$ and density $\rho$ are just advected with the flow according to

$$\frac{dT}{d\tau} = -\left(\vec{\boldsymbol{v}} \cdot \nabla\right) T \tag{3}$$

$$\frac{d\rho}{d\tau} = -\left(\vec{\boldsymbol{v}} \cdot \nabla\right) \rho. \tag{4}$$

In order to make warm air rise and dense air sink, explicit buoyancy terms can be used as

$$\vec{\boldsymbol{f}}_{\text{buoy}} = -\alpha \rho \hat{\boldsymbol{g}} + \beta \left(T - T_{\text{amb}}\right) \hat{\boldsymbol{g}}, \tag{5}$$

where $\hat{\boldsymbol{g}}$ is the normalized direction of gravity, $\rho$ the density and $\alpha$ and $\beta$ are user-defined parameters [Fedkiw et al. 2001].

## 3.2 Smoothed Particle Hydrodynamics

In SPH, quantities of a continuous field $A$ are defined through a discrete set of particles $i$ with quantities $A_i$ at particle positions $\boldsymbol{x}_i$. The continuous field is reconstructed by interpolating particle quantities using a weighting kernel $W$ with compact support $h$ as

$$A_i = \sum_j A_j V_j W_{ij}, \tag{6}$$

where $V_j$ is the particle's dynamic volume [Orthmann et al. 2013] and $W_{ij} = W(\|\boldsymbol{x}_i - \boldsymbol{x}_j\|, h) = W_i(\boldsymbol{x}_j, h)$ a compact kernel with support radius $h$ [Monaghan 2005]. The above interpolation is not able to reconstruct fields of zeroth order which in some cases, however, is necessary and can be enforced using a corrected interpolation as

$$\tilde{A}_i = \sum_j A_j V_j \tilde{W}_{ij}, \tag{7}$$

with $\tilde{W}_{ij} = W_{ij} / \sum_k V_k W_{ik}$ [Bonet and Kulasegaram 2002].

## 3.3 Heat conduction

As evaporation and condensation are thermodynamic processes, they depend on the temperature which has to be simulated. In a homogeneous medium heat is transferred as

$$\frac{dT}{d\tau} = \frac{\nabla \cdot (\kappa \nabla T)}{\rho C_p}, \tag{8}$$

where $C$ denotes the specific heat capacity and $\kappa$ the heat conductivity. Cleary and Monaghan [1999] arrive at

$$\frac{dT_i}{d\tau} = \frac{V_i}{m_i C_i} \sum_j \frac{4\kappa_i \kappa_j}{\kappa_i + \kappa_j} V_j \frac{\left(T_i - T_j\right)}{\|\boldsymbol{x}_{ij}\|} \nabla W_{ij} \tag{9}$$

to model heat conduction for homogeneous and heterogeneous materials in SPH.

### 3.4 Evaporation and condensation

Evaporation and condensation depend on the states of the liquid and air phase and follow the difference between the saturation vapor pressure at the surface temperature and the partial pressure of the vapor in the air phase, i.e., the vapor pressure [Shah 2014; Smith et al. 1994]. The vapor pressure can easily be described using the ideal gas equation [Rogers and Yau 1996] as

$$p_c = \frac{m_c R_w (T_c - 273.15\ ^\circ\mathrm{C})}{V_c}, \tag{10}$$

where $m_c$ is the water vapor mass in the air, $V_c$ its volume, $T_c$ the temperature in °C and $R_w$ the specific gas constant of water. The saturation vapor pressure at surface temperature $T_s$ can be calculated from Magnus' formula [Rogers and Yau 1996] as

$$p_s^{\mathrm{sat}} = 611.2\,\mathrm{Pa} \cdot \exp\left(\frac{17.62 \cdot T_s}{243.12\ ^\circ\mathrm{C} + T_s}\right). \tag{11}$$

To calculate the time rate of change of mass of a liquid surface due to evaporation, Smith et al. [1994] proposed the following formula based on empirical data

$$\frac{\partial m_s}{\partial \tau} = A_s \left(a + b \cdot \|\vec{v}\|\right) \left(p_s^{\mathrm{sat}} - p_c\right). \tag{12}$$

It depends on the surface of contact $A_s$, the saturation vapor pressure of the liquid surface $p_s^{\mathrm{sat}}$, the vapor pressure of the air phase $p_c$, the air velocity at the interface $\vec{v}$, and two parameters $a$ and $b$.

## 4 ALGORITHM OVERVIEW

Our algorithm can be subdivided into three phases: Phase 1 comprises a standard SPH-simulation including rigid particles and adaptive particle sizes. Phase 2 is a standard grid-based simulation of the air phase with vapor and temperature advection. In Phase 3 the couplings between the grid based air phase, the SPH-based liquid phase and rigid objects are realized. The coupling comprises heat transfer (see Sec. 5) and mass transfer due to evaporation and condensation (see Sec. 6). In order to add sub-particle details of evaporation and condensation, rigid objects use textures into which water can condense and evaporate from. Alg. 1 gives an overview over our algorithm. Fig. 1 shows the paths of heat transport and mass transport in more detail. Except for the deposition of mass, we do not simulate dynamic behavior of water inside textures. However, there are different approaches to simulate texture-based water flows on surfaces which could be included [El Hajjar et al. 2009; Wang et al. 2007].

### 4.1 Particle simulation

The SPH-simulation is based on standard components and comprises a PCISPH pressure solver [Solenthaler and Pajarola 2009], rigid particles and surface tension and adhesion [Akinci et al. 2013, 2012]. As evaporation and condensation takes place at surfaces, our approach depends on a stable measure of a particle's surface area $A_i$ which we calculate as proposed by Orthmann et al. [2013].

As the simulation of evaporation causes particles to lose mass, we allow particles to have adaptive sizes. In literature, particle sizes are usually adjusted by modifying the support radius. If particles of different support radii interact, this can, however, lead to severe instabilities [Orthmann and Kolb 2012; Winchenbach et al. 2017].

**Algorithm 1** Overview over our proposed simulation. Colored parts are added by our approach. Red color denotes heat transport and green color denotes mass transport. Coupling directions are highlighted in bold face.

```
1:  while Simulating do
Phase 1 - Simulate particles (liquid and rigid)
2:      Move particles according to forces of last iteration
3:      Neighborhood search
4:      Heat transport
5:      External forces
6:      Pressure solve
Phase 2 - Simulate grid (air phase)
7:      Classify cells as air or liquid/rigid
8:      Add forces, enforce boundary conditions
9:      Advection of velocity, vapor and temperature
10:     Diffusion of heat, vapor
11:     Pressure projection
Phase 3 - Coupling of grid, particles, texture
12:     Neighborhood search:
            Texture ↔ (Rigid) particles ↔ Grid ↔ Texture
13:     Heat transfer: particles ↔ grid
14:     Temperature interpolation: rigid particles → texture
15:     Particle evaporation: particles → grid
16:     Sub-particle evaporation and condensation:
            grid ↔ texture
17:     Particle condensation: texture → particles
18:     Mass redistribution and merging of particles
19:     Particle path force
20: end while
```

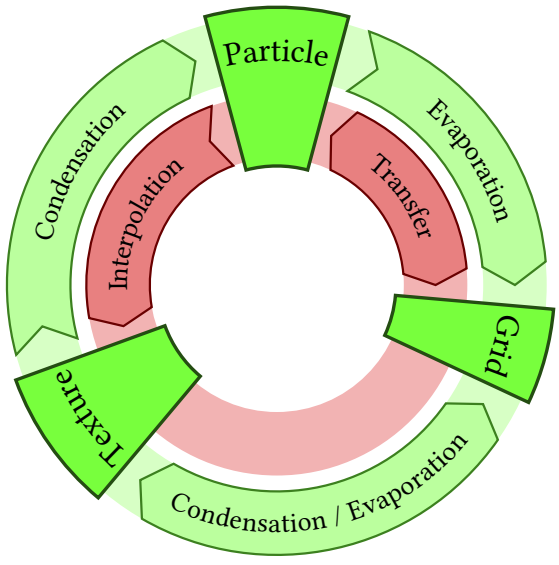


**Figure 1: Mass transfer (outer green circle) and heat transfer (inner red circle) between different systems.**

Thus, we leave the support radius unchanged and only scale particle masses using a weighting term $w_i \in [0,1]$. Due to the fact that the weighted mass $m_{w_i} = w_i m_i$ is used in the density calculation, the fluid volume is automatically scaled accordingly.

### 4.2 Grid simulation

The grid is mainly used as a means to simulate the air phase and transport water vapor, thus, we use a coarse grid with a cell size larger than two times the particle support radius. We use semi-Langrangian advection [Stam 1999], a simple conjugate gradient solver to enforce a divergence-free velocity field and vorticity confinement [Fedkiw et al. 2001] to retain fine details of the flow dynamics. The liquid and rigid particles are considered as boundary condition for the grid simulation, i.e., cells are classified as air or solid/liquid.

### 4.3 Texture-based rigid surface representation

To allow for a sub-particle detail evaporation and condensation at rigid surfaces, we use textures as a means to describe condensed water mass. For our approach to work, we pre-compute several static textures, i.e., textures that store the world position $\boldsymbol{x}_t$ of each texel $t$ and surface area spanned by the texel $A_t$. In order to transfer mass from the texture to particles, we use a texture that stores particle seeding positions. These are created using Poisson disk sampling [Bridson 2007] to prevent overlapping seeding positions which cause instabilities if particles are simultaneously emitted at very close positions. Additionally, we use a texture that stores the maximum amount of mass that is allowed to be deposited in each texel $m_t^{\max}$.

The only dynamic textures are the mass texture which stores the actual amount of water at each texel $m_t$, the temperature texture $T_t$ and a wetting history [Wang et al. 2005]. The latter allows particle paths along surfaces to be rendered and to prevent emitting particles in the neighborhood of texels that are already covered by particles.

### 4.4 Neighborhood search

As depicted in Fig. 1, our simulation includes direct interactions between every pair of systems, thus, neighborhoods for grid-cells, texels and rigid and liquid particles have to be calculated. As rigids are static in our simulation and textures are fixed to their rigid bodies, neighborhoods between rigid particles, grid cells and texels have to be calculated only once. For liquid particles, however, neighborhoods have to be recomputed in every time step. In order to calculate neighborhoods between grid cells, particles and texels, we follow the cell-based approach of Green [2009].

## 5 HEAT TRANSFER

Thermodynamic processes like evaporation and condensation depend on the temperatures of the different phases. Thus, all components of the simulation first have to be coupled in terms of heat transport to achieve realistic behavior. Heat transport between particles is realized according to Cleary and Monaghan [1999]. Heat, however, has to be transferred across phase transitions between grid cells and particles and to the surface textures.

### 5.1 Heat transfer between grid and particles

To transfer heat between grid cells and particles we start from Fourier's law [Lienhard IV and Lienhard V 2016]

$$\frac{dT}{d\tau} = \frac{\frac{dQ}{d\tau}}{\rho \cdot C} = \frac{-\kappa A \Delta T}{\rho \cdot C} \tag{13}$$

where $A$ is the interface area over which heat is transferred, $\kappa$ is the conductivity at the interface and $C$ is the specific heat capacity.

Particles $i$ can interact with more than one neighboring cell $c$, and cells interact with more than one particle, thus, to calculate heat transfer we have to iterate over neighborhoods. The time rate of change of temperature due to heat transfer across the air-liquid or air-rigid interface then can be expressed as

$$\frac{dT_i}{d\tau} = \frac{1}{\rho_i C_i} A_i \sum_c \frac{4\kappa_i \kappa_c}{\kappa_i + \kappa_c} \left(T_i - T_c\right) \alpha_{ic} \tag{14}$$

for particles $i$ and

$$\frac{dT_c}{d\tau} = \frac{1}{\rho_c C_c} \sum_j \alpha_{jc} \frac{4\kappa_j \kappa_c}{\kappa_j + \kappa_c} \left(T_c - T_j\right) A_j \tag{15}$$

for cells $c$. The coefficients $\alpha_{ic}$ denote trilinear interpolation weights for position $\boldsymbol{x}_i$ that are used to weight cell-particle interactions. Note that the temperature flux is not antisymmetric as the actual physical quantity transferred is heat.

### 5.2 Heat transfer to texture

As textures are only used as a means to increase the surface detail of rigid bodies, we don't want to add much computational overhead. Thus, we don't directly include textures in heat transport but only interpolate the rigid particle temperatures onto the texels in each time step using a corrected SPH interpolation, see Eq. 7.

## 6 EVAPORATION AND CONDENSATION

As evaporation and condensation only take place at rigid and liquid surfaces, we base our model on the general notion of surface elements or surfels. We will denote surfels with $s$ which can mean texel or liquid particle and denote cell values with $c$. We first describe the general model of evaporation and condensation before we detail the specific algorithmic solution to allow a mass-preserving coupling between texels, grid cells and particles.

In order to allow evaporation and condensation to transfer variable amounts of mass between air and liquid, particles of variable size are used. However, we try to keep particles as close as possible to $w_i = 1$ to prevent a system of a lot of very tiny particles. This is done by allowing particles to exchange mass among each other and by merging neighboring particles.

### 6.1 Modelling evaporation and condensation

Although the time rate of change of mass in Eq. 12 originally only described evaporation, we will also use it to model condensation. Depending on the sign of $\frac{\partial m_s}{\partial \tau}$, either evaporation or condensation

takes place which allows us to separately describe both rates as

$$\text{evaRate}(c,s) = \max\left(\frac{\partial m_s}{\partial \tau}, 0\right), \quad (16)$$

$$\text{condRate}(c,s) = -\min\left(\frac{\partial m_s}{\partial \tau}, 0\right), \text{ with } b = 0, \quad (17)$$

for a cell $c$ and a surfel $s$. To get spatially smooth values, we use trilinear interpolation of cell quantities at position $\boldsymbol{x}_s$. As the velocity term in Eq. 12 introduces energy, it can only cause an increase of evaporation but not of condensation, thus, we remove its contribution in case of a condensation process by setting $b = 0$. In order to guarantee mass-preservation, the same amount of mass that is taken from the liquid has to be transferred to the air phase and vice versa, i.e., we have to enforce $\frac{\partial m_s}{\partial \tau} = -\frac{\partial m_c}{\partial \tau}$.

## 6.2 Texel evaporation and condensation

As the model of evaporation and condensation is based mainly on temperatures, it can easily calculate evaporation or condensation rates that would cause either negative masses or make texels exceed their maximum allowed mass $m_t^{\max}$. Thus, a balancing step is necessary that guarantees that both the cell's vapor mass and the surfel's water mass stay non-negative. Additionally, the maximum amount of mass $m_t^{\max}$ of texels must not be exceeded.

Our approach works by taking the evaporation and condensation rates above only as initial estimates and correcting them using scaling factors. Alg. 2 gives a detailed description of evaporation and condensation using textures. First the texel mass is bounded. Evaporation is scaled by factor $w_t^{\text{eva}} \in [0,1]$ which enforces non-negativity of texel masses while the factor $w_t^{\text{cond}} \in [0,1]$ is applied to condensation and enforces a maximum texel mass of $m_t^{\max}$. After properly scaling the mass transfer to respect the texel bounds, a third scaling factor $w_c^{\text{cond}}$ is applied to condensation that enforces non-negativity of vapor mass in cells. After scaling, the masses of cells and texels are updated.

## 6.3 Evaporation and condensation of particles

While particles can directly evaporate by transferring mass to the grid, condensation is only realized through the texture. In order to generate particles from texels due to condensation, we use our precomputed texture that stores particle seeding positions. Particle condensation directly takes mass from texels and only takes place if sufficient mass is present, thus, no special balance has to be calculated.

The same is not true for particle evaporation for which we apply a similar correction as outlined in Alg. 2. However, adaptive particle masses are determined using weighting terms $w_i$, which thus have to be adjusted instead of changing the particle mass, directly.

When a texel has just been in contact with a particle, we exclude the texel from condensation for a small amount of time using a wetting history texture [Wang et al. 2005]. This achieves two goals: Firstly, it prevents fluctuating evaporation and condensation of particles back and forth into the texture and, secondly, it allows for a realistic rendering of particle paths down surfaces.

**Algorithm 2** Evaporation from and condensation into textures.

Phase 1 - Initial estimate

**for all** Texel t **do**
  **for all** Cell c **do**
    $m_{c\leftarrow t}^{\text{eva}} \leftarrow \text{evaRate}(c,t) \cdot d\tau$ ▷ see Eq. 16
    $m_{t\leftarrow c}^{\text{cond}} \leftarrow \text{condRate}(c,t) \cdot d\tau$ ▷ see Eq. 17
  **end for**
**end for**

Phase 2 - Balance

**for all** Texel t **do**
  $w_t^{\text{eva}} \leftarrow \min\left(\frac{m_t}{\sum_c m_{c\leftarrow t}^{\text{eva}}}, 1\right)$ ▷ Keep texel non-negative
  $w_t^{\text{cond}} \leftarrow \min\left(\frac{m_t^{\max} - m_t}{\sum_c m_{t\leftarrow c}^{\text{cond}}}, 1\right)$ ▷ Keep texel mass below $m_t^{\max}$
  **for all** Cell c **do**
    $m_{c\leftarrow t}^{\text{eva}} \leftarrow m_{c\leftarrow t}^{\text{eva}} \cdot w_t^{\text{eva}}$ ▷ Scale mass transport
    $m_{t\leftarrow c}^{\text{cond}} \leftarrow m_{t\leftarrow c}^{\text{cond}} \cdot w_t^{\text{cond}}$
  **end for**
**end for**
**for all** Cell c **do**
  $w_c^{\text{cond}} \leftarrow \min\left(\frac{m_c - \sum_t m_{c\leftarrow t}^{\text{eva}}}{\sum_t m_{t\leftarrow c}^{\text{cond}}}, 1\right)$ ▷ Keep cell non-negative
**end for**

Phase 3 - Update mass

**for all** Texel t **do**
  $m_t \leftarrow m_t + \sum_c (m_{t\leftarrow c}^{\text{cond}} \cdot w_c^{\text{cond}} - m_{c\leftarrow t}^{\text{eva}})$
**end for**
**for all** Cell c **do**
  $m_c \leftarrow m_c - \sum_t (m_{t\leftarrow c}^{\text{cond}} \cdot w_c^{\text{cond}} - m_{c\leftarrow t}^{\text{eva}})$
**end for**

## 6.4 Dynamic particle adjustment

As particles can shrink due to evaporation, particles of different sizes may interact. Recently, Winchenbach et al. [2017] proposed an adaptive SPH simulation with continuously adjustable particle size that builds on particle merging and redistribution of mass among neighboring particles in order to arrive at prescribed sizes. In order to prevent particles from interacting that have strongly different sizes, we adopt the same idea.

In our case, we want to achieve particles of uniform size $w_i = 1$, thus, particles are allowed to merge if the sum of their weights does not exceed 1, else mass between particles of different size can be redistributed in order to make particles more uniform.

In order to merge neighboring particles $i$ and $j$ to form a new particle $n$, we can just use their weights. The new weight is the sum of the old weights $w_n = w_i + w_j$. All other quantities, except for the mass which remains constant, are determined using a weighted average as

$$\boldsymbol{x}_n = \frac{\boldsymbol{x}_i w_i + \boldsymbol{x}_j w_j}{w_n} \quad (18)$$

to guarantee conservation of momentum [Winchenbach et al. 2017]. After merging only particle $n$ remains and $i$ and $j$ are removed. In case of mass redistribution, only a the fraction $w_i - \frac{w_i + w_j}{2}$ of the larger particle $i$ is redistributed to the smaller particle $j$. And only the remaining quantities of particle $j$ are updated like above.

## 7 RENDERING

In order to achieve a realistic fluid appearance, it is crucial to have a proper interaction of fluids with rigid surfaces. Although fluid rendering that uses implicit surface definitions is not new, only recently Morgenroth et al. [2016] proposed a method to achieve prescribed static contact angles at rigid surfaces. We start from the original level-set definition of Zhu and Bridson [2005]

$$\phi(\boldsymbol{x}) = \|\boldsymbol{x} - \bar{\boldsymbol{x}}(\boldsymbol{x})\| - \bar{r}(\boldsymbol{x}) \tag{19}$$

where $\bar{\boldsymbol{x}}(\boldsymbol{x}) = \Sigma_i \boldsymbol{x}_i \bar{W}_i(\boldsymbol{x})$ is the weighted average position and $\bar{r}(\boldsymbol{x}) = \Sigma_i r_i \bar{W}_i(\boldsymbol{x})$ the weighted average particle radius and propose to apply two subsequent correction steps to $\phi(\boldsymbol{x})$ to achieve dynamic contact angles depending on the velocity of the fluid.

Deserno [2004] derived a parametric description for the fluid meniscus at a vertical solid wall for arbitrary angles between the horizontal plane and the fluid surface $\psi_0$ as

$$\begin{aligned} x(s,\psi_0) &= l\frac{\frac{s}{l}\cosh\frac{s}{l} + (\frac{s}{l}\cos\frac{\psi_0}{2} - (1-\cos\psi_0))\sinh\frac{s}{l}}{\cosh\frac{s}{l} + \cos\frac{\psi_0}{2}\sinh\frac{s}{l}} \\ y(s,\psi_0) &= l\frac{2\sin\frac{\psi_0}{2}}{\cosh\frac{s}{l} + \cos\frac{\psi_0}{2}\sinh\frac{s}{l}}, \end{aligned} \tag{20}$$

where $y(s,\psi_0)$ describes the vertical offset from the horizontal plane and $l$ is the capillary length which we set to the particle radius. $x(s,\psi_0)$ is close to identity, i.e., $s \approx x(s,\psi_0)$ for moderate angles $\psi_0$ so that $y(s,\psi_0)$ can be used to correct the surface distance function as proposed by Morgenroth et al. [2016].

As the derivation in Eq. 20 only describes an offset from the horizontal plane, in practice, the correct initial contact angle between fluid and rigid has to be enforced. Therefore, Morgenroth et al. [2016] mirror ghost particles across solid walls which, however, can cause problems if fluid particles are on both sides of thin rigid walls and is costly due to the sampling of additional particles. We thus propose an explicit correction which in turn extrudes a projected footprint of the fluid onto the wall. Fig. 2 (left) shows a schematic of our approach. Assume a fluid is in close vicinity to a wall with unit normal $\vec{\boldsymbol{n}}$, the weighted average position is $\bar{\boldsymbol{x}}$ and the grid point to evaluate the signed distance for is $\boldsymbol{x}$. Then we adjust the distance function according to

$$\begin{aligned} \phi(\boldsymbol{x}) &= \|\boldsymbol{x} - \bar{\boldsymbol{x}}(\boldsymbol{x})\| - \left(\frac{\bar{r}(\boldsymbol{x})}{\sin(\beta)} + d(\boldsymbol{x})\right), \text{ with} \\ \beta &= \arccos\left(\max\left(\frac{\bar{\boldsymbol{x}} - \boldsymbol{x}}{\|\bar{\boldsymbol{x}} - \boldsymbol{x}\|}\cdot\vec{\boldsymbol{n}}, 0\right)\right). \end{aligned} \tag{21}$$

Using the max restricts the correction to work in the direction of the solid wall. In a next step we calculate the term $d(\boldsymbol{x})$ which allows to enforce dynamic wetting angles as depicted in Fig. 2 (right). In general, there is a dependency between the capillary number, which relates viscosity and velocity to surface tension, and the third power of the dynamic contact angle [Kister 1993]. As surface tension and viscosity are constant, we can describe the receding and advancing contact angles as functions of the velocity as

$$\begin{aligned} \alpha_{\text{rec}}(\vec{\boldsymbol{v}}) &= \frac{\pi}{2} - A_{\text{rec}}\|\vec{\boldsymbol{v}}\|^{\frac{1}{3}} \\ \alpha_{\text{adv}}(\vec{\boldsymbol{v}}) &= \frac{\pi}{2} + A_{\text{adv}}\|\vec{\boldsymbol{v}}\|^{\frac{1}{3}}, \end{aligned} \tag{22}$$

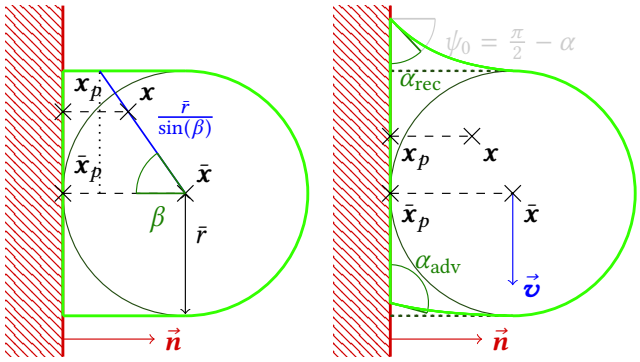


**Figure 2: Adjustment of distance function according to advancing $\alpha_{\text{adv}}$ and receding $\alpha_{\text{rec}}$ angles yields nicely rendered details at the surface (red) and dynamic appearance even for single particles (green outline).**

where $A_{\text{rec}}$ and $A_{\text{adv}}$ control the influence of the velocity. Using two different control parameters allows to capture hysteresis effects that can cause advancing and receding angles to differ.

The actual dynamic contact angle to achieve is determined by the relative projected positions of the weighted average particle position $\bar{\boldsymbol{x}}$ and the grid position of evaluation $\boldsymbol{x}$. These are orthogonally projected onto the surface to yield $\bar{\boldsymbol{x}}_p$ and $\boldsymbol{x}_p$. The distance between these points is projected onto the velocity vector as $p = \left(\frac{(\boldsymbol{x}_p - \bar{\boldsymbol{x}}_p)\cdot\vec{\boldsymbol{v}}}{\|(\boldsymbol{x}_p - \bar{\boldsymbol{x}}_p)\|\cdot\|\vec{\boldsymbol{v}}\|}\right)$, and the contact angle $\alpha$ is then varied between $\alpha_{\text{base}}$, i.e., the static contact angle, and $\alpha_{\text{rec}}$ and $\alpha_{\text{adv}}$ as

$$\alpha(p) = \begin{cases} -1 \le p < 0 & p^4 \cdot \alpha_{\text{rec}}(\vec{\boldsymbol{v}}) + (1-p^4)\cdot\alpha_{\text{base}} \\ 0 \le p \le 1 & p^{\frac{1}{4}} \cdot \alpha_{\text{adv}}(\vec{\boldsymbol{v}}) + (1-p^{\frac{1}{4}})\cdot\alpha_{\text{base}} \end{cases}, \tag{23}$$

The power terms are used to force the receding angle to form a narrow tail and the advancing front to form a more drop-like shape. Finally the corrections are applied by plugging

$$d(\boldsymbol{x}) = y\left(\|\boldsymbol{x}_p - \boldsymbol{x}\|, \frac{\pi}{2} - \alpha(p)\right) \tag{24}$$

into Eq. 21. Fig. 3 shows a how a droplet consisting of a single particle is modified using our method. In order to not make the

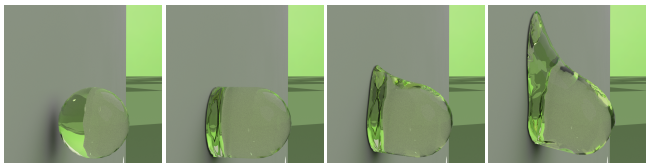

**Figure 3: A drop of one particle is modified using our proposed approach. Left: Unmodified Drop, middle left: drop with projected footprint, middle right: drop with moderate downward velocity, right: drop with fast downward velocity.**

surface correction expand from fluids far away to the surface, in practice we restrict the correction terms to distances $\|\bar{\boldsymbol{x}} - \bar{\boldsymbol{x}}_p\| < \bar{r}$ which has, however, been omitted in Fig. 3 to make the corrections more articulate. Fig. 4 shows a comparison of an uncorrected surface extraction and our proposed method in one of our demo scenes.

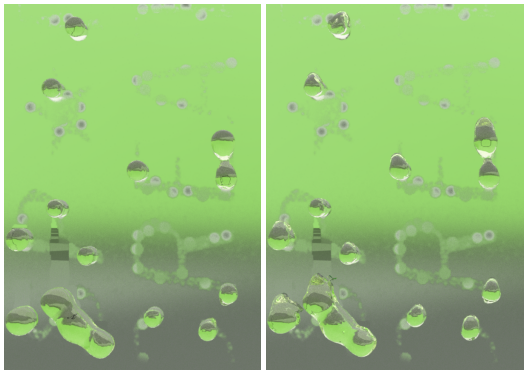

**Figure 4: Uncorrected surface rendering (left) and our proposed modification to render dynamic contact angles (right).**

## 8 RESULTS

We tested our proposed simulation on four different scenes. In order to evaluate evaporation, we put a liquid drop on a hot surface (see Fig. 6). In the SCA Logo Tex scene, we impregnated a mirror and blow humid air onto it using purely texture-based evaporation and condensation (see Fig. 5) while in the SCA Logo scene, we also emit particles from the texture seeding points (see Fig. 7). In the Glass scene, the outside of a glass filled with a cold liquid is steamed with a stream of warm vapor from the left hand side while the liquid inside remains at rest (see Fig. 8). All simulations have been carried out using an NVIDIA GeForce GTX Titan with 6 GiB VRAM. The simulation framework has been implemented using C++, CUDA 8.0. Tab. 1 shows a summary of the resolutions and timings of our demo scenes. All scenes were rendered using Mitsuba [Jakob 2010] and our modified surface extraction. Surface steaming effects were rendered using a rough material where the roughness was controlled by the water mass textures. The impregnation of the SCA logo has been realized by adjusting the maximum mass texture.

**Table 1: Scenes with particle '#Ptcl' and rigid particle '#RigPtcl' counts, grid '#Cell' and texture '#Texel' resolution and run times. 'Ptcl' gives the time spent for the SPH simulation, 'Grid' the run time of the grid simulation. The remaining timings describe the coupling: 'Neigh' denotes the time for neighborhood searches, 'Heat' for heat transfer and interpolation, and 'Eva' for evaporation and condensation, $d\tau$ denotes the corresponding simulation time step.**

| Scene | Resolution | | | | Run time per step (ms) | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | #Ptcl | #RigPtcl | #Cell | #Texel | Ptcl | Grid | Neigh | Heat | Eva | $d\tau$ |
| Drop (no tex) | 1207 | 5108 | $64^3$ | – | 18 | 65 | 2 | 6 | 1 | 4 |
| Drop | 1207 | 5108 | $64^3$ | $512^2$ | 18 | 65 | 11 | 6 | 26 | 4 |
| SCA Logo Tex | – | 28 $K$ | $64^3$ | $1024^2$ | – | 60 | – | 16 | 15 | 5 |
| SCA Logo | 135 | 28 $K$ | $64^3$ | $1024^2$ | 13 | 60 | 8 | 17 | 31 | 5 |
| Glass | 174 $K$ | 89 $K$ | $64^3$ | $1024^2$ | 210 | 31 | 32 | 23 | 40 | 2 |

In the drop scene (see Fig. 6), the initial liquid and air temperatures were set to 20°C while the ground plane was set to a temperature of 150°C causing a fast evaporation of the liquid. We simulated two versions of the drop in order to assess the overhead of using textures, while the general simulation outcome was not changed. Due to the comparably low number of particles, the run time was mainly determined by the grid solver which took 65 *ms* per time step while the particle simulation only took 18 *ms* and the coupling between air and liquid phase only 9 *ms* (see line 'Drop (no tex)' in Tab. 1). When adding textures to the simulation (see line 'Drop' in Tab. 1), the timings for the coupling increased to 53 *ms*. However, due to the fact that we only interpolate heat from rigid particles to texels, the run time for heat transfer does not measurably increase when using textures.

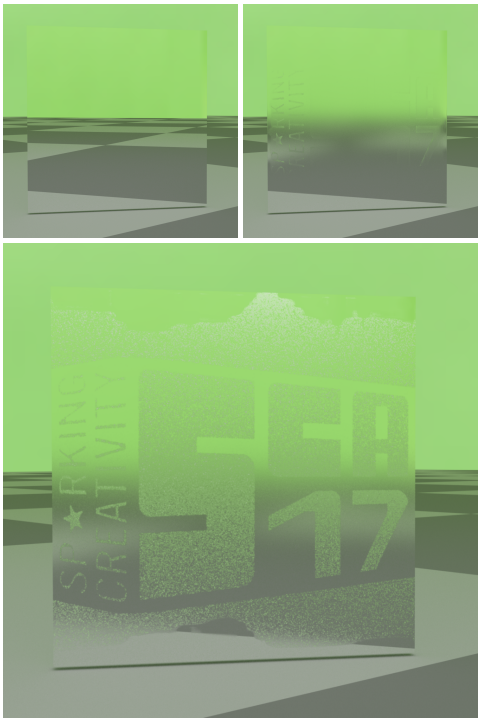

**Figure 5: A mirror is steamed by humid air revealing the impregnated SCA logo.**

For the SCA Logo scene, we also simulated two versions, one that uses only textures (SCA Logo Tex) and one that uses the full simulation proposed. In the logo scene, moist air at 20°C is blown onto a cold mirror at 4°C which causes condensation to set in. The SCA Logo Tex scene demonstrates that very fine surface detail can be captured using our proposed coupling. The run time was again dominated by the grid solver with 60 *ms* per time step while the coupling only took 31 *ms* in total. As the neighbor search for rigid-texel coupling is done in a pre-processing step it did not introduce any additional cost per time step. When additionally introducing liquid particles into the scene, neighbor search has to be performed in each step increasing the run time for the coupling to 56 *ms*.

The glass scene demonstrates that our method is able to properly work in a well balanced way, i.e., allowing to simulate a cold fluid at rest that does not evaporate while particles in the vicinity condense at the outside of the glass. The liquid simulation for the glass scene

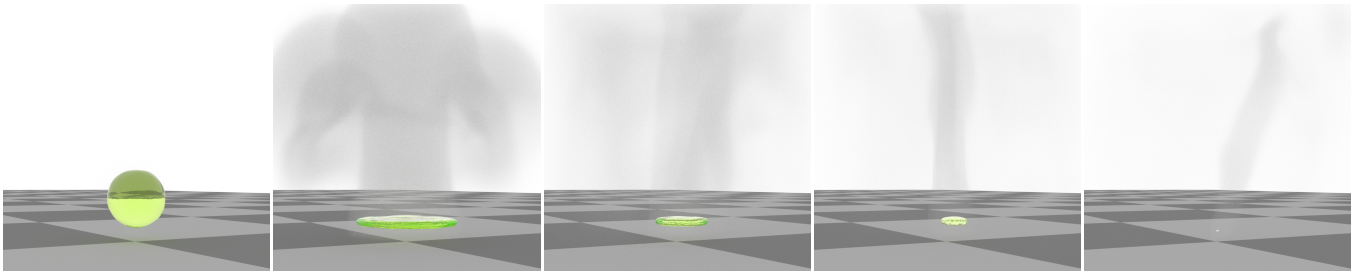

**Figure 6: A spherical drop of water is dripped onto a hot surface, evaporates and transfers its mass into the grid simulation.**

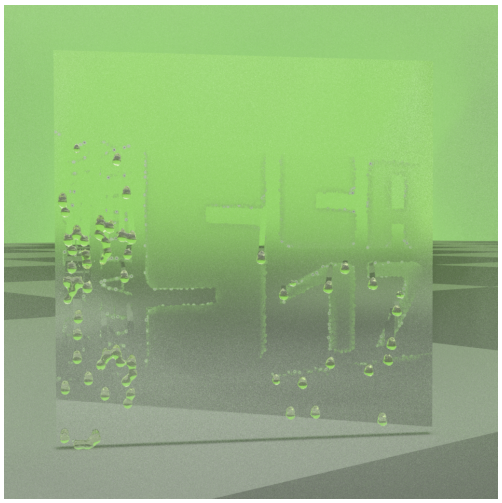

**Figure 7: Blowing moist air onto the impregnated SCA logo causes particles to condense.**

took 210 *ms* per time step and dominated the overall run time. The grid solver only took 31 *ms* and the coupling took 95 *ms* in total.

Our marching cubes [Lorensen and Cline 1987] based renderer runs solely on the CPU. Rendering times for the Glass scene are 31.7 *s* using the original surface definition and 53.8 *s* using our approach, i.e., our proposed modification lead to an overhead of 70%. For the SCA Logo scene rendering times were 9.3 *s* per frame without and 14.6 *s* with modification, i.e., an overhead of 60%.

*Limitations.* As evaporating particles can become very small, they sometimes are able to penetrate rigid bodies. This a known limitation of current adaptive approaches [Winchenbach et al. 2017]. Since we used explicit meshes of large triangle counts, the overhead for the proposed implicit surface definition was quite big. It could be reduced by using more efficient surface representations.

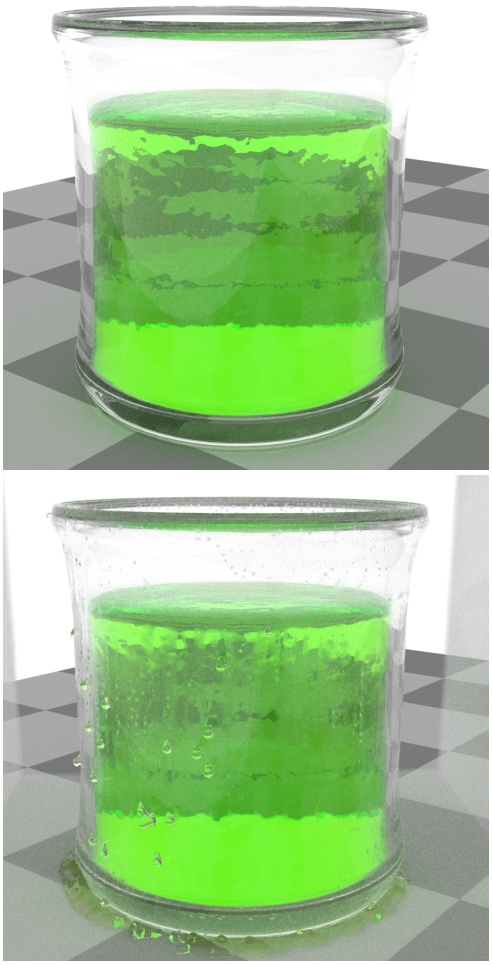

**Figure 8: A glass with cold liquid surrounded by moist air (top) causes water to condense at the glass surface (bottom).**

## 9 CONCLUSIONS

In this paper, we described the modelling and simulating of the evaporation and condensation of SPH-based fluids. The air phase is modeled using a coarse Eulerian grid solver while the liquid phase is based on an adaptive SPH solver. In order to achieve very fine surface detail, we use textures on rigid objects into which mass can be transferred. The texels exchange mass with the grid solver and if sufficient mass has been gathered, are used to generate particles. Particles can evaporate by transferring mass into grid cells which in turn yields a mass preserving cycle of evaporation and condensation. In order to achieve realistic high-quality surface renderings of fluids, we proposed a modified implicit surface definition that is able to enforce static and dynamic contact angles of fluids at rigid surfaces.

## REFERENCES


Bart Adams, Mark Pauly, Richard Keiser, and Leonidas J. Guibas. 2007. Adaptively sampled particle fluids. *ACM Trans. Graph.* 26 (2007).

Nadir Akinci, Gizem Akinci, and Matthias Teschner. 2013. Versatile Surface Tension and Adhesion for SPH Fluids. *ACM Trans. Graph.* 32, 6 (2013), 182:1–182:8.

Nadir Akinci, Markus Ihmsen, Gizem Akinci, Barbara Solenthaler, and Matthias Teschner. 2012. Versatile Rigid-Fluid Coupling for Incompressible SPH. *ACM Trans. Graph.* (2012), 62:1–62:8.

Markus Becker and Matthias Teschner. 2007. Weakly compressible SPH for free surface flows. In *Proc. Symp. Comp. Anim.* 209–217.

Markus Becker, Hendrik Tessendorf, and Matthias Teschner. 2009. Direct Forcing for Lagrangian Rigid-Fluid Coupling. *IEEE Trans. Vis. & Comp. Graph.* 15, 3 (2009), 493–503.

Javier Bonet and Sivakumar Kulasegaram. 2002. A simplified approach to enhance the performance of smooth particle hydrodynamics methods. *J. Applied Mathematics & Computation* 126, 2-3 (2002), 133–155.

Robert Bridson. 2007. Fast Poisson disk sampling in arbitrary dimensions.. In *SIGGRAPH sketches*. 22.

Paul W. Cleary and Joseph J. Monaghan. 1999. Conduction Modelling Using Smoothed Particle Hydrodynamics. *J. Comput. Phys.* 148 (1999), 227–264.

Markus Deserno. 2004. The shape of a straight fluid meniscus. (2004). http://www.cmu.edu/biolphys/deserno/pdf/meniscus.pdf

Jean François El Hajjar, Vincent Jolivet, Djamchid Ghazanfarpour, and Xavier Pueyo. 2009. A model for real-time on-surface flows. *Vis. Comput.* 25, 2 (2009), 87–100.

Ronald Fedkiw, Jos Stam, and Henrik Wann Jensen. 2001. Visual simulation of smoke. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques (SIGGRAPH '01)*. ACM, New York, NY, USA, 15–22.

Nick Foster and Dimitri Metaxas. 1996. Realistic animation of liquids. *Graph. Models Image Process.* 58, 5 (Sept. 1996), 471–483. https://doi.org/10.1006/gmip.1996.0039

R.A. Gingold and J.J. Monaghan. 1977. Smoothed particle hydrodynamics: theory and application to non-spherical stars. *Notices of the Royal Astronomical Society* 181 (1977), 375–389.

Simon Green. 2009. *Particle Simulation using CUDA*. Technical Report. NVIDIA.

Markus Ihmsen, Jens Cornelis, Barbara Solenthaler, Christopher Horvath, and Matthias Teschner. 2014a. Implicit incompressible SPH. *IEEE Trans. Vis. Comput. Graph.* 20, 3 (2014), 426–435. https://doi.org/10.1109/TVCG.2013.105

Markus Ihmsen, Jens Orthmann, Barbara Solenthaler, Andreas Kolb, and Matthias Teschner. 2014b. SPH Fluids in Computer Graphics. In *Eurographics 2014 - State Art Reports*, Sylvain Lefebvre and Michela Spagnuolo (Eds.). The Eurographics Association, 21–42. https://doi.org/10.2312/egst.20141034

Wenzel Jakob. 2010. Mitsuba renderer. (2010). http://www.mitsuba-renderer.org.

S. F. Kister. 1993. *Wettability*. Marcel Dekker, Chapter Hydrodynamics of wetting.

J.H. Lienhard IV and J.H. Lienhard V. 2016. *A Heat Transfer Textbook* (4th ed.). Phlogiston Press, Cambridge, MA. http://ahtt.mit.edu Version 2.05.

William E. Lorensen and Harvey E. Cline. 1987. Marching Cubes: A High Resolution 3D Surface Construction Algorithm. In *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '87)*. ACM, New York, NY, USA, 163–169.

Frank Losasso, Tamar Shinar, Andrew Selle, and Ronald Fedkiw. 2006. Multiple interacting liquids. *ACM Trans. Graph.* 25, 3 (2006), 812–819.

L. B. Lucy. 1977. A numerical approach to the testing of the fission hypothesis. *Astronomical Journal* 82 (1977), 1013–1024. https://doi.org/10.1086/112164

Miles Macklin and Matthias Müller. 2013. Position based fluids. *ACM Trans. Graph.* 32, 4 (jul 2013), 104:1–104:12.

Miles Macklin, Matthias Müller, Nuttapong Chentanez, and Tae-Yong Kim. 2014. Unified Particle Physics for Real-Time Applications. *Siggraph 2014* 33, 4 (2014), 1–12.

J. J. Monaghan. 2005. Smoothed particle hydrodynamics. *Reports on Progress in Physics* 68 (2005), 1703–1759.

D. Morgenroth, D. Weiskopf, and B. Eberhardt. 2016. Direct raytracing of a closed-form fluid meniscus. *Vis. Comput.* 32, 6-8 (jun 2016), 791–800.

Matthias Müller, David Charypar, and Markus Gross. 2003. Particle-based fluid simulation for interactive applications. In *Proc. Symp. Comp. Anim.* 154–159.

Matthias Müller, Barbara Solenthaler, Richard Keiser, and Markus Gross. 2005. Particle-based fluid-fluid interaction. In *Proc. Symp. Comp. Anim.* 237–244.

Jens Orthmann, Hendrik Hochstetter, Julian Bader, Serkan Bayraktar, and Andreas Kolb. 2013. Consistent Surface Model for SPH-based Fluid Transport. In *Proc. Symp. Comp. Anim.* 95–103.

Jens Orthmann and Andreas Kolb. 2012. Temporal Blending for Adaptive SPH. *Computer Graphics Forum* 31, 8 (2012), 2436–2449.

R R Rogers and M K Yau. 1996. *A short course in cloud physics* (3rd ed.). Butterworth-Heinemann.

Mirza Mohammed Shah. 2014. Methods for Calculation of Evaporation from Swimming Pools and Other Water Surfaces. *ASHRAE Transactions* 120, 2 (2014), 3–17.

Charles C. Smith, George Löf, and Randy Jones. 1994. Measurement and analysis of evaporation from an inactive outdoor swimming pool. *Sol. energy* 53, 1 (1994), 3–7.

Barbara Solenthaler and Renato Pajarola. 2009. Predictive-corrective incompressible SPH. *ACM Trans. Graph.* 28 (2009), 40:1–40:6.

Barbara Solenthaler, Jürg Schläfli, and Renato Pajarola. 2007. A unified particle model for fluid solid interactions: Research Articles. *Comput. Animat. Virtual Worlds* 18, 1 (2007), 69–82.

Jos Stam. 1999. Stable fluids. In *Proc. SIGGRAPH (SIGGRAPH '99)*. ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 121–128.

Sebastian-T Tillmann and Christian-A Bohn. 2015. Simulation of Water Condensation based on a Thermodynamic Approach. In *Vision, Model. Vis.*, Thomas Bommes, David Ritschel, and Tobias Schultz (Eds.). The Eurographics Association.

Huamin Wang, Gavin Miller, and Greg Turk. 2007. Solving General Shallow Wave Equations on Surfaces. In *Proc. 2007 ACM SIGGRAPH/Eurographics Symp. Comput. Animat.* Eurographics Association, 229—-238.

Huamin Wang, Peter J. Mucha, and Greg Turk. 2005. Water drops on surfaces. *ACM Trans. Graph.* 24, 3 (jul 2005), 921–929.

Rene Winchenbach, Hendrik Hochstetter, and Andreas Kolb. 2017. Infinite Continuous Adaptivity for Incompressible SPH. *ACM Transactions on Graphics* 36, 4 (jul 2017), 102:1–102:10.

Jihun Yu and Greg Turk. 2010. Reconstructing surfaces of particle-based fluids using anisotropic kernels. In *Proc. Symp. Comp. Anim.* 217–225.

Yongning Zhu and Robert Bridson. 2005. Animating sand as a fluid. *ACM Trans. Graph.* 24, 3 (2005), 965–972.