Computational Visual Media
https://doi.org/10.1007/s41095-0xx-xxxx-x

Review Article

Physics-based fluid simulation in Computer Graphics: Survey, research trends, and challenges

Xiaokun Wang[1,2,†], Yanrui Xu[1,3,†], Sinuo Liu[1,4], Bo Ren[5], Jiří Kosinka[3], Alexandru C. Telea[6], Jiamin Wang[1], Chongming Song[1], Jian Chang(✉)[2], Chenfeng Li[7], Jian Jun Zhang[2] and Xiaojuan Ban(✉)[1,8]

Abstract Physics-based fluid simulation has played an increasingly important role in the computer graphics community. Recent methods in this area have greatly improved the generation of complex visual effects and its computational efficiency. Novel techniques have emerged to deal with complex boundaries, multiphase fluids, gas-liquid interfaces, and fine details. The parallel use of machine learning, image processing, and fluid control technologies has brought many interesting and novel research perspectives. In this survey, we provide an introduction to theoretical concepts underpinning physics-based fluid simulation and their practical implementation, with the aim for it to serve as a guide for both newcomers and seasoned researchers to explore the field of physics-based fluid simulation, with a focus on developments in the last decade. Driven by the distribution of recent publications in the field, we structure our survey to cover physical background; discretization approaches; computational methods that address scalability; fluid interactions with other materials and interfaces; and methods for expressive aspects of surface detail and control. From a practical perspective, we give an overview of existing implementations available for the above methods.

Keywords Computer graphics, physical simulation, fluid simulation, fluid coupling

[1] Beijing Advanced Innovation Center for Materials Genome Engineering, and School of Intelligence Science and Technology, and Shunde Innovation School, University of Science and Technology Beijing, Beijing, China
[2] National Center for Computer Animation, Faculty of Media and Communication, Bournemouth University, Bournemouth, Dorset, United Kingdom
[3] Bernoulli Institute, University of Groningen, Groningen, Netherlands
[4] School of Computer Science, Peking University, Beijing, China
[5] College of Computer and Control Engineering, Nankai University, Tianjin, China
[6] Department of Information and Computing Sciences, Utrecht University, Utrecht, Netherlands
[7] College of Engineering, Swansea University, Swansea, United Kingdom
[8] Institute of Materials Intelligent Technology, Liaoning Academy of Materials, Shenyang, China
[†] These two authors contributed equally to this work
Corresponding Authors: jchang@bournemouth.ac.uk;banxj@ustb.edu.cn
Manuscript received: 2022-01-01; accepted: 2022-01-01

1 Introduction

Given their ubiquitous existence in natural environments, fluids are a crucial element in visual simulations. Their versatile motion and complex behavior make them an attractive but also difficult to describe and compute target for graphics simulations. As such, the simulation of fluids has long been one of the most important subjects in computer graphics. The development of computer technology has made it possible to simulate complex fluid phenomena directly using the governing equations of fluid dynamics. Many physics-based methods and techniques have been proposed to this end, ranging from simple but inaccurate models to progressively refined, complex techniques that capture increasingly challenging dynamics of the interacting media. The diversity of such methods, with widely varying assumptions, modeling power, and underlying implementation techniques, has made understanding the current state-of-the-art challenging, especially for practitioners and newcomers.

This survey aims to address the above understanding challenge by covering the major research topics of physics-based fluid simulation in computer graphics and new trends in those topics over the last decade. It discusses the different goals of fluid simulation, techniques proposed to address such goals, challenges of such techniques, and key findings in the field. We structure our survey in a top-down manner as follows: Section 2 presents our methodology used to collect relevant work from the literature (with a focus on the last decade) and proposes a classification of fluid simulation into seven main topics. Section 3 introduces the physical background and discretization approaches used by fluid simulation required to understand the remainder of the survey. Sections 4–10 discuss the papers found in each of the above-mentioned seven topics as follows. Section 4 presents the various classes of advanced adaptive, parallel, and data-driven computational approaches used to accelerate the implementation of the simulation models outlined in Sec. 3. The next three sections

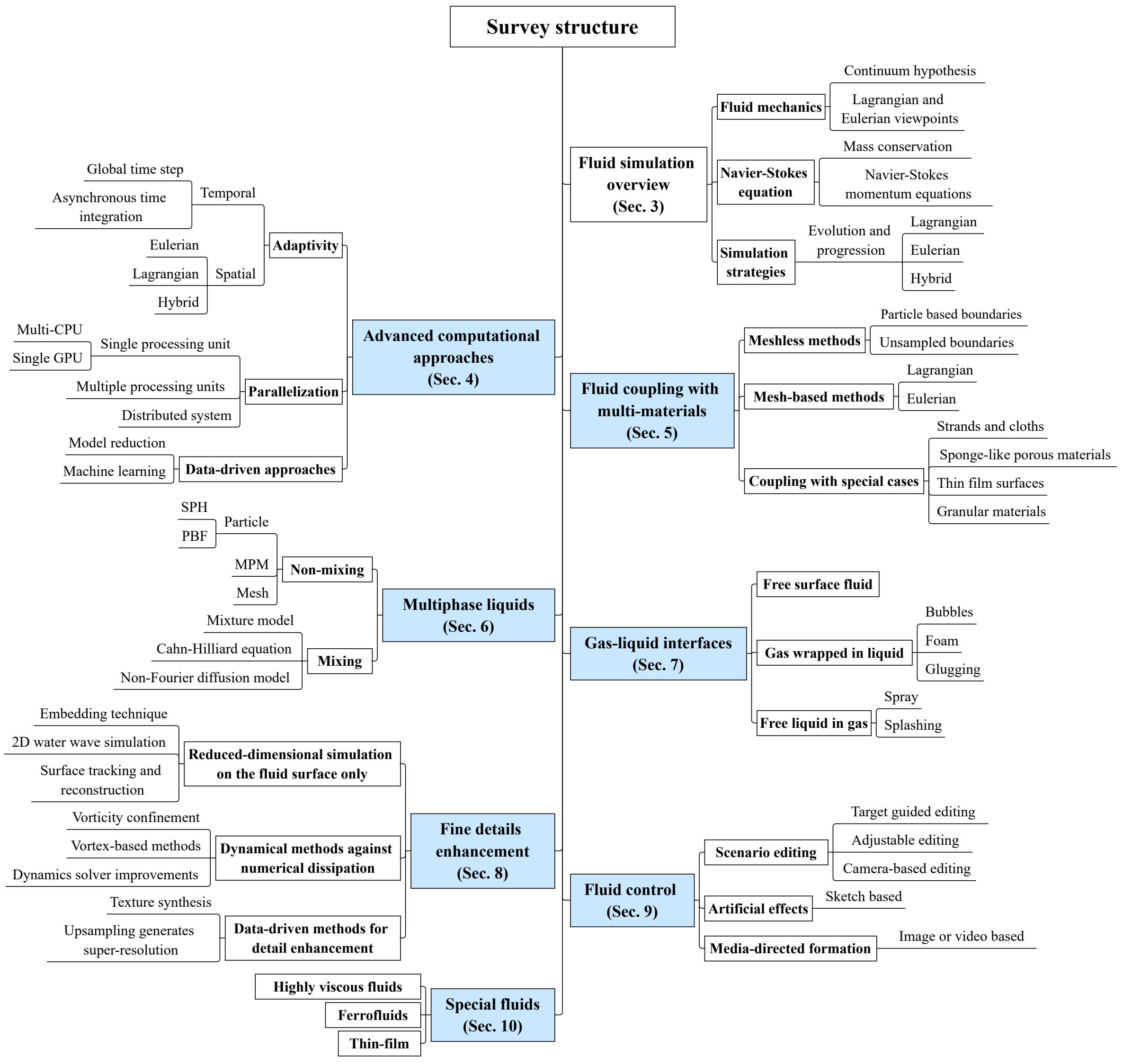


Fig. 1 Overview of the survey structure (see Sec. 2).

detail more specific and challenging simulation contexts: fluid interaction with different materials (Sec. 5), multiphase simulation (Sec. 6), and gas-liquid interfaces (Sec. 7). Sections 8 and 9 discuss artistic measures for improving the quality by controlling the appearance and motion, respectively, of the simulated fluids. Section 10 considers special fluids. Finally, Section 11 concludes our survey by identifying key directions for future research. Appendix A presents and discusses software implementations of fluid simulation covering numerical simulation, modeling, and rendering aspects.

2 Survey construction methodology

Physics-based fluid simulation is a research area that has been active for many decades, with input from a diverse range of fields, including engineering, physics, mathematics, and computer science. A fluid simulation survey covering the scope of *all* these fields would be too extensive for a single paper. Furthermore, we believe that the interests of typical researchers and practitioners in *computer graphics* focus on a subset of the above aspects, and we structure our survey accordingly as follows.

As main information sources, we selected articles published in *ACM Transactions on Graphics* (TOG), *IEEE Transactions on Visualization and Computer Graphics* (TVCG), and *Computer Graphics Forum* (CGF), which are arguably the three most influential and representative computer graphics journals. As our survey aims to cover recent tendencies, we included all relevant papers from these journals published in the last decade (2012–2022). We further included some papers presented at key graphics conferences like the *ACM SIGGRAPH / Eurographics Symposium on Computer Ani-*

mation (SCA), *ACM SIGGRAPH* (SIGGRAPH), and *ACM SIGGRAPH Asia* (SIGGRAPH Asia). Finally, we included earlier papers that have significantly impacted recent research. In total, we collected and further analyzed 327 papers meeting the above criteria.

The origins of physics-based fluid simulation in computer graphics can be traced back to the 1970s with the development of "particle systems" [1]. However, a fully-developed, reasonably stable physics-based system for fluid animation was not achieved until the end of the last century [2]. In the decade following, various simulation strategies continued to evolve, focusing on improving stability, accuracy, and efficiency in fluid simulations. We discuss early development progress in this regard in Sec. 3.3. Subsequently, research on specific fluid phenomena and the tailoring of simulated effects began to emerge and gain momentum.

As we entered the 2010s, which is the primary focus of this survey paper, the main research interests in fluid simulation shifted towards addressing *specific effects* that are challenging to achieve using conventional fluid simulation methods. Alongside this shift, advances in machine learning technologies have opened up new ways to integrate neural networks with simulation algorithms, pushing the boundaries of what can be accomplished in fluid simulations. As this survey aims to discuss the current advancements in this area comprehensively, we *classify* the collected papers into seven relevant topics that span the past decade based on our detailed analysis of these papers. We then selected a subset of representative papers within each topic and discuss these in greater detail.

Figure 1 shows the seven identified topics at the first level of a hierarchy depicting our survey's structure. Further levels refine these into sub-topics. The seven main topics are as follows:

- **Advanced computational approaches:** (Sec. 4) Methods that aim to make full use of powerful parallel computing resources for fluid simulation.
- **Fluid coupling with multi-materials:** (Sec. 5) Methods that model the interaction between fluid and solid objects of various shapes and textures.
- **Multiphase liquids:** (Sec. 6) Methods for the simulation of liquid–liquid interaction effects of various phases.
- **Gas–liquid interfaces:** (Sec. 7) Methods dealing with scenarios where forces on gas–liquid interfaces dominate the fluid motion.
- **Fine detail enhancement:** (Sec. 8) Methods that concentrate on preserving/enhancing fluid motion on a detailed level.
- **Fluid control:** (Sec. 9) Methods that allow visual designers to control the appearance and style of fluid simulations.
- **Special fluids:** (Sec. 10) Methods that simulate nonconventional fluids, *e.g.*, highly viscous/thin, sensitive to magnetic fields, or targeting materials that are, strictly speaking, not fluids but behave like fluids.

Our seven-topic classification aims first and foremost to identify salient *trends* in the past decade. As such, the emergence of these topics is based on a significant portion of the found papers that can be grouped within each topic; in other words, the topics reflect a "data-driven organization" of how research on fluids proceeded in the past decade. This is in contrast with other surveys that group works based on a *predefined, model-driven* taxonomy proposed by their authors.

It is insightful to analyze the distribution of topics in the found papers and how these topics evolve over time. Figure 2 shows the numbers of papers found for each of the seven identified topics. We see that while variations exist (some topics being more popular than others), each topic has a significant number of papers, with 4–8 top-tier papers per year on average, thereby supporting the claim that our identified topics are a good way to organize the research. Figure 3 refines the above insight, showing the number of articles per topic and per year. We see that while some topics show an increase in publications (*e.g.,* advanced computational approaches or red curve in Fig. 3), all identified topics have been "alive" over the past decade – another indication that they are suitable for the organization of our survey.

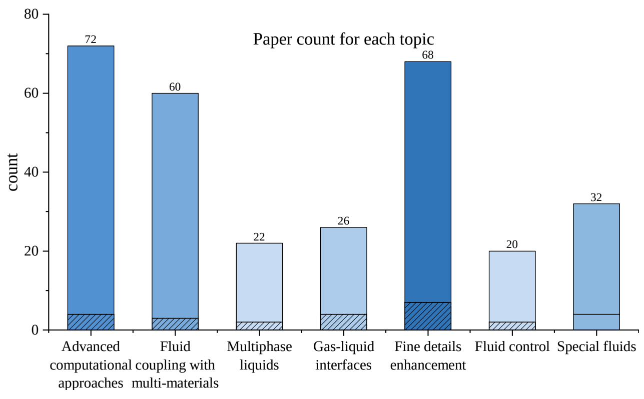


Fig. 2 Number of studied papers per identified topic. For each topic, the top (pure color) bar segment represents papers published in TOG, CGF, and TVCG. The bottom (shadowed) bar segment represents other key papers considered in our survey.

3 Fluid simulation overview

The development of fluid simulation in computer graphics is deeply rooted in the history of physics. In the 19th century,

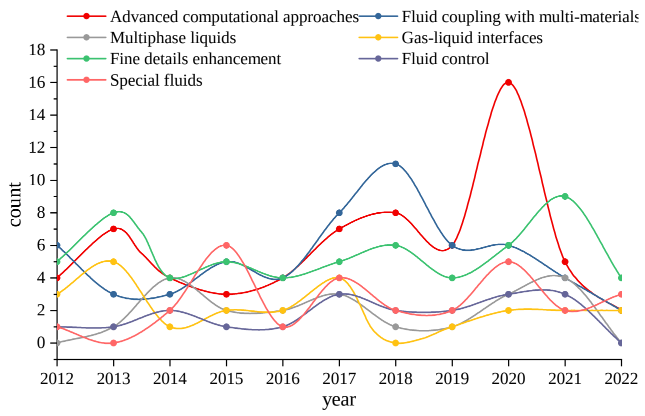


Fig. 3 Trends in number of papers published per topic over the past decade.

scientists such as Sir Isaac Newton and Claude-Louis Navier contributed significantly to the understanding of fluid mechanics, paving the way for the Navier–Stokes equations. These equations, which govern fluid motion, form the foundation of modern fluid simulation algorithms.

This section offers a basic introduction to fluid simulation and provides background knowledge for the remainder of the survey. For a more comprehensive understanding of fluid simulation, we refer to Bridson's book [3]. For more specific knowledge about Lagrangian-based smoothed particle hydrodynamics and material point methods, we refer to the surveys of Koschier *et al.* [4] and Jiang *et al.* [5], respectively. Readers less familiar with this field are highly encouraged to read this section before reading onwards. We first introduce relevant physical principles behind fluid simulation, such as the continuum hypothesis (Sec. 3.1) and Navier–Stokes equations (Sec. 3.2). We next present the early development of this area (Sec. 3.3), including a brief overview of the ideas behind different discretization strategies.

Table 1 summarizes the main notations used in our survey. We use these notations, accompanied by subscripts, superscripts, and parentheses, with parameters to denote these quantities under various conditions, as explained further in context.

3.1 Fluid mechanics

Matter in nature is built up of atoms and molecules that are discrete and separated by space. Simulating fluid at the microscopic level to describe macroscopic phenomena is only possible on supercomputers with weeks, if not months, of computing time. Computer graphics pursues a balance between efficiency and fidelity. For this, fluid mechanics based on a continuum hypothesis is the level at which physical properties are modeled.

Symbol	Description	Unit
p	*Pressure*	Pa
m	*Mass*	kg
ρ	*Density*	$kg \cdot m^{-3}$
μ	*Dynamic viscosity*	$Pa \cdot s$
γ	*Surface tension coefficient*	$N \cdot m^{-1}$
S	*Area*	m^2
V	*Volume*	m^3
c	*Fraction*	-
$\mathbf{f}$	*Force density*	$N \cdot m^{-3}$
$\mathbf{F}$	*Force*	N
$\mathbf{u}$	*Velocity*	$m \cdot s^{-1}$
$\mathbf{n}$	*Normal vector*	-
$\mathbf{x}$	*Position vector*	m
$\mathbf{g}$	*Gravitational acceleration*	$m \cdot s^{-2}$
ω	*Angular velocity*	$rad \cdot s^{-1}$
$\mathbf{T}$	*Stress tensor*	Pa

Table 1 Common notations (and their descriptions) used in this paper. For vectors and tensors, the "Unit" column shows the unit of their norm values.

Fluid mechanics models an object with matter continuously distributed over its body, an approximation called the *continuum hypothesis*. This means that any infinitely small volume element in the fluid is seen as a continuous medium, also called a *fluid parcel*. As Landau and Lifshitz stated [6], a fluid parcel is "very small compared with the volume of the body under consideration, but large compared with the distances between molecules."

In fluid mechanics, the *continuity equation* describes the transportation of physical properties in space and time as follows:

$$\frac{\partial A\left(\mathbf{x},t\right)}{\partial t}+\nabla\cdot\left(A\left(\mathbf{x},t\right)\mathbf{u}\left(\mathbf{x},t\right)\right)=s\left(\mathbf{x},t\right), \qquad (1)$$

where A can be an arbitrary scalar, vector, or tensor physical property, $\mathbf{u}$ is the velocity, and s is the source term for A, all described at time t and location $\mathbf{x}$. Equation (1) states that the change rate of any physical property at a fixed position $\partial A/\partial t$ depends on the variation brought by the flux of $A\mathbf{u}$ and source term s.

Lagrangian and Eulerian viewpoints. Considering the physical attribute A in Eqn. (1), a flow field can be analyzed from a Lagrangian or Eulerian viewpoint as follows.

The *Eulerian viewpoint* studies the physical field using fixed positions. The change rate of the physical value A at a given position $\mathbf{x}$ is the $\partial A\left(\mathbf{x},t\right)/\partial t$ term in Eqn. (1), which comes from both the flux and source terms. While intuitive, this does not explicitly express the motion of the fluid parcel in the continuum hypothesis, as parcels constantly travel through fixed locations at all times.

In contrast, the *Lagrangian viewpoint* studies the change rate of physical attributes with respect to the fluid parcel by recasting Eqn. (1) as

$$\underbrace{\frac{\partial A(\mathbf{x},t)}{\partial t} + (\mathbf{u}(\mathbf{x},t)\cdot\nabla) A(\mathbf{x},t)}_{\frac{DA(\mathbf{x},t)}{Dt}} + A(\mathbf{x},t)\nabla\cdot(\mathbf{u}(\mathbf{x},t)) = s(\mathbf{x},t) \tag{2}$$
$$\frac{DA_\iota(t)}{Dt} + A_\iota(t)\nabla\cdot\mathbf{u}_\iota(t) = s_\iota(t),$$

where $D(\cdot)/Dt$, the so-called *material derivative*, is the change rate of A within a fluid parcel. In Eqn. (2), $\mathbf{u}$ and s are the velocity and source term of a specific fluid parcel, respectively. Hence, all positions $\mathbf{x}$ can be substituted with the parcel identifier ι. For brevity, hereafter, we omit the explicit mention of $(\mathbf{x},t)$, (t), and ι unless required by the context.

3.2 Navier–Stokes equations

Numerous methods for calculating fluid motion have been developed, spanning from Lagrangian to Eulerian perspectives. However, the underlying physical principles for almost all of these approaches are rooted in the *Navier–Stokes equations*, which govern the dynamics of fluid flow and serve as a fundamental foundation for fluid simulations. Thus, we describe these briefly next.

Mass conservation. In a closed system, fluid mass is conserved over time. This principle is represented by the continuity equation (Eqn. (1)). Letting A be the fluid density and setting $s \equiv 0$, Eqn. (1) can be rewritten as

$$\frac{D\rho}{Dt} + \rho\nabla\cdot\mathbf{u} = 0. \tag{3}$$

For the case of *incompressible flow*, the density variation within the flow is conserved, i.e., $D\rho/Dt = 0$. This condition further implies a divergence-free velocity field, as expressed by

$$\nabla\cdot\mathbf{u} = 0. \tag{4}$$

Navier–Stokes momentum equation. To further describe the motion of incompressible fluid flow, one can analyze the momentum of each fluid parcel. By introducing a momentum term $\rho\mathbf{u}$ in Eqn. (1) and next using Eqn. (3), we obtain

$$\frac{\partial\rho\mathbf{u}}{\partial t} + \nabla\cdot(\rho\mathbf{u}\otimes\mathbf{u}) = \rho\frac{D\mathbf{u}}{Dt} = s_m, \tag{5}$$

where s_m is the momentum source altering the speed of each fluid parcel, and $\otimes$ represents the outer product operation. Following this, a basic form of the Navier–Stokes momentum equation for viscid compressible flow further specifies s_m into three separate terms as

$$\rho\frac{D\mathbf{u}}{Dt} = -\nabla p + \mu\nabla^2\mathbf{u} + \rho\mathbf{g}, \tag{6}$$

where p is pressure, $\mathbf{g}$ is gravitational acceleration, and μ is the dynamic viscosity coefficient describing how viscous a fluid is. Equation (6) states that the velocity change rate for a fluid parcel is affected by three force terms: pressure ($-\nabla p$), viscosity ($\mu\nabla^2\mathbf{u}$), and gravity ($\rho\mathbf{g}$).

3.3 Simulation strategies

3.3.1 *Early developments*

As computer technology advanced in the 20th century, numerical methods became popular for solving partial differential equations, including the Navier–Stokes equations. With the advent of powerful computer hardware and software, computer graphics began to incorporate these physics-based algorithms, enabling increasingly realistic fluid simulations.

Dating back to the 1970s, William T. Reeves, a member of Lucasfilm's Computer Division, Computer Graphics Group, pioneered the development of 'particle systems [1, 7]. These systems enabled the realistic depiction of elements such as smoke and fire in films, as seen in "Star Trek II: The Wrath of Khan." This breakthrough laid the foundation for early fluid simulation techniques in computer graphics. In the 1990s, physics-based fluid simulation began to gain traction. Wejchert and Haumann [8] used a simplified version of the Navier–Stokes equations to animate irrotational, incompressible linearized fluid flow, providing a physics-based foundation for their fluid animations. Subsequently, Stam and Fiume [9] incorporated the complete Navier–Stokes equations to create turbulent wind effects.

On the *Lagrangian* side, Desbrun and Cani [10] introduced Smoothed Particle Hydrodynamics (SPH) to the computer graphics field for simulating highly deformable bodies. On the *Eulerian* side, Foster and Metaxas [11] used the Navier–Stokes equations on fixed grids to simulate fluid motion. The study of fluid simulation reached a significant milestone at the end of the 20th century with Stam's Stable Fluids method [2]. This finally made stable, three-dimensional, physics-based fluid simulation an attainable goal, producing realistic fluid effects. It was the first unconditionally stable method for fluid simulation and introduced the concept of *semi-Lagrangian advection*, and it was one of the earliest works to apply the idea of *hybrid* simulation in the field.

Hybrid methods in fluid simulation merge the strengths of both Lagrangian and Eulerian approaches, yielding more versatile and robust systems. Two foundational principles that underpin hybrid fluid simulation are Harlow's [12] *particle-in-cell (PIC)* method and the refined *fluid implicit particle (FLIP)* method of Brackbill and Ruppel [13]. These techniques have contributed significantly to the widespread success and adoption of hybrid fluid simulation in the 21st century. The field also saw a major advancement when Zhu and

Bridson [14] applied the FLIP method to incompressible flow simulation. This moved hybrid fluid simulation to new heights as it enabled the exploration of complex fluid dynamics with enhanced precision and stability. The continuous evolution of hybrid fluid simulation techniques has had a profound impact on computer graphics, facilitating the creation of realistic and visually stunning effects.

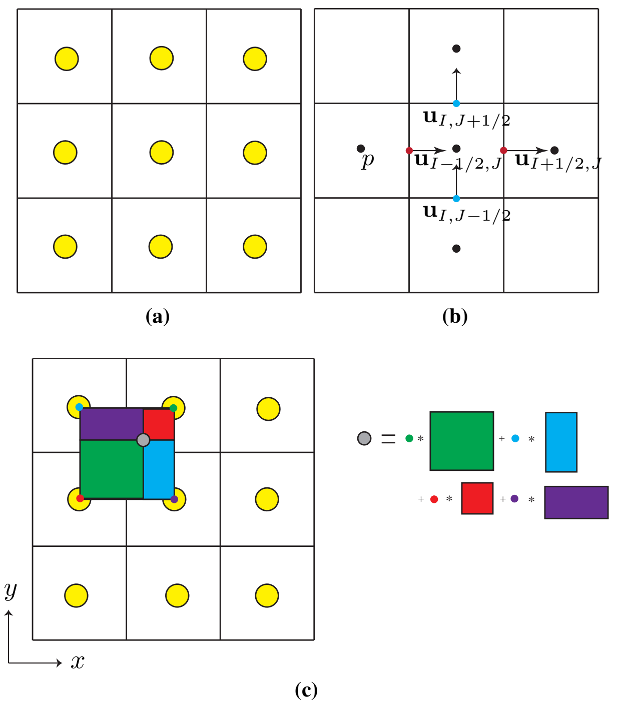


Fig. 4 Schematic diagram of Eulerian grids. (a) Collocated grid where physical quantities are stored in the cell centers (yellow points). (b) Staggered grid where different variables are stored at different locations; in this example, pressure is stored at the cell centers (black points), while velocity is split into its two Cartesian components and stored at the centers of the vertical cell edges (red and blue points). Subscripts I and J denote spatial indices. (c) Using bilinear interpolation to obtain the value of a physical quantity at any position.

3.3.2 *Discretization Strategies*

As fluid simulations in computer graphics have evolved since the early development in the 20th century, the field has branched out in three distinct directions: Eulerian, Lagrangian, and hybrid schemes. Each of these approaches offers unique advantages and challenges, contributing to the comprehensive understanding of fluid dynamics in computer graphics.

Eulerian schemes. These simulation methods use the Eulerian viewpoint introduced in Sec. 3.1, *i.e.*, compute property values at fixed points in the simulation domain. For this, the domain is typically divided into evenly-distributed cells. In a traditional *collocated grid* structure (Fig 4a), all physical values are evaluated at the center of each cell. To derive a continuous flow field with values at arbitrary positions, *e.g.*, the grey dot in Fig. 4c, one can use a weighted interpolation of neighboring cell values. The *staggered grid* (Fig. 4b) stores physical values at cell edges and centers separately. Compared with collocated grids, staggered grids are currently more popular for simulating incompressible fluids given their higher stability. It is noteworthy that staggered grids are related to the *marker-and-cell (MAC)* method [15], which was used in the early days of computational fluid dynamics to solve incompressible flow problems.

Lagrangian schemes. In the Lagrangian framework, domain discretization is based on a set of particles moving with the fluid flow, each approximating the physical values of a fluid parcel. Hence, Lagrangian schemes conserve mass by construction. Because particle locations can be much more freely distributed over the computational domain compared to covering the same domain with a grid, Lagrangian schemes are also good at modeling complex free surface details.

Currently, SPH is one of the most popular Lagrangian methods for fluid simulation, with origins in the works by Lucy [16] and Gingold and Monaghan [17]. SPH has evolved significantly over time, with various advancements and improvements.

Figure 5 shows how SPH performs interpolation, where the physical value A at location $\mathbf{x}_i$ of particle i is computed using a *smoothing kernel* W as follows:

$$A\left(\mathbf{x}_i\right) = \sum_j V_j A_j W\left(\mathbf{x}_i - \mathbf{x}_j, h\right), \tag{7}$$

where h is called the smoothing length, V is the volume of (the parcel of) each particle, and j indicates all particles closer to i than the distance h. To compute higher-order quantities, *e.g.*, pressure gradients, one can simply replace the kernel function W in Eqn. (7) with its higher-order counterpart (gradient in our example).

Initially, the weakly-compressible SPH (WCSPH) [18] approach was introduced, where pressure computation was performed explicitly. Later, the Predictive-Corrective Incompressible SPH (PCISPH) [19] method was proposed, which introduced a prediction-correction scheme for implicit pressure computation. This technique improved the stability and accuracy of fluid simulations by enforcing incompressibility more effectively. Further developments led to the introduction of Implicit Incompressible SPH (IISPH) [20], which provided a more strictly incompressible simulation with increased computational efficiency. Most recently, the Divergence-Free SPH

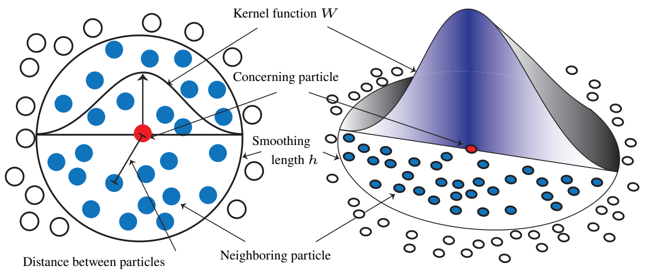


Fig. 5 Schematic diagram of Lagrangian-based smoothed particle hydrodynamics.

(DFSPH) [21] method has been developed, which further enforces the divergence-free condition within a simulation.

Position-based Dynamics (PBD) is a versatile and efficient simulation method for handling various physical phenomena, including fluids, deformable solids, and cloth. PBD was first introduced by Müller *et al.* [22] as an alternative to traditional force-based dynamics, focusing on the direct manipulation of object positions instead of computing forces and accelerations. In the context of fluid simulation, the *Position-based Fluids (PBF)* method was proposed by Macklin and Müller [23], building upon the principles of PBF and enforcing incompressibility by iteratively adjusting particle positions.

Hybrid schemes These schemes combine the advantages of Lagrangian and Eulerian schemes by representing the motion of the fluid flow with Lagrangian particles while computing dynamics (forces) on an Eulerian grid.

As Figure 6 shows, to combine particles and grids, physical values must be separately mapped from particles to grids (P2G) and from grids to particles (G2P) before and after the dynamic simulation. A so-called shape function, similar to the kernel function W for the SPH method, performs these mapping procedures.

In the original PIC [24], only the momentum term is transferred between P and G. The later proposed FLIP [13] transfers the differential of momentum to obtain better dynamic effects at the cost of stability. The *Material Point Method (MPM)* introduced by Sulsky *et al.* [25] is another extension of the original PIC. It adds a new dimension to fluid simulation by considering the deformation gradient information along with the momentum term, making it suitable for simulating a wide range of materials, including fluids, granular materials, and deformable solids.

Throughout the development of the PIC, FLIP, and MPM methods, these techniques have evolved and merged to form more advanced approaches. The *Affine Particle-In-Cell (APIC)* method [26] extends the MPM framework by incorporating affine velocity fields, which reduces numerical dissipation and offers improved stability compared to both PIC and FLIP. The *Polynomial Particle-In-Cell (PolyPIC)* [27] method takes the MPM framework one step further by incorporating higher-order polynomial velocity fields, building upon the advancements made by the APIC method. Finally, the *Moving Least Squares Material Point (MLS-MPM)* [28] method utilizes moving least squares for grid interpolation and differentiation in MPM simulations, further enhancing the accuracy and robustness of the approach.

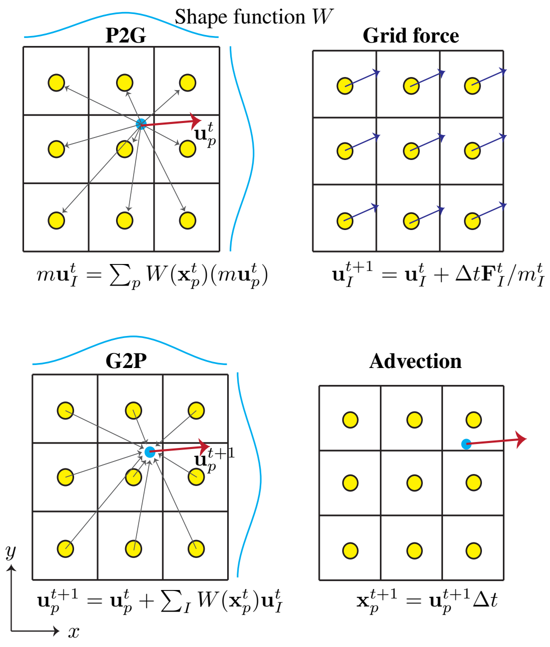

Shape function W
P2G
Grid force
$\mathbf{u}_p^t$
$m\mathbf{u}_I^t = \sum_p W(\mathbf{x}_p^t)(m\mathbf{u}_p^t)$
$\mathbf{u}_I^{t+1} = \mathbf{u}_I^t + \Delta t \mathbf{F}_I^t / m_I^t$
G2P
Advection
$\mathbf{u}_p^{t+1}$
y
x
$\mathbf{u}_p^{t+1} = \mathbf{u}_p^t + \sum_I W(\mathbf{x}_p^t)\mathbf{u}_I^t$
$\mathbf{x}_p^{t+1} = \mathbf{u}_p^{t+1}\Delta t$

Fig. 6 Hybrid scheme of the Particle-In-Cell method. This example uses a cell-centered grid, where information is stored at the yellow points. During simulation, momentum and weight (which can be used to obtain the velocity $\mathbf{u}_I^t$ on the grid) are transferred from particles to the cell centers. In the next step, forces are applied to grid nodes to compute the new velocity $\mathbf{u}_I^{t+1}$. Finally, the velocity is re-transferred from the grid to particles. When particles get their new velocities, the new positions can be easily found by forward Euler integration.

4 Advanced computational approaches

Fluid simulation requires a high discretization resolution to reach high visual quality. However, more discrete particles or denser grids demand more computing resources. This section surveys recent approaches for improving computational efficiency. We organize these into approaches that use adaptive time and/or space sampling (Sec. 4.1), GPU or CPU parallelization (Sec. 4.2), and the more recent data-driven approaches (Sec. 4.3). For a more extensive survey of this area, we refer to the work of Manteaux *et al.* [29].

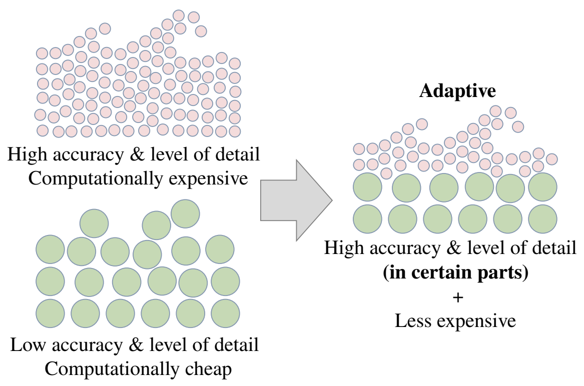

High accuracy & level of detail
Computationally expensive
Low accuracy & level of detail
Computationally cheap
Adaptive
High accuracy & level of detail
(in certain parts)
+
Less expensive

Fig. 7 Schematic diagram of particle-based adaptivity. Particle-based adaptivity adjusts particle size dynamically to reduce cost and preserve detail simultaneously.

4.1 Adaptive solutions

A stable, sufficiently accurate, and detailed simulation requires adequate temporal and spatial resolution. Time steps must be short enough to ensure stability, and high-resolution grids or dense particles are needed to capture fine details. However, computational cost increases with both spatial and temporal resolution, and an overall high resolution is not always needed. For example, time steps must be small for violent motion but can be longer when the overall movement is slow; high spatial resolution is needed to capture delicate splashes and sprays, but it is less important deep inside the fluid, where such detail is not visible.

As such, adaptivity uses high resolution only at necessary time and space instances and uses low resolution elsewhere to reduce computing costs. Figure 7 illustrates this strategy with particles as an example. Adaptive methods can be categorized into temporal and spatial adaptivity. *Temporal* adaptivity dynamically changes the time step, either globally or locally, for different parts of the fluid. *Spatial* adaptivity adjusts the resolution for different fluid regions or changes the method of discretization for a similar effect. These two approaches are described next.

4.1.1 Temporal adaptivity

Temporal adaptivity adjusts the time step length dynamically. A straightforward strategy is to adapt the time step globally, *i.e.*, use the same time step for the entire simulation domain. The time step size is determined dynamically at each time step under a restriction. For further performance gains, different time steps can be used for different spatial domain zones, thereby reducing the total number of integration steps needed.

Global time step The *Courant–Friedrichs–Lewy (CFL)* condition [31] is a well-known method for determining the time step size. Most current simulation methods compute a global time step according to the CFL condition at each time step. Generally, the CFL condition takes the form

$$C \equiv \frac{\|\mathbf{u}_c\|\Delta t}{\Delta x} \leqslant C_{\max}, \tag{8}$$

where $\|\mathbf{u}_c\|$ is the speed of information propagation, Δx is the grid-cell size for Eulerian and hybrid simulations or smoothing length for Lagrangian ones, $C_{\max}$ is a constant based on the size of discrete operators, and C is the CFL or Courant number. In practice, $\|\mathbf{u}_c\|$ typically represents the speed of sound in the material or maximum velocity in the simulation. The time step length Δt is usually chosen so that C is in the range of $[0, 1]$. The choice of the maximum Courant number $C_{\max}$ is generally dictated by the type of simulation algorithm being used, but it should not exceed 1. Methods such as PIC or MPM tend to offer greater flexibility in choosing $C_{\max}$ compared to SPH. Using the same method with an implicit time integration scheme allows for larger $C_{\max}$ values while maintaining simulation stability.

Determining an optimal value for $C_{\max}$ often involves an extensive trial-and-error process tailored to a specific scenario. Sun *et al.* [32] addressed this issue by considering metrics related to the stability of MPM simulations, such as the deformation gradient. By using these metrics, they were able to more effectively identify the performance limits and improve the overall stability of simulations.

Asynchronous time integration. When dealing with scenarios involving both intense waves and calm regions, implementing a global time step restriction can be inefficient and wasteful. To address this, the concept of *regional time stepping* was initially introduced to the SPH method by Goswami *et al.* [33]. This approach subdivides the simulation space into smaller regions, allowing each region to have its own independent time step. Recognizing the grid-based nature of the subdivided regions, Fang *et al.* [34] extended this idea to the MPM method. In their technique, a scheduler determines the order in which blocks are updated, while a buffer block is employed to handle boundaries between blocks with different time steps. This resulted in significant performance improvements, achieving speed-ups of $9.8\times$ compared to traditional synchronous MPM implementations. Inspired by Fang *et al.* [34], Koike *et al.* [30] proposed an asynchronous time integrator for Eulerian liquid simulation, with an interpolation strategy to deal with boundaries between different-time-step zones and an advection scheme to prevent seams at the boundaries. While the abovementioned methods effectively enable the time step to be asynchronized

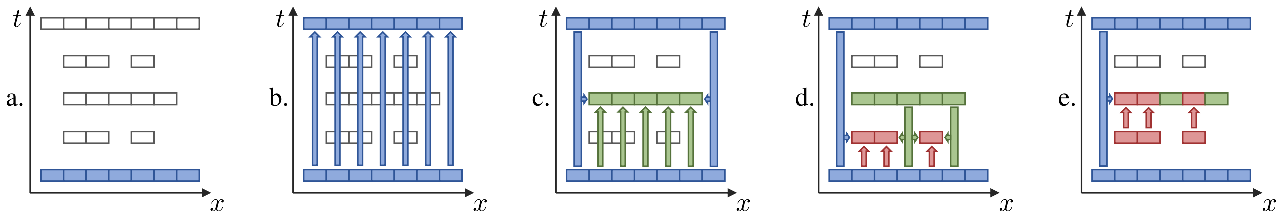
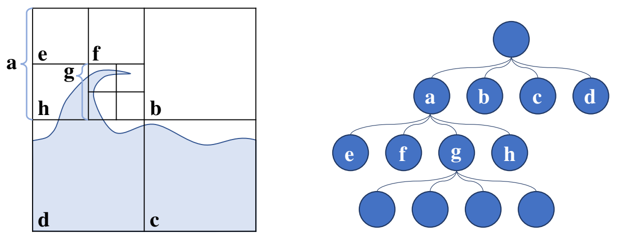


Fig. 8 Schematic diagram of the asynchronous time integration scheme in [30]. Boxes refer to discretized variables, such as velocity, and color represents the time step. A simulation (a) is first advanced with the largest time step (b). Next, smaller time steps are used (c, d). If values needed for computation are calculated using a larger time step, they are interpolated to match the current step size. Once the smallest time step is reached, it is used to overwrite previous results (e). This procedure is applied recursively to update all variables.

for separate regions, they still necessitate synchronization for all regions at simulation time barriers. Reinhardt *et al.* [35] presented a fully asynchronous time integration model for SPH fluid animation, where each particle has an individual time step and is processed using a priority queue.

4.1.2 Spatial adaptivity

These methods change the spatial resolution or discretization method in different spatial regions to keep fine detail in some regions but use coarser (and faster to compute) detail in less important regions. Spatial adaptivity methods are heavily dependent on the underlying discretization. We next detail different spatial adaptive approaches for the Eulerian, Lagrangian, and hybrid approaches.



(a) Example of an octree **(b)** Corresponding data structure

Fig. 9 Schematic diagram of an octree represented in 2D, where each cell has four children. 3D octrees have eight children per cell.

Eulerian grids. Grid-based Eulerian methods use adaptive grid structures to achieve dynamic spatial resolution. However, compared to uniformly distributed Eulerian grids, it is challenging to design a stencil on an adaptive grid for pressure solving to attain high-order accuracy and form a symmetric positive-definite linear system that can be efficiently solved on non-symmetric adaptive grids.

The *octree data structure* is one grid adaptivity approach that allows the resolution of axis-aligned structured grids to be changed. As shown in Fig. 9, each cell is divided into eight equal children by cutting it in half along each axis. Octrees have the advantage of regularity, supporting fast discretization, and simple implementation. However, on the transition between different grid levels, octrees have T-junctions, which cause challenging numerical issues.

Losasso *et al.* [36] proposed the first octree-based liquid solver using a set of symmetric differential operators, which enables the Poisson equation to be solved on unrestricted octree grids. In the octree, velocity is stored on cell faces, while pressure is stored at cell centers. The velocity divergence $\nabla \cdot \mathbf{u}$ at cell centers is computed considering all cell faces f as

$$\nabla \cdot \mathbf{u} = \frac{1}{V_c} \sum_f (\mathbf{u}_f \cdot \mathbf{n}_f) S_f, \tag{9}$$

where V_c is the cell volume, and $\mathbf{n}_f$, $\mathbf{u}_f$, and S_f are the outward-pointing normal, velocity, and area of face f, respectively. The pressure gradient on each face is computed from the pressure of the two adjacent cells using

$$\frac{\partial p}{\partial x} = \frac{p_2 - p_1}{(\Delta x_1 + \Delta x_2)/2}, \tag{10}$$

where Δx denotes the cell size, and subscripts 1 and 2 denote the adjacent cells to that face.

Dynamically adjusted octree grids also present challenges in terms of modifying and accessing data. Setaluri *et al.* [37] proposed a *sparse paged grid (SPGrid)* data structure that constructs the octree as a hierarchy of sparsely populated regular grids instead of a standard pointer-based tree. Goldade *et al.* [38] recognized the limitations of the first-order accuracy of the velocity field for octrees and applied a variational finite difference discretization method to it, enabling a more efficient viscous simulation. Ando and Batty [39] focused on using octree grids to enhance surface detail exclusively. This approach further reduces implementation complexity while retaining the benefits of octrees. While the particular attention to maintaining data order for efficient computation is advantageous, it also presents a challenge in system design. Shao *et al.* [40] noted this issue and identified an underutilized potential within the regular Cartesian grid structure.

They ingeniously integrated the single instruction, multiple data (SIMD) approach with a multigrid structure, aiming to streamline and minimize the required number of multiplications. Their method showed significant speed-ups of 2.0 to 14.6 $\times$ compared to contemporary adaptive octree solvers found in commercial software for large-scale simulations.

Several approaches have been inspired by and extended from the octree grid concept. These works aim to improve efficiency and accuracy in various ways. Ferstl *et al.* [41] proposed a hexahedral finite element discretization multigrid solver on adaptive octree grids. By specially treating boundary conditions on the free surface, they achieved second-order accuracy on the surface. Aanjaneya *et al.* [42] focused on enhancing pressure projection on octrees. They used a finite volume power diagram to accurately recover irregular embedded boundaries that cross grids, satisfying both second-order accurate and symmetric positive definite (SPD) conditions. Xiao *et al.* [43] introduced an adaptive staggered-tilted (AST) grid for conducting adaptive fluid simulations on a regular discretization. By adding a tilted grid to an octree structure, they avoided T-junctions and further improved the adaptivity of the simulation.

Some fluid simulation methods employ multiple grids with different resolutions or structures to simulate various parts of the fluid, later compositing these elements together. This approach contrasts with using a single adaptive grid for the entire fluid domain. Gao *et al.* [44] devised a technique that divides the domain into nested partitions with different resolutions, effectively handling multi-resolution fluid behavior. English *et al.* [45] used overlapping Cartesian grids with varying scales and rotations to represent the fluid domain, constructing a local Voronoi diagram for managing pressure projection near grid interfaces. Li *et al.* [46] introduced an adaptive relaxation method for kinetic approaches, enabling fluid sampling at arbitrary overlapping resolutions and providing efficient representation of fluid behavior across a wide range of scales.

While many adaptive methods that use single or multiple grid structures can disrupt the uniform data structure of the original Cartesian grid, some works have found a balance between maintaining uniformity and introducing adaptivity. Zhu *et al.* [47] used a uniform grid within a cubic region of interest, extending the grid into the far-field by stretching cells along an axis. This approach retains the benefits of a uniform grid while providing adaptivity in specific areas. Ibayashi *et al.* [48] proposed a technique for dynamically warping uniform grids, combining the advantages of both unstructured and structured grids.

Lagrangian methods. Particle-based Lagrangian approaches, such as SPH, achieve spatial adaptivity by defining a desired resolution for each particle with a sizing function. By adjusting particle sampling through local merging or splitting of particles (as shown in Fig. 7), these methods are able to dynamically change the resolution, offering more efficient and accurate simulations while focusing on areas of interest.

The early study of adaptive SPH can be traced back to the work of Adams *et al.* [49]. They introduced a sizing function based on geometric local feature size that allows computational resources to be focused on geometrically complex regions. However, adaptive particles yield density errors due to different resolution scales, which can lead to instabilities. To address this issue, Orthmann and Kolb [50] proposed a temporal blending technique to limit the rate of temporal resolution change, thereby significantly reducing the error. With the advent of more strictly incompressible implicit SPH approaches, the size difference between neighboring particles must be minimized to avoid instability. Winchenbach *et al.* [51] achieved this by forming a continuous transition of particle resolution by tuning the splitting and merging pattern and introducing mass redistribution between particles. A simplified version of temporal blending was also incorporated. In their method, splitting supports arbitrary $1 : n$ patterns. Merging uses an $(n + 1) : n$ pattern, where one particle is merged into the others. Mass redistribution divides the excessive mass m_{ex} of one particle i equally among n particles. The physical attributes A of the mass-receiving particles j are updated to A_j^* by

$$A_j^* = \frac{\frac{m_{\text{ex}}}{n} A_i + m_j A_j}{\frac{m_{\text{ex}}}{n} + m_j}. \tag{11}$$

Zhai *et al.* [52] took inspiration from this method to propose an adaptive scheme for Power Particles. Winchenbach *et al.* [53] proposed a semi-analytic boundary handling approach to solving the problem that particle-based boundary representation is difficult to couple with fluid particles with very different sizes. Recently, Winchenbach and Kolb [54] introduced optimized refinement for splitting patterns with a discretized objective function that models the error, thereby significantly improving stability. Neighbor search for adaptive particles also needs to be specifically optimized. A particle in an adaptive simulation can have a widely varying number of neighbors within a given distance h, which are costly to compute. To solve this, Winchenbach and Kolb [55] proposed constrained neighbor lists to determine the neighbors in a user-specified range. To further accelerate the neighbor search process, Winchenbach *et al.* [56] introduced a sparse data structure for efficient neighbor search and ray tracing for

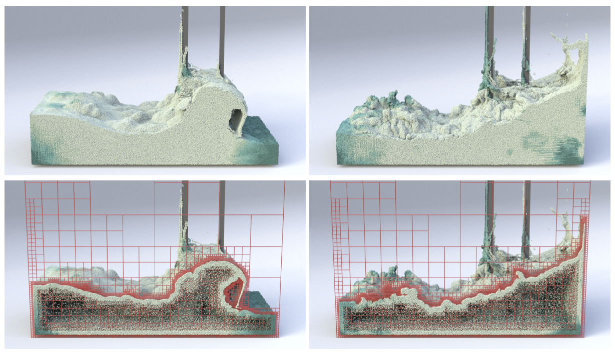


Fig. 10 Adaptive fluid simulation combining non-graded octrees and adaptively sampled particles [60].

adaptive SPH based on hash-maps.

Among vortex methods, ways to solve the Poisson problem efficiently with adaptive data structures have also been studied. The Poisson problem is an N-body problem, where the interaction between each object and the remaining objects is considered. Naively solving this problem requires $O(N^2)$ computations, so adaptive methods are used to reduce this complexity. The Fast Multipole Method (FMM) [57] uses an octree to approximately solve the N-body problem in $O(Nlog^{\eta}N)$, where $\eta \in \{0, 1\}$ by approximating interactions between far-away bodies by the body centers instead of computing all pairwise interactions. Zhang and Bridson [58] proposed a novel Particle–Particle Particle–Mesh method, which is easier to implement and parallelize on GPUs, and applied it to a vortex segment solver. Angelidis [59] used FMM with added support for non-uniform particle sampling to simulate incompressible smoke with vortices.

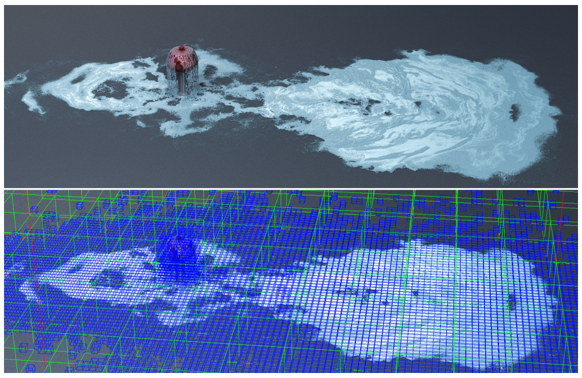


Fig. 11 A massive and sparse FLIP scene simulated using an efficient parallel GPU implementation [61], achieving one frame/second on an NVIDIA® Quadro GP100 GPU with 2 million particles and a $3360 \times 160 \times 2272$-cell grid.

Hybrid methods. Hybrid methods offer greater flexibility in implementing adaptive schemes due to their intrinsic combination of Lagrangian and Eulerian representations for fluid simulation. Ando *et al.* [63] applied the particle splitting–collapsing scheme, similar to traditional Lagrangian adaptive mechanisms, to adjust the granularity of fluid representation in relation to the distance to the fluid surface for FLIP. They used the finest particles to represent splashes and sheets. Ando *et al.* [64] later introduced an adaptive liquid solver on tetrahedral meshes, which combined a variant of FEM with FLIP advection, adapting the sizes of both the particles and tetrahedral meshes coherently for more efficient simulation. For highly stable situations, Yue *et al.* [65] explored the possibility of simulating the interior area as soft continuum materials to reduce computational costs at the solver level.

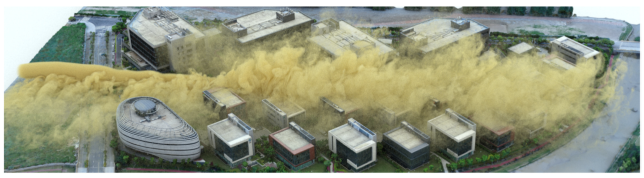


Fig. 12 GPU parallel computing accelerates complex large-scale high-resolution scene simulation [62].

To further save computational costs, more recent hybrid approaches aimed to “hollow” the inner area of the simulated fluid by using Eulerian simulation only, with particles applied near the surface. Chentanez *et al.* [66] proposed coupling pure Lagrangian and Eulerian methods to simulate single fluid bulks, addressing the issue of fluid representation transition and coupling stability between two different fluid solvers. However, coupling two different solvers can still be prone to instability, so Ferstl *et al.* [67] later returned to adaptive FLIP simulation using FLIP particles within a narrow band of the fluid surface. Nakanishi *et al.* [60] focused on constructing a proper data structure to adaptively merge and split octree grids for FLIP, adapting the size of the background grid only near the fluid surface (Fig. 10).

Sato *et al.* [68] took adaptivity to the extreme by extending Ferstl *et al.*’s work [67] to replace some of the fluid surface using level set surfaces via an introduced transition function. This idea represents a breakthrough in the field, enabling a seamless merging of physical simulation details and large-scale representations. To address the setback of inefficient, highly-dissipative wave propagation during the transition of surface representation, Huang *et al.* [69] employed hybridization of volumetric and surface-based advection-projection discretizations, with the Boundary Element Method (BEM) applied for long-lasting waves.

4.2 Parallelization

Parallelization of simulation algorithms offers a promising pathway to augment fluid simulation speed by capitalizing

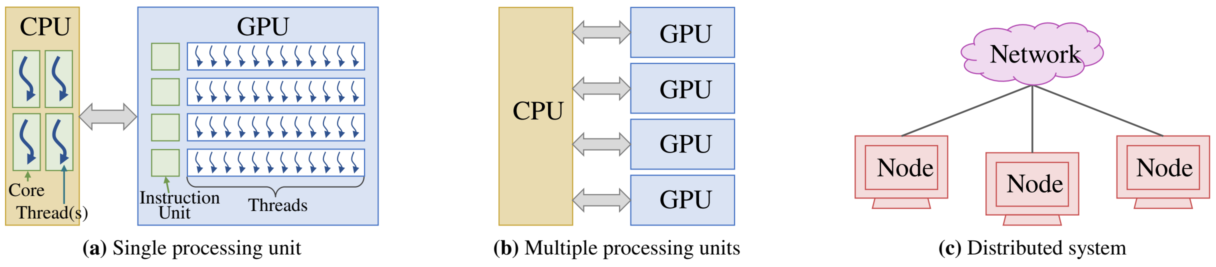


Fig. 13 Schematic diagrams of different parallelization types. A single processing unit can be a multicore-CPU or GPU. A CPU has few but high-processing-power cores. A GPU has weaker but many more cores, which can result in better performance for repetitive operations.

on the existing parallel computation capacities of modern GPUs and CPUs. We divide such methods into three classes: parallelization on a single processing unit (multi-core CPU or a GPU); parallel techniques with multiple processing units, especially multiple GPUs; and using distributed systems (see also Fig. 13).

4.2.1 Single processing unit

Parallelization can be most easily achieved on a single processing unit, such as a multicore CPU or GPU. Pure Eulerian and Lagrangian simulation techniques can be readily parallelized under these conditions. However, hybrid methods necessitate meticulous measures due to their composite nature of particles and grids. The Voxel Data Block (VDB) method pioneered by Museth *et al.* [70] offers a robust framework for manipulating sparse volumetric data. Wu *et al.* [61] successfully integrated this approach into the FLIP method by resolving the parallel particle-to-grid rasterization problem inherent in hybrid methods (Fig. 11). Gao *et al.* [71] accomplished a similar integration for MPM based on the SPGrid structure [37], wherein the particle-to-grid transfer mechanism was a crucial aspect to be solved. Furthermore, Chu *et al.* [72] applied and optimized the Schur complement theory for FLIP simulations. They divided the simulation domain into multiple subdomains with face edges and cross-points to establish a parallel-friendly data structure. Aiming to streamline the development of parallel programs, Hu *et al.* [73] proposed a new data-oriented programming language — Taichi. This language facilitates efficient authorship, access, and maintenance of sparse data structures and is accompanied by a compiler designed to automatically optimize and parallelize code on CPU or GPU platforms. Subsequently, Hu *et al.* [74] enhanced Taichi to include bit-level memory control over numerical data types.

4.2.2 Multiple processing units

Multi-GPU techniques are the approach of choice for scaling up the simulation size. In this case, multiple GPUs cooperate with the CPU(s), together forming a heterogeneous computing structure. The most challenging aspect of this structure is to reduce both the frequency and amount of data exchange between CPU–GPU and GPU–GPU, which is very expensive. Liu *et al.* [75] initially introduced the Schur complement method to the graphics community as a strategy to tackle this issue. As previously outlined, the Schur complement method has the significant benefit of segmenting the entire simulation domain into several regions. Each region can be efficiently computed using a single GPU, with the interaction boundary between these regions elegantly managed by the CPU(s). This setup eliminates the need for data transmission from a GPU to CPU and subsequently to another GPU. Meanwhile, Wang *et al.* [76] took a different approach by directly adapting the MPM algorithm structure to fit a multi-GPU framework. They accomplished this by developing a particle data structure to encourage coalesced memory access and circumvent atomic operations during particle-to-grid data writing. Additionally, they proposed a kernel fusion strategy to reduce the number of GPU kernel launches and global memory requirements. Chen *et al.* [62] further optimized kinetic methods for scenarios containing complex solids (Fig. 12). They introduced a multi-kernel launch methodology for parallelism enhancement and a parametric cost model to improve performance optimization. This exploration of fluid simulation parallelization demonstrates the richness and versatility of strategies within this domain.

4.2.3 Distributed systems

In the pursuit of scalability in fluid simulation, the potential of distributed platforms is harnessed to delegate tasks across multiple computing nodes. These distributed simulations frequently utilize automatic task allocation to ensure efficient processing. Biddiscombe *et al.* [77] laid a crucial groundwork by providing a GUI-based interface and analysis system for high-performance computing (HPC). Their innovative approach involved substituting the I/O layer in the Hierarchical Data Format Version 5 (HDF5) with a parallel data transfer driver, thereby enabling parallel simulation, analysis, and GUI operation concurrently on one or multiple devices. Mashayekhi *et al.* [78] developed a system that automatically distributes tasks across many multi-core cloud computing nodes to dynamically manage fluid partitions. Shah *et al.* [79] proposed a load-balancing scheme for sparse fluid simulations. Qu *et al.* [80] outlined a simple yet effective solution to accelerate distributed fluid simulations, which uses micro-partitioning to greatly improve load balance and communication performance.

4.3 Data-driven approaches

Although great progress has been made in space–time adaptive and parallel computing in recent years, the simulation of fluid by traditional physical methods still requires high computational resources. Strictly limited time steps are needed to ensure simulation stability when solving governing equations in a discretized space. Implementing efficient and accurate end-to-end fluid simulation pipelines is also technically highly challenging. *Data-driven* approaches provide an alternative solution for real-time interactive fluid simulation, which we outline next.

4.3.1 Model reduction

Model reduction is achieved by precomputing a set of simulation sequences to obtain a low-dimensional representation of fluid motion, allowing for efficient and fast re-simulation. Treuille *et al.* [81] first introduced model reduction to fluid simulation via Galerkin projection. They constructed vector field basis functions for fluid dynamics based on principal component analysis (PCA) to generate real-time fluids. For this, they computed the Galerkin projection of the following differential equation $\mathcal{F}$ onto a reduced-dimensional space:

$$\mathcal{F}(\mathbf{r}) = P \circ \mathcal{F}(\mathbf{v}) \circ P^{-1}. \tag{12}$$

The high-dimensional space vector $\mathbf{v} \in \mathbf{R}^n$ and low-dimensional space vector $\mathbf{r} \in \mathbf{R}^m$ are transformed by the projection operator $P(\mathbf{v}) = \mathbf{r}$ and its inverse $P^{-1}(\mathbf{r}) = \mathbf{v}$.

Later, improvements, such as the extension of the Galerkin projection to non-polynomial systems [82] and a multidimensional cubature method supporting semi-Lagrangian advection [83], were proposed. De Witt *et al.* [84] used Laplace eigenfunctions instead of PCA eigenvectors. The method performs a Galerkin projection of the vorticity form of the Navier–Stokes equations and is therefore not data-driven but physically driven. Liu *et al.* [85] further stabilized the method using a variational integrator, providing structure coefficients without artifacts. Zhai *et al.* [86] proposed a model reduction method based on empirical modal decomposition (EMD), which can decompose the flow field into various frequency components as the basis vectors for model reduction. This method can extract the characteristic parameters of the original fluid to achieve inverse modeling. To reduce the memory requirement to store basis functions, Cui *et al.* [87] generalized the dynamics to Neumann boundary conditions using analytic eigenfunctions and the Fast Fourier Transform (FFT). This approach allows the use of thousands of basis functions to produce more convincing and fine-grained fluid dynamics. Using this, they proposed an analytical extension of the Laplace eigenfunction method [88]. This spiral–spectral fluid simulation method is capable of producing realistic turbulent effects over a variety of radial domains, both surface and bulk. Mercier and Nowrouzezahrai [89] constructed an anisotropic vector field basis function that can accommodate curved boundaries and coupling with dynamic obstacles. The method sacrifices a physically accurate solution for a visually plausible simulation. A reduced model for fluids based on incompressible polynomial vector fields was proposed by Panuelos *et al.* [90] to reduce the computational cost of highly viscous fluids.

4.3.2 Machine learning

Machine learning methods have brought about a revolution in physics-based fluid simulation. In particular, deep learning techniques have proven the possibilities of data-driven approaches. Ladicky *et al.* [91] expressed physics-based fluid simulation as a regression problem and used a regression forest algorithm to approximate the dynamic behavior of fluid particles. This method strongly generalizes to simulating large-scale scenes in real time. Raveendran *et al.* [92] used interpolation on existing fluid simulations to rapidly generate a large number of new simulation results. Thuerey [93] improved this method using a signed distance function to fully automate the matching process. Recently, Oh and Lee [94] proposed a temporal interpolation network based on optical flow and forward advection that can derive high-frame-rate smoke simulations from low-frame-rate simulations.

With the development of deep learning, Convolutional Neural Networks (CNNs) [96] and Artificial Neural Networks (ANNs) [95] have been introduced to solve pressure calculations in fluid simulations and accelerate the pressure projection step (Fig. 14). Wiewel *et al.* [97] proposed an LSTM architecture to predict the evolution of fluids over time. They used CNNs to map the 3D fluid simulation to a low-dimensional latent space, greatly speeding up the simulation. They further improved the method using latent space subdivision [98], allowing for more stable predictions of complex long-term series. Takahashi and Lin [99] proposed a

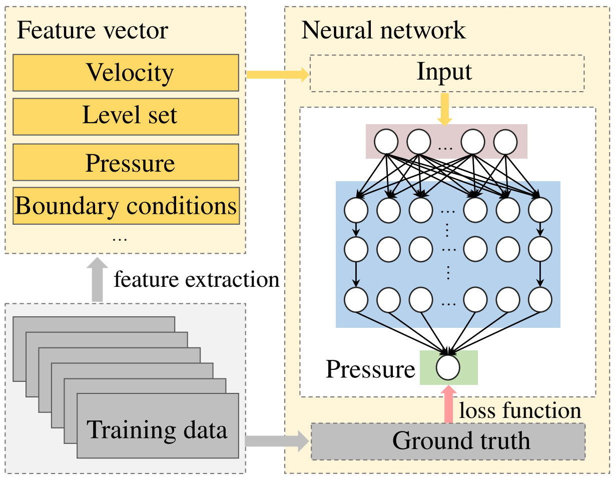


Fig. 14 Using Artificial Neural Networks (ANNs) to solve the pressure projection [95]. The network inputs feature vectors extracted from training data to output a pressure that is as close as possible to the ground-truth.

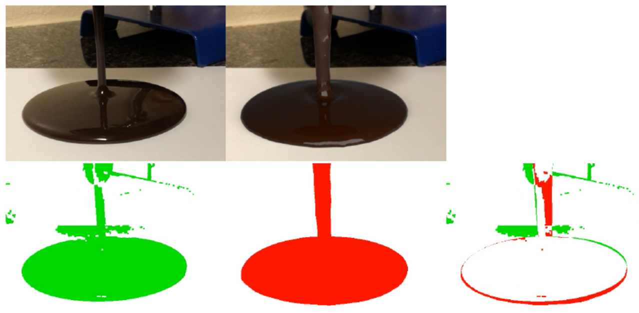


Fig. 15 Position information of the fluid extracted from example videos as a reference (left). 3D simulation projected onto the screen space (middle). The difference between the two is then iteratively reduced (right) [99].

framework capable of extracting physical parameters from real fluid videos and applying them to new scenarios to generate the user's ideal fluid behavior (Fig. 15). Eckert *et al.* [100] created ScalarFlow, the first large-scale volumetric dataset for real smoke reconstruction using computer graphics and machine learning. ScalarFlow also makes an important contribution by providing reliable benchmark data and evaluation criteria.

5 Multi-material fluid coupling

A key topic frequently mentioned in fluid simulation is the *coupling* of fluids with their environment, which is composed of different materials. Indeed, in computer graphics, fluids are mostly attractive due to the way they interact with their surroundings, as the attention of the spectator is arguably attracted by the interfaces, or boundaries, between fluids and the rest of the virtual world. Moreover, the behavior of fluid is strongly affected by how these surrounding factors themselves evolve. In this section, we explore this topic with a focus on recent works that aim to accurately and efficiently model coupling with multiple complex materials. We split the discussion into three separate subtopics: meshless methods (Sec. 5.1), mesh-based methods for handling fluid boundary conditions in the case of solid boundaries (Sec. 5.2), and solutions designed to model more complex couplings with multiple boundaries (Sec. 5.3).

5.1 Meshless Methods

Particle-based boundaries For most Lagrangian simulation approaches, the solid boundary sampled by so-called boundary particles is the main enabler of inter-particle interactions between fluid and other objects (Fig. 16). To couple fluid with solid boundaries, early methods used various approaches such as collision detection and ghost particles. Becker *et al.* [101] computed the contact point between the fluid and solid particles and controlled the normal and tangential velocities to impose boundary conditions. Yang *et al.* [102] facilitated the interaction between an SPH fluid and nonlinear FEM deformable solid by sampling proxy particles across the boundary, and they handled fluid–solid coupling with momentum-conserving collisions. Schechter and Bridson [103] used the ghost particle method to generate a thin layer of ghost particles in the nearby solid and air, reducing numerical errors caused by non-uniform particle distributions near boundaries. He *et al.* [104] generated staggered particles between neighboring particles to resolve the problem of shape functions losing the Kronecker delta property, thereby supporting various slip boundary conditions.

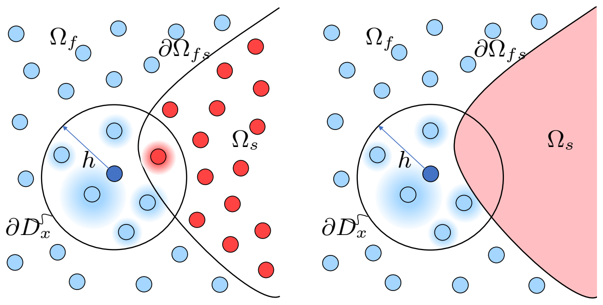


Fig. 16 Schematic diagram of fluid–solid coupling using meshless methods. Left: Discretization of the SPH approach in coupling with materials. Fluid and solid particles both contribute to the boundary handling. Ω denotes the domain, f denotes fluid, s denotes solid, and ∂D_x denotes the surrounding spherical neighborhood. Right: Solid domain in the neighborhood of a fluid particle representing the contribution of the boundary density value at the particle position.

To further reduce computation time and numerical errors, Akinci *et al.* [105] proposed a versatile and efficient method for SPH boundary handling without the need for collision detection or generating extra particles. The method resolved to directly handle the problem of uneven boundary sampling by evaluating a relative contribution of each boundary particle using Shepard interpolation (Fig. 17), given by

$$\rho_{f_i} = m_{f_i} \sum_j W_{ij} + \sum_k \Theta_{s_k}\left(\rho_{0_i}\right) W_{ik}, \tag{13}$$

$$\Theta_{s_i}\left(\rho_0\right) = \rho_0 V_{s_i} \tag{14}$$

to compute the correct density without depending on boundary sampling. The subscripts f and s represent fluid and solid particles, respectively, ρ_0 denotes the fluid rest density, and Θ_s computes the contributions of a boundary particle according to its volume. Compared to the volume of fluid particles with the same size V_f, the volume of solid particles V_s varies based on the local solid sampling distribution. Denser sampled regions have smaller solid volumes to maintain interaction stability. At the mere cost of a one-time evaluation of the above procedure, thin boundary geometries with only one layer of particles and non-manifold geometries can be supported.

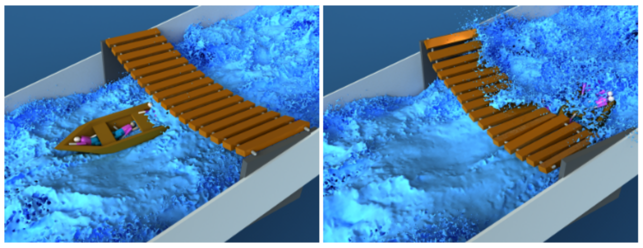


Fig. 17 Large-scale fluid–solid interaction [105]. A boat with ragdolls sails under a bridge (left). As the flow rate increases and the bridge is released, a second boat impacts the bridge.

The method of Akinci *et al.* [105] has gained widespread popularity due to its simplicity and efficiency. Macklin *et al.* [106] applied this method to unify various physical behaviors using a single-particle system for real-time applications built on the PBD. Cornelis *et al.* [107] used the concept of [105] to couple high-resolution FLIP with a low-resolution implicit SPH method. Peer *et al.* [108] built an implicit formulation for the simulation of incompressible linearly elastic solids embedded in the ISPH pressure solver, which further enables a pressure-based boundary treatment using the method of [105]. Takahashi *et al.* [109, 110] integrated the approach of [105] into their multilevel particle-based solver, in which they adaptively assigned various roles to the particles to guarantee the solvability of the linear system in a unified manner regardless of the arrangements of the particles.

Although Akinci *et al.* [105] eliminated a variety of artifacts of particle-based fluid–rigid coupling, numerical issues, such as penetration across the boundary, and the lack of higher-order accuracy of pressure due to the mirroring scheme of physical values from fluid to boundary particles, still limit the time step size and stability under drastic scenarios. Shao *et al.* [111] treated surface and inner boundary particles differently to prevent particle deficiency and penetration issues. Instead of using the state of density compression to denote the magnitude of the pressure force, Band *et al.* [112] introduced the notion of "volume compression" into coupling problems, where the rest volumes rather than the mass of boundary particles are applied to derive a continuous pressure force:

$$\mathbf{F}^{\mathrm{p}}_{f_i} = -V_{f_i} \sum_j V_j \left(p_{f_i} + p_j\right) \nabla W_{f_i j}, \tag{15}$$

which considers fluid samples f_i with all fluid and boundary neighbours j of the fluid sample f_i in the same way. The results of their experiments showed a significant improvement of stability and a wider range of possible time step sizes. Gissler *et al.* [113] resolved the stability issue in [105] by interlinking an artificial pressure solver for boundary particles with the fluid pressure, achieving a fully dynamic two-way coupling (Fig. 18). Truong *et al.* [114] prevented penetration by treating particle collisions with particle merging and splitting.

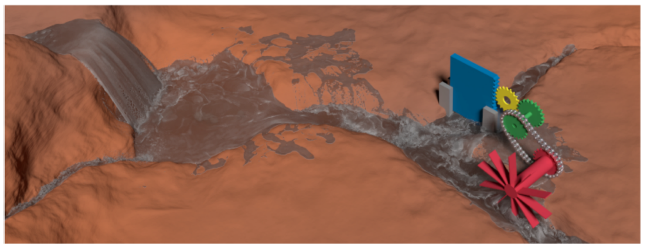


Fig. 18 SPH fluid (43.8 M particles) in a terrain (50 M static rigid particles) is two-way coupled with a water wheel that is connected to a gate via gears and a chain [113]. The gears, chain, and water gate are modeled with 2.3 M dynamic rigid particles. Up to 90 k simultaneous rigid–rigid contacts are handled.

Unsampled boundaries. The influence of a solid model on the surrounding fluid particles can also be handled carefully using a mesh-based (Eulerian) solid representation for a stable and effective coupling. Vines *et al.* [115] coupled Lagrangian vortex particles to mesh-based solids by generating vorticity at the solid boundary. Fujisawa and Miura [116] considered the influence of triangle mesh boundaries on the integration of a kernel function for SPH without the need for boundary particles. However, the method cannot handle solid boundaries with complex geometry and is computationally expensive compared to the method of [105]. Chang *et* *al.* [117] extended [116] to support arbitrarily shaped solid boundaries by converting the volume integral inside the solid boundary to a surface integral. Koschier and Bender [118] presented the "density maps" method, precomputing a continuous boundary density field to efficiently handle arbitrary boundary geometries. Bender *et al.* [119] also targeted the expensive renormalization process in [116] by storing the volume contribution from the boundary on a spatial grid that can be efficiently queried at runtime.

5.2 Mesh-based Methods

Another mainstream approach for fluid simulation is to use Eulerian and Lagrangian meshes, as shown in Fig. 19. However, loss of mass and difficulties in handling drastic deformations often limit the practicability of these approaches. We organize mesh-based methods into several classes as follows.

Lagrangian/Eulerian meshes Early on, Clausen *et al.* [120] used fully Lagrangian tetrahedral meshes to significantly reduce numerical viscosity in simulations with relatively low resolutions and long time steps. Azevedo and Oliveira [121] proposed a semi-Lagrangian method that introduced curvilinear grids and achieved more accurate boundary conditions in simulations with moderate resolutions. Besides these Lagrangian-meshed methods, Teng *et al.* [122] later incorporated previous work into an Eulerian fluid solver and resolved complex contact scenarios between multiple solids and fluids. Recently, Takahashi and Lin [123] formulated implicit viscosity integration as a minimization problem in which the volume fractions are consistently evaluated to handle sub-grid details.

Focusing on the poor boundary conditions for irregular boundaries defined under coarse grids, the *cut-cell* approach

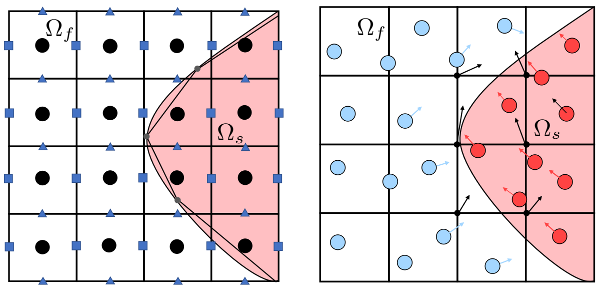


Fig. 19 Schematic diagram of fluid–solid coupling using mesh-based methods. Left: A sample MAC grid used in the fluid–solid coupling. Grey dots denote nodes of the solid regions. Fluid pressure values are stored at cell centers. Fluid velocity components are stored on cell faces. Right: A coupled solid–fluid system with the MPM method, which allows the system with particles to be discretized on a Cartesian grid and the pressure to be updated based on DOFs on the grid.

became a major trend to improve convergence for Neumann boundary conditions. Fluid grid cells are clipped against the solid boundary represented by a triangle mesh, forming several distinct polyhedral sub-cells at each time step, allowing small details to be handled without refining or rotating the grid. Based on the multigrid scheme proposed by Chentanez and Mueller-Fischer [124] with a variational discretization compatible on all levels, Weber *et al.* [125] presented a cut-cell-based multigrid scheme on staggered grids that is second-order accurate for Neumann boundaries. To better capture the flow across thin solids and gaps, Azevedo *et al.* [126] further proposed a topology-preserving pressure projection scheme on cut-cell meshes. Following this, Zarifi and Batty [127] used cut-cell discretization for two-way fluid-deformable interaction, enforcing the free-slip boundary condition at the actual interface. Their method computed the pressure based on the MAC grid of three dimensions x,y, and z as

$$\int_{\partial\Omega_f} \nabla p d\mathbf{n} \approx \sum_{\beta\in\{x,y,z\}} \left(S\left(\Omega_\beta\right) \frac{\partial p}{\partial \beta}\left(\Omega_\beta\right) + S\left(\Omega_{-\beta}\right) \frac{\partial p}{\partial \beta}\left(\Omega_{-\beta}\right) \right) \tag{16}$$

where Ω contributes to the domain region, and $S(\Omega)$ is the area of the domain region. The cut-cell was further extended by Chen *et al.* [128] to represent sub-grid structures on the free surface of the liquid, with a new iso-surface Poisson solver with desirable properties, such as second-order accuracy and symmetric positive definiteness. Tao *et al.* [129] introduced an algorithm based on VEM for simulating fluid flow on cut-cell meshes, which effectively handles complex geometries and accurately captures intricate features, including thin tubes and extremely thin walls.

Material Point Method Since its introduction to computer graphics, the Material Point Method (MPM) has garnered significant attention. By integrating features of Lagrangian particle representation and Eulerian grid representation, MPM offers a powerful technique for coupling fluid and solid simulations. However, conventional MPM solvers have drawbacks, such as computational inefficiency and limited capability to handle self-contact collisions, despite their physical realism and geometric convenience. To improve upon these, Gao *et al.* [130] presented an adaptive Generalized Interpolation Material Point (GIMP) method with extensively optimized particle–grid transfer memory efficiency and parallelism. Hu *et al.* [28] proposed Compatible Particle-In-Cell (CPIC), which enables the handling of discontinuous material points and infinitely thin boundaries by leveraging the relative positions of grid nodes and particles. They also embedded the Moving Least Squares (MLS) method into MPM to double the computation speed. However, such MPM approaches do not address the inconsistent tangential velocities at the interface between multiple materials, leading to visually unpleasant artificial stickiness. To alleviate this, Fang *et al.* [131] presented a ghost matrix operator-splitting scheme for monolithic coupling between incompressible fluids and elastic solids and designed a novel interface quadrature cut-cell MPM formulation for free-slip boundary conditions. Subsequently, Cao *et al.* [132] extended some ideas of [131] from incompressible to compressible flow.

Monolithic schemes Monolithic solvers simulate various materials and their interactions within a unified system that includes boundary conditions. These schemes naturally ensure a more robust interface of large density ratios and enable large time steps. They not only occur in SPH methods [113] but also in mesh-based (*e.g.*, MPM) methods [131, 132]. Aanjaneya [133] proposed a monolithic solver for efficiently simulating the interaction between rigid bodies and incompressible fluids. The solver remains robust even in poorly conditioned scenarios with large density ratios between the solid objects and fluid. Lai *et al.* [134] introduced a V-cycle of the Full Approximation Scheme (FAS) multigrid method to solve the linear complementary problems to achieve better scalability and efficiency compared to previous methods. Takahashi and Batty [135] proposed a monolithic pressure-viscosity-contact solver to simulate the complex interactions between rigid bodies and liquids, efficiently managing incompressibility and offering the option for implicit viscosity integration in liquids. The method also addresses contact resolution for rigid bodies and handles mutual coupling.

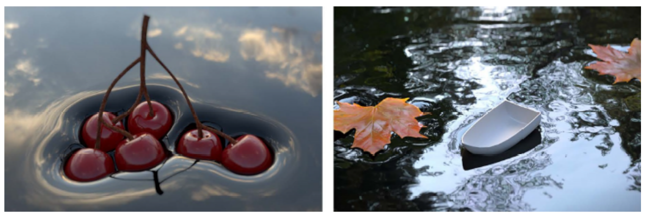


Fig. 20 Three-way coupling method to simulate surface-tension-dominant solid–liquid contact [136].

Partitioned schemes Compared to the monolithic scheme, partitioned schemes can deal more flexibly with multiple co-existing solvers by alternating between the solid and fluid while applying suitable boundary conditions. Akbay *et al.* [137] employed fluid and solid solvers as independent components with restricted interfaces, promoting modularity and facilitating code reusability. Lee *et al.* [138] used a partitioned methodology to connect a coarse-background Eulerian grid with a fine ALE mesh in their simulation framework for character– and hair–water interactions.

Recently, several coupling approaches for the special treatment of fluid have emerged. Brandt *et al.* [139] modeled fluids and deformable objects as incompressible media, avoiding expensive operations such as interface tracking and boundary condition handling. Ruan *et al.* [136] used a three-way coupling method, employing a thin liquid membrane to model contact between solid objects and fluid driven by strong surface tension (Fig. 20).

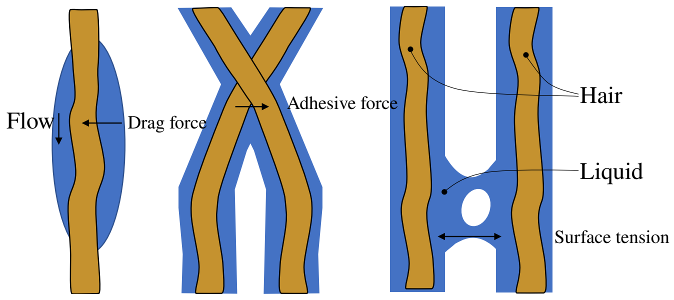


Fig. 21 Schematic diagram of wet hairs. Left: Wet hairs with a thin liquid layer flowing over their surface; their motion is influenced by the surrounding liquid flow. Middle: Proximity and collision between wet hairs cause adhesive and contact forces between hairs. Right: Cohesion of wet hair. Surface tension creates liquid bridges between closely positioned wet hairs.

5.3 Coupling with complex boundaries

Besides interacting with the dynamic or static boundaries of surroundings composed of rigid or elastoplastic materials, fluids often interact with other solid matter with complex and diverse physical attributes. Such cases require intricate boundary models and corresponding solvers. Next, we discuss recent advances in fluid coupling with thin film surfaces and/or porous materials, including hair, cloth, sponges, and sand.

Strands and cloths Coupling between hairs and fluid is complex due to the wetting of hairs. Such phenomena are shown in Fig. 21. Rungjiratananon *et al.* [140] used an Eulerian approach to capture hair porosity and wetting effects and a Lagrangian approach to simulate individual hair strands and their interactions, resulting in a detailed and dynamic hair simulation. Chen *et al.* [141] proposed a real-time painting system that aimed to generate realistic paintings by simulating the interactions among the brush bristles, paint, and canvas. Fei *et al.* [142] proposed a multi-component framework to model wet hair. PIC and Kirchhoff Rods were applied to model fluid and hair separately. A height-field was introduced to represent the liquid volume around each hair strand, considering the wet condition. Fei *et al.* [143] next extended the fluid attributes to compressible, shear-dependent liquids. A modified second-order Coulomb cone model was also designed to capture cohesion and friction during strand collisions. Lee *et al.* [144] used a tetrahedral volume mesh to embed hair, enabling the hairs to adhere to their embedded positions and facilitating simulations with millions of hairs during water–hair interactions.

For the interaction between cloths and fluid, Huber *et al.* [145] proposed an efficient method for two-way interaction between particle-based fluid and thin triangular meshes, enabling cloth–fluid coupling even under large time steps. Jiang *et al.* [146] created an anisotropic hyperelastic model that distinguishes the response to manifold strain, shearing, and compression in orthogonal directions. This model facilitates the coupling of various materials, such as elastic surfaces, curves, fluids, and granular materials. Fei *et al.* [147] introduced a method for simulating the intricate dynamics of woven or knitted fabrics, both partially and fully saturated, interacting with liquids, using the method of Jiang *et al.* [146] to deal with contact and collisions. To simulate stain formation and evolution on cloths, Wang *et al.* [148] developed a pigmented solution by utilizing a homogenization process that combines inhomogeneous and/or anisotropic properties with bulk anisotropic diffusion tensors. Zheng *et al.* [149] formalized the spreading of stains in woven fabric as in-yarn diffusion and cross-yarn diffusion and introduced a triple-layer model to manage wetting and wicking calculations.

Sponge-like porous materials Patkar and Chaudhuri [150] simulated liquid flow within a porous object with a deforming unstructured mesh and modeled liquid diffusion based on saturation, as well as allowing the liquid to be absorbed by, or leave, the solid.

Thin film surfaces. Real-life situations often involve surface flow phenomena, such as rainwater cascading down a tree trunk or the gradual progression of a water front in a shower room. For such flows, Vantzos *et al.* [151] proposed a triangle mesh model for simulating the motion of a thin viscous fluid film on a curved surface. The model includes discretization for curvature and advection operators to ensure accurate simulation results. Ren *et al.* [152] expanded the standard shallow-water flow model to accommodate general triangle meshes. They introduced a feature-based bottom friction model, allowing non-viscous flow motion to be captured along edges and creases on detailed 3D meshes.

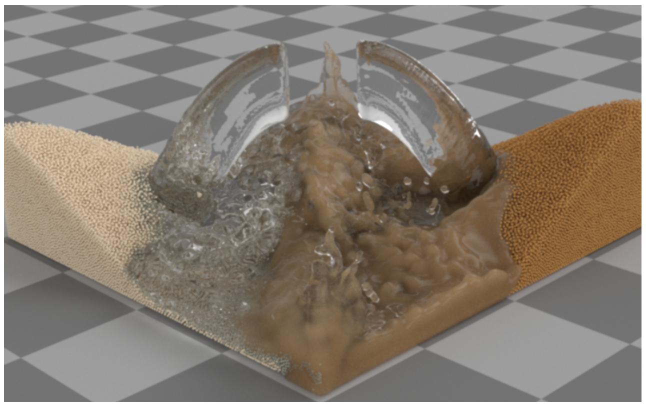


Fig. 22 Soluble and insoluble materials show different phenomena when coupled with fluid. Soluble and wettable granular materials [153].

Granular materials Such materials, *e.g.*, sand, can themselves exhibit flow-like behavior. In addition to their porous behavior, simulating dissolution of granular materials has also attracted much attention. Yan *et al.* [154] combined a hypoplastic model with SPH to simulate granular materials diffusing into fluid. Yang *et al.* [153] integrated the phase-field method to simulate liquids and multiple types of solids, with dissolution achieved by evolving the granular particle's concentration and phase (Fig. 22). Tampubolon *et al.* [155] used continuum mixture theory to simulate water–sand coupling in MPM, where different phases are coupled by momentum exchange. Gao *et al.* [156] modeled the motion of solid sediment particles inside fluids with MPM. Sediment was modeled by Drucker–Prager elastoplasticity, achieving two-way coupling between sediment and fluid. He *et al.* [157] proposed position-based constraints for granular flows, using cohesion and friction models that vary across space, with cohesion affected by water saturation. Takahashi and Batty [158] simulated two-way coupling between rigid bodies and continuum granular materials or liquids with a monolithic solver that combines pressure, friction, and contact interactions. Gao *et al.* [159] used a hybrid scheme to accurately simulate the behavior of discontinuous fluid-like substances. This approach integrated an affine particle-in-cell solver with density fields, enabling transformations across granular particles, dust clouds, powders, and their mixtures within a unified framework.

6 Multiphase liquids

The real world is replete with complex fluid phenomena, including dissolution, dispersion, and Rayleigh-Taylor instabilities, all of which are closely related to multiphase environments. The study of multiphase fluid simulation, a distinct subject within Computational Fluid Dynamics (CFD), has garnered significant attention in the realm of computer graphics. We next give an overview of this topic, focusing on liquid–liquid interactions. We group various phenomena and their corresponding simulations into non-mixing (Sec. 6.1) and mixing fluids (Sec. 6.2) based on whether a clear immiscible interface can be formed between two different phases. For non-mixing fluids, we classify their simulations according to the discretization methods used. We also survey recent methods by grouping them into two categories: mixture models where phase velocities are separately calculated and non-trivial-diffusion based models where phase velocities are considered equally.

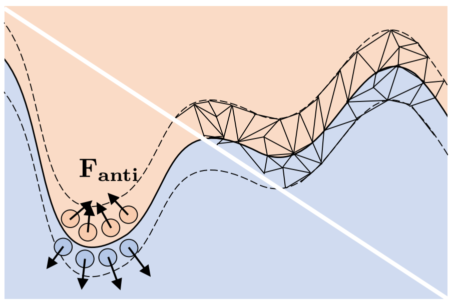


Fig. 23 Schematic diagram of non-mixing fluids. The lower-left part shows the anti-penetration force ($\mathbf{F}_{\mathbf{anti}}$), which is applied to particles in opposite directions along the surface normal. The embedding 3D fluid mesh can be re-tessellated to accommodate vertex displacement and produce changes in the interface topology.

6.1 Non-mixing fluids

Recent methods for simulating non-mixing fluids include particle-based methods, *e.g.*, SPH [160], PBF [161], and MPM [162], and mesh-based methods [163–166]. Phases in non-mixing fluids typically do not merge together. Non-mixing fluid phases inherently resist fusion, creating challenges in (a) tracking the interfaces between different phases and (b) accurately calculating force interactions between phases at liquid–liquid or liquid–solid interfaces. The unique properties of gas–liquid interfaces are discussed separately in Sec. 7.

SPH simulations of multiple fluids, particularly those with high density ratios, can produce erroneous interface tension and non-physical separation between the fluids. In response to this issue, Solenthaler and Pajarola [160] modified the standard SPH equations to account for the density discontinuity across the interface. They used an SPH density interpolation formulation, $\rho_i = m_i \sum_j W(\mathbf{x}_i - \mathbf{x}_j, h)$, instead of the standard one, $\rho_i = \sum_j m_j W(\mathbf{x}_i - \mathbf{x}_j, h)$.

Alduán *et al.* [161] proposed a versatile simulation framework based on PBF methods designed to address VFX production demands. On the phase interface, they used an SPH density interpolation formulation, which is similar to the method in [160]. Subsequent PBF calculations were also modified using this uniform-density formulation. High viscosity was handled by breaking the XSPH calculation into multiple lower-viscosity stable iterations. The surface tension effect was bounded for stable artistic controls.

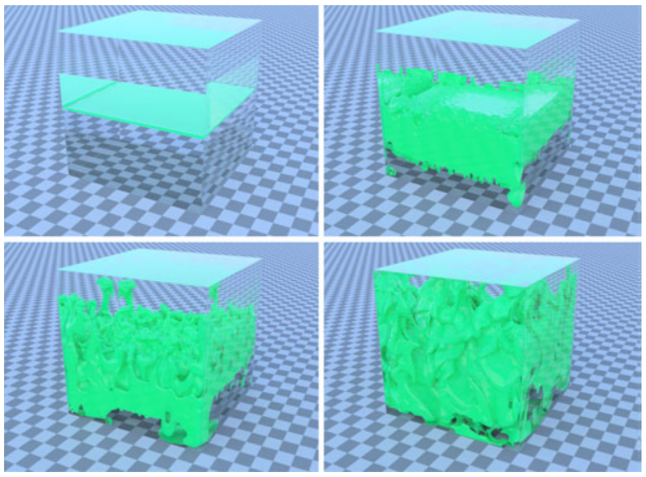


Fig. 24 Rayleigh–Taylor instability appearing at the interface between a high-density medium atop a low-density medium due to gravity [165].

Yan *et al.* [162] extended the MPM method to cope with interacting solid–fluid simulations. Upon the detection of a solid–fluid collision along the phase boundary, an anti-penetration force is applied to both the fluid and solid particles in opposite directions along the surface normal. The same strategy was used to simulate non-mixing fluids, where the anti-penetration force is calculated between each distinct phase and all other phases collectively (Fig. 23).

Mesh-based methodologies offer the advantage of explicit interface tracking using high-resolution geometric structures. Da *et al.* [164] considered a special emphasis on topological change handling in 2D surface tracking for non-mixing fluid simulations (Fig. 23). They constructed highly distorted interfaces featuring thin sheets and tiny regions. Meanwhile, Misztal *et al.* [163] aimed to prevent mismatches between phase occupancy regions and the simulation quantity storage grid; they achieved this by discretizing each phase region using unstructured 3D tetrahedral grids and tracking the deformation of the tetrahedra. They managed topology changes and mesh quality enhancements using a modified 3D deformable simplicial complex method. In contrast, Li *et al.* [165] avoided complex remeshing operations by combining mesh-based tracking and surface reconstruction from distance fields (Fig. 24). They reconstructed the surface meshes between each phase using an unsigned distance function and indicator function. The mesh was stored as later interpolation reference in a semi-Lagrangian update of the two functions in the next time step. This approach was extended in [166] for surface tracking of more than three phases. In this method, special care is taken to handle mesh penetrations and ensure consistency between meshes and the regional level-set functions on the multi-fluid interfaces. An extended triangulation template strategy was also proposed to handle triple junctions, which standard marching cube algorithms fail to reproduce.

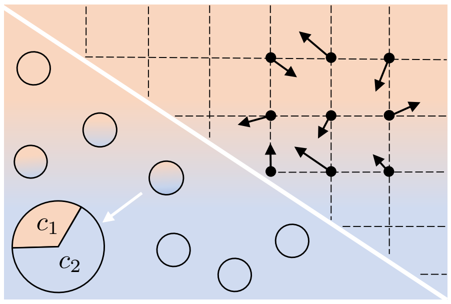


Fig. 25 Schematic diagram of mixing fluids. The lower-left part shows the fractions c_i (volume fraction, mass fraction, or concentration) of different phases in each particle during the diffusion process. The upper-right part shows the 3D fluid grid; black arrows represent grid forces due to diffusion.

6.2 Mixing fluids

Another category of multiple-fluid flows involves miscible or dispersed fluid mixtures, for which interfaces can be challenging to track continuously or may not exist at all. In contrast to the non-mixing case (Sec. 6.1), different phases now always co-exist at the same spatial position (Fig. 25). A key problem is to calculate how the local volume fractions c_k^V of each phase k change during the simulation, *i.e.*, solving the multiphase continuity equations. Other challenges include simulating diffusive behavior, incompressibility enhancements, and stability improvements. We summarize several typical methods that can be integrated into the SPH [167, 168], PBF [153, 169], and MPM [170] solvers.

Mixture model Ren *et al.* [167] used a multiphase mixture model for complex multiphase mixing and unmixing effects. Their SPH-based approach (called WCSPH) solves the mixture continuity and momentum equations for each particle as follows:

$$\frac{\partial}{\partial t}\rho_m + \nabla\cdot(\rho_m\mathbf{u}_m) = 0, \tag{17}$$

$$\frac{\partial}{\partial t}(\rho_m\mathbf{u}_m)+\nabla\cdot(\rho_m\mathbf{u}_m\mathbf{u}_m) = -\nabla p_m+\rho_m\mathbf{g}+\nabla\cdot(\mathbf{T}_m+\mathbf{T}_{Dm}), \tag{18}$$

where $\rho_m = \sum_k c_k^V\rho_k$ is the mixture density, $\mathbf{u}_m = \frac{1}{\rho_m}\sum_k c_k^V\rho_k\mathbf{u}_k$ is the mixture velocity, p_m is the mixture pressure, and $\mathbf{T}_m, \mathbf{T}_{Dm}$ are the mixture viscous stress and diffusion tensors, respectively. Phase velocities are assumed to be different from each other. For each phase k, a drift velocity $\mathbf{u}_{mk} = \mathbf{u}_k - \mathbf{u}_m$ is analytically computed at the start of each time step. These phase-wise drift velocities are used to calculate the phase volume fraction change Dc_k^V/Dt as well as the mixture diffusion tensor $\mathbf{T}_{Dm}$. Following this, the aggregate particle motion, individual phase velocities, and phase volume fraction changes on the particles during the simulation are solved. Yan *et al.* [154] extended this mixture model to cope with solid phases. Ren *et al.* [171] further introduced a virtual phase concept for multiphase simulations containing porous solids, considering the absorbed and non-absorbed parts of a single phase as two virtual phases that can be universally handled by the mixture model. The result was a unified algorithm framework for multiphase flows inside and outside porous solids. To alleviate the incompressibility issue of the WCSPH framework [167], Jiang *et al.* [168] used volume-weighted mixture velocities $\mathbf{u}_m = \sum_k c_k^V\mathbf{u}_k$ to ensure a divergence-free mixture velocity field solvable by an iterative incompressible SPH solver for the single-fluid case. To capture multiphase fluids with highly dynamic relative motions, Jiang and Lan [172] presented a dynamic mixture model that abandoned the local equilibrium condition. This method also allowed for fluid control in the multiphase environment by solving the Navier–Stokes equations for each phase flexibly. In contrast, Ren *et al.* [173] used the deformation gradient to construct a set of linear equations that match the local volume change resulting from the momentum-equation-solved velocities, which resulted from the continuity-equation-solved fraction changes, and solved these equations for enhanced incompressibility.

Non-trivial diffusion Traditionally, phase-mixing effects in fluid simulation have been modeled using the diffusion equation $\frac{Dc}{Dt} = \alpha\nabla^2 c$, where α is the diffusion coefficient and c is the concentration, which assumes uniform phase velocities and movement in accordance with the aggregate motion. This approach has been employed in various works, such as Im's [174] diffusive dissolved air transfer model for calculating bubble distribution in freezing ice blocks and He *et al.*'s [175] two-phase diffusive model for simulating diffusive appearances with varying sharpness in materials like ink and bubbles.

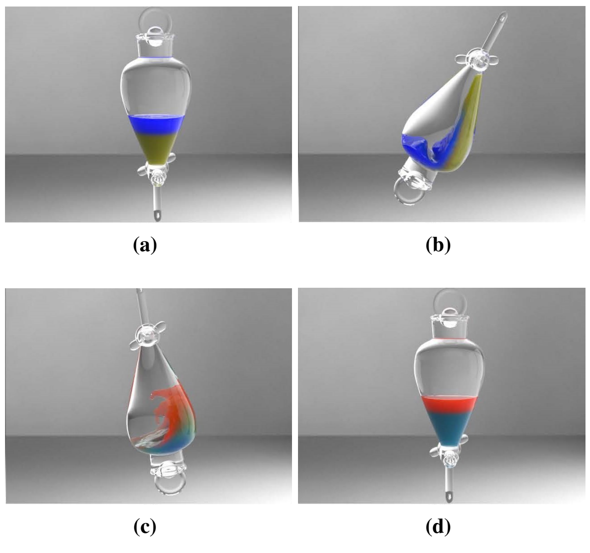


Fig. 26 Fluid extraction. From left to right, we see the change of solution due to greater solubility of the green liquid within the blue liquid compared to the red liquid. (a) At the beginning, the blue liquid and yellow mixture are put in a separating funnel. (b) The funnel is toppled. (c) Shaking vigorously mixes the fluids. (d) Turning the funnel upright results in a clear interface between the red liquid and cyan mixture [169].

Other researchers have advanced this field with more sophisticated models. Yang *et al.* [169] integrated the Cahn–Hilliard equation into multiphase simulation using an energy-based model to capture complex multiphase effects, such as unmixing and extraction. They computed the change of the mass fraction c_k^m of each phase k as

$$\frac{Dc_k^m}{Dt} = \nabla \cdot (\mathcal{M}\nabla\phi_k), \tag{19}$$

where $\mathcal{M}$ is a degenerate mobility constant and ϕ_k is the k-th phase's chemical potential relying on the derivative of a case-specified Helmholtz free energy function at the current concentration composition. This model was able to capture complex multiphase effects, such as unmixing and extraction (Fig. 26). Yang *et al.* [153] extended this model using a unified Helmholtz free energy form to handle both solid and liquid phases, thereby expanding the capacity of the PBF multiphase solver. Chen *et al.* [170] proposed a moving least square reproducing kernel particle method for better precision and stability of particle-based simulations. Using an advanced interpolation scheme, they integrated the Cahn–Hilliard equations into MPM solvers and achieved good mass conservation, stability, and sub-grid details in multiphase fluids.

Xue *et al.* [176] modeled anisotropic diffusive effects using non-Fourier diffusion, which was integrated into a phase field formulation using an MPM solver. The resulting constitutive model is given by

$$\begin{aligned} \boldsymbol{q} &= \boldsymbol{q}_C + \boldsymbol{q}_F, \\ \boldsymbol{q}_C + \tau\dot{\boldsymbol{q}}_C &= -\left(1 - G_T\right)\alpha\nabla X, \\ \boldsymbol{q}_F &= -G_T\alpha\nabla X, \end{aligned} \tag{20}$$

where q is the associated diffusion flux, q_C and q_F represent Cattaneo and Fourier diffusion, respectively, τ is the relaxation time with respect to the flux, α is the diffusion coefficient, and X is the quantity being diffused. G_T is an non-dimensional parameter that represents the weight between Cattaneo-type and Fourier-type diffusion. Their method reproduced complex folding effects of poroelastic materials during wetting and also directional diffusive transportation effects. Su *et al.* [177] adopted the anisotropic diffusive model in [176] for temperature transport in an extended MPM phase change solver, allowing the simulation of richer phenomena. They also introduced an integration scheme that provides second-order accuracy with only first-order algorithmic overhead.

In recent years, additional works have studied other mixing-related phenomena. Stomakhin *et al.* [178] proposed an MPM approach to solve heat-induced phase change of various materials. A carefully designed projection solver allowed them to simulate nearly incompressible phase-changing materials in MPM. Hochstetter and Kolb [179] presented an SPH method to simulate evaporation and condensation of liquids. Their technique utilized particles to signify the liquid phase, while the grid primarily served as a medium for simulating the air phase and facilitating water vapor transport. This method used Fourier's law as a basis for heat transfer between grid cells and particles, thereby advancing the understanding of multiphase heat and mass transfer phenomena.

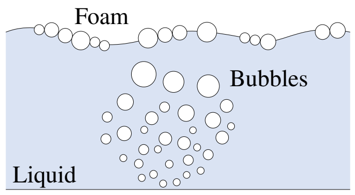


Fig. 27 Schematic diagram of bubbles and foam. Gas accumulates under water and forms bubbles. As bubbles rise to the water surface, their volume increases due to a pressure decrease. When reaching the surface, bubbles may form foam.

7 Gas–liquid interfaces

In fluid simulations, the influence of gas is often ignored. However, numerous real-world fluid phenomena, including the formation of water droplets and bubbles, cannot be accurately represented without considering the role of gases. The phenomena formed by gas–fluid interactions are complex and diverse. In this section, we discuss gas–liquid interface phenomena by grouping them into three categories: free surface fluids (Sec. 7.1), bubbles, foam, and glugging (Sec. 7.2), and spray and splashing (Sec. 7.3).

In free surface fluid simulation, the emphasis is typically placed on calculating the fluid surface and accounting for surface tension, rather than explicitly modeling the presence of air or gases. Here, we introduce some typical methods, such as contact angle, surface tracking, and continuous surface force. We then discuss bubbles, foam, and glugging together because of their similar characteristics, i.e., they are the result of a small amount of gas being wrapped by the closed fluid. The bubbles we are discussing here are those formed by air gathering in water, while foam is formed by bubbles rising to the surface of the fluid (Fig. 27). Glugging occurs, *e.g.,* when a liquid is rapidly poured from a bottle with a narrow opening (Fig. 28) and is a multiphase phenomenon where bubbles are generated automatically. Finally, spray and splashing are formed by free liquids in the gas. These are usually produced by violent collisions of fluids and require more accurate simulation methods.

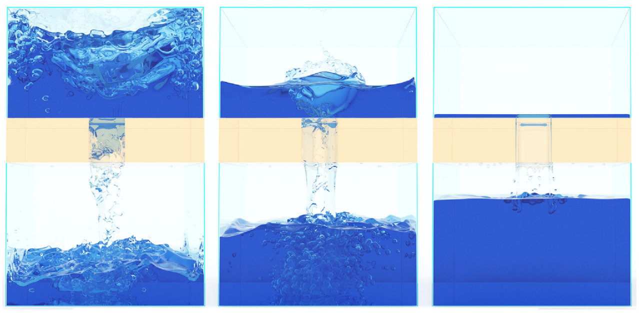


Fig. 28 The glugging effect [180].

7.1 Free surface fluids

Physically correct or at least plausible gas–liquid interface modeling is challenging. This is largely due to the fact that while scalar fields such as pressure can be approximated well using particles or grids at macroscopic scales, surface tension (and similar) effects are the result of microscopic intermolecular forces (Fig. 29). This makes introducing surface tension effects into standard Lagrangian and Eulerian solvers (see Sec. 3.3) non-trivial.

Wang *et al.* [181] introduced the *contact angle* to calculate surface tension. This angle exists at the junction of solid, liquid, and gas, which indicates the hydrophilicity and hydrophobicity of solid materials (Fig. 30). They used signed distance fields to represent such surfaces and constructed a

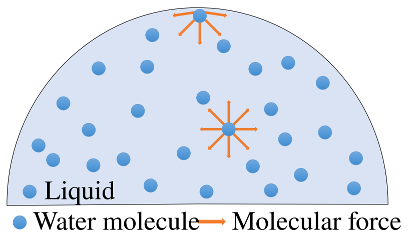


Fig. 29 Schematic diagram of surface tension. Liquids have forces between the same-kind molecules (cohesion) and different-kind molecules (adhesion). The molecular force on the liquid surface is unbalanced, resulting in surface tension effects.

virtual surface below the solid one to replace the real solid–fluid interface. The distance field can be modified by the virtual surface. Following this, the stable contact angle θ_s can be obtained to estimate the surface tension from

$$\gamma_{sa} - (\gamma_{la} \cos\theta_s + \gamma_{ls}) = 0, \tag{21}$$

where γ_{sa}, γ_{la}, and γ_{ls} are the interfacial tension coefficients for the solid–air, liquid–air, and liquid–solid surfaces, respectively. However, this method uses a grid to represent the internal volume of the fluid, which requires significant memory and computation time.

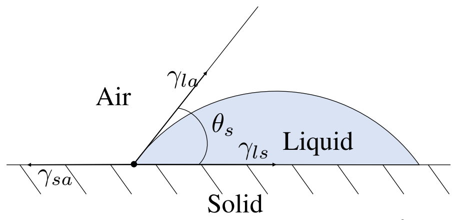


Fig. 30 Schematic diagram of stable contact angle θ_s. γ_{sa}, γ_{la}, and γ_{ls} are the interfacial tension coefficients for solid–air, liquid–air, and liquid–solid surfaces, respectively. When a drop of liquid rests on a solid surface in equilibrium, the angle between the solid–liquid interface and gas–liquid interface is called the stable contact angle.

Water drop animation was the main focus of Zhang *et al.* [182]. The crucial part of their Lagrangian system that allows efficient simulations of water drop motions is the reduction of volumetric fluid dynamics over the whole liquid volume to a deformable surface model. While also using the contact angle method like [181], their model focuses only on the surface, and as such, it is more computationally efficient.

Da *et al.* [183] proposed a surface-only model that avoids dealing with degrees of freedom inside liquids and (often) far away from their surface. This is the first such model for 3D liquids with the first advection–projection scheme for surface-based liquids, albeit partly limited to bodies dominated by surface tension and inertia, although still capable of modeling effects such as crown splashing.

Akinci *et al.* [184] employed SPH to model surface tension and adhesion forces. Theirs is the first method that correctly

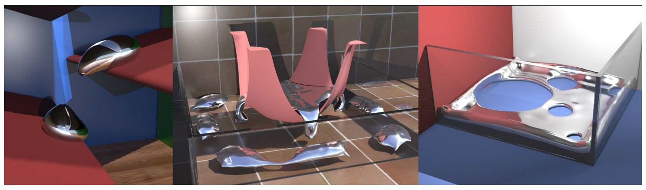


Fig. 31 Simulation of high-surface-energy liquids, including shape change and two-way coupling with solids [188].

handles large surface tension (and adhesion) without the need for ghost particles or artificial pressure forces. The cohesion force is described as

$$\mathbf{F}_{i\leftarrow j}^{coh} = -\gamma m_i m_j B\left(||\mathbf{x}_i - \mathbf{x}_j||\right) \frac{\mathbf{x}_i - \mathbf{x}_j}{||\mathbf{x}_i - \mathbf{x}_j||}, \tag{22}$$

where γ represents the surface tension coefficient and B denotes a spline function. The method is simple to integrate with existing SPH solvers and can simulate effects such as water crown formation and rolling water droplets.

By assigning each particle a value corresponding to an estimate of its surface area, leading to an implicit definition of the free surface of the fluid, Orthmann *et al.* [185] achieved conservative transport within and between surfaces, including correct handling of thin sheets and other singularities. This allows for effective simulations of detergents, cleansing, and coating.

Yang *et al.* [186] used a pairwise-force model called PF-SPH, which relies on larger support radii than traditional SPH. The method improves the accuracy of the surface tension calculation by using anisotropic filtering to scale neighboring particle interaction forces.

Energy-based methods have also been used to simulate free surface fluids. He *et al.* [187] modified earlier surface tension and air pressure formulations for SPH-based free surface flows, building on the diffuse interface model. They introduced a modified surface tension energy formulation $\mathcal{E}^s$ as

$$\mathcal{E}^s = \int_V \frac{\kappa}{2} ||\nabla l||^2 dV, \tag{23}$$

where V represents the volume of the liquid, κ is a coefficient associated with squared gradient energy, and l denotes the condensation field. The surface tension energy $\mathcal{E}^s$ is directly related to the surface area of a fluid interface. Its gradient can be computed to determine the surface tension force acting on the interface. This improves the robustness of the model *vs* particle sparsity and in turn leads to increased stability. The model can simulate delicate surface tension effects, such as water/milk crowning.

Classical methods often struggle with relatively high coefficient/parameter values, such as those controlling surface energy. To address this, Hyde *et al.* [188] developed an implicit Lagrangian formulation that specifically targets liquids with significant surface energy, such as liquid metals (Fig. 31). By treating discrete forces as gradients of the potential energy that are proportional to the surface area of the liquid, this approach enables more accurate and stable simulations. Chen *et al.* [189] proposed an MPM approach that generalizes the work of [188] by improving resampling via new types of temporary "balance" particles, achieving the perfect conservation of grid linear and angular momenta.

7.2 Bubbles, foam, and glugging

Indeed, the influence of gas on fluid simulation extends beyond just the free surface of the fluid. It encompasses the behavior of the fluid interior and involves more intricate interaction processes. Single-phase liquid simulations typically struggle to capture phenomena like bubbles, foam, and glugging effects, which necessitates the modeling of gas and liquid as two-phase flows.

Patkar *et al.* [190] presented a hybrid Lagrangian–Eulerian scheme for converting between small (*i.e.*, sub-grid and under-resolved) Lagrangian bubbles and larger well-resolved bubbles modeled with an Eulerian approach based on level sets. Their framework includes a bubble seeding mechanism to realistically simulate fluid structure interaction with complex (moving) objects. Cho and Ko [191] combined the volume of fluid (VOF) with sub-grid refinement of the level set method to simulate moving interfaces in two-phase flows.

Goldade *et al.* [192] developed a model for immersed bubble simulation, which avoids advection and projection inside bubbles. The method is based on constraint-based incompressible bubbles (with zero density) and affine fluid regions (to account for non-zero density coefficients). The simulation region is divided into a fluid region Ω_f, solid region Ω_s, and air region Ω_a. Any enclosed and continuous region filled with air is treated as a bubble. Linear velocity constraints are imposed on each bubble via

$$\iint_{\Omega_f \cap \partial\Omega_{a_i}} \mathbf{u}_f \cdot \mathbf{n}_f \, da + \iint_{\Omega_s \cap \partial\Omega_{a_i}} \mathbf{u}_s \cdot \mathbf{n}_s \, da = 0, \tag{24}$$

where Ω_{a_i} is the continuous region of bubble i. $\mathbf{n}_f$ and $\mathbf{n}_s$ are the fluid's normal and solid boundary's normal, respectively.

Additional special bubble simulations have been conducted. Paddilla *et al.* [193] modeled bubble rings via vortex filaments of variable thickness, assuming that advective inertial forces are small compared to viscous forces. Filaments are expressed as a configuration manifold on which the equations of motion are geodesic. Langlois *et al.* [194] introduced a set of techniques aimed at generating sound representations for intricate two-phase liquid animations. They extended the open-source

Gerris solver [195] (a finite-volume-based multigrid solver) to achieve audio–visual fluid (and bubble) simulations.

Although some of the above methods can handle foam to some extent, specialized methods exist for this. Busaryev *et al.* [196] animated bubble interactions in liquid foams by treating (small) bubble particles as sites of Voronoi cells in a weighted diagram. Their framework handles bubble–bubble, bubble–liquid, and also bubble–solid interactions, giving rise to foam simulations with bursting and coalescing. Kim *et al.* [197] modeled foam waves using the FLIP solver. Foam particles projected to 2D give rise to depth and acceleration maps, making the method efficient. The method provides the option to art-direct the foam effects using sketches and level-of-detail controls.

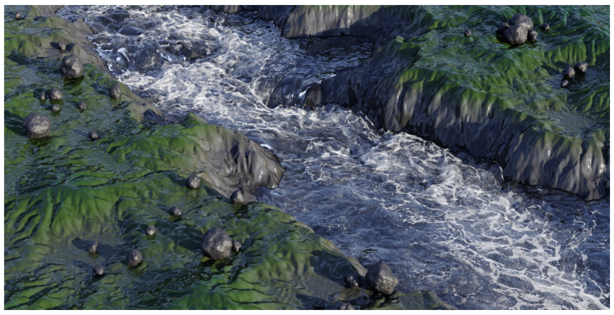


Fig. 32 River. A river flowing around sharp creases of a winding canyon creates a combination of calm and dynamic regions, including waterfalls and backdrafts. [198] ©Weta FX.

Recently, Wretborn *et al.* [198] presented a realistic model for white-water simulation (Fig. 32). Their method enhances simulations with (tiny) bubble and foam detail by a stable coupling scheme between bubbles and water, a novel bubble emission scheme, and manifold advection for accurate foam tracking.

For the simulation of glugging, Boyd and Bridson [199] proposed the MultiFLIP method, which extended the FLIP method to two-phase flows. They treated not only liquid but also air as incompressible phases, both modeled via particles. This (re)produces, among other effect, the glugging effect.

Ando *et al.* [200] introduced a stream function-based solver as a FLIP variant. In this approach, the stream function ψ is used to determine the divergence-free velocity field $\mathbf{u}$, given by $\mathbf{u} = \nabla \times \psi$. Interestingly, their work shows that solvers based on stream functions are just as viable as regular pressure solvers. The method is able to simulate glugging without modeling the second phase (air) explicitly.

7.3 Spray and splashing

Spray and splashing are very common phenomena in fluid scenes (Fig. 33). For scenes with intense collisions like turbulence, the final visual effect largely depends on the fidelity of the spray and splash simulation.

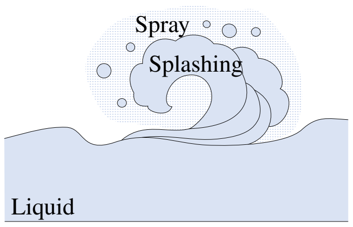


Fig. 33 Schematic diagram of spray and splashing. Spray and splashing are different in simulation scale. Compared with splashing, spray is composed of finer droplets.

Nielsen and Østerby [201] modeled spray as a two-way coupled two-continua with different volume fractions to achieve realistic spray motion. However, a grid-based density field cannot capture the motion of a single droplet. In contrast, Jones and Southern [202] focused on efficient physics-based droplet interaction. They introduced coalescence, separating, and fragmenting collision outcomes into a novel particle interaction model to simulate droplets. This provides a ballistic particle system for liquid droplets and spray.

Yang *et al.* [203] focused on spray simulation, such as that arising from high-speed/violent liquid streams. Similar to [190], they also used a hybrid Lagrangian–Eulerian model (with FLIP components in their case) to model mixture phenomena with high fidelity. Their efficient CUDA implementation allows droplet and spray effects, such as those arising in waterfall and fountain simulations, to be modeled.

Guo *et al.* [204] addressed the stability challenges encountered in the two-phase lattice Boltzmann model (TP-LBM) by introducing a novel density-aware sub-grid-scale model. Their approach can uniformly simulate different gas–liquid phenomena, allowing for realistic and visually compelling representations of gas–liquid flow dynamics.

Li *et al.* [180] proposed a multiphase flow method to simulate complex effects, such as bubbling, glugging, wetting, and splashing. A single model captures all these effects by building on the kinetic-based Lattice Boltzmann Phase-Field (LBM-PF) method. The interface motion is governed by the conservative phase-field equation

$$\frac{\partial c_\Phi}{\partial t} + \nabla \cdot (c_\Phi \mathbf{u}) = \nabla \cdot \left[\mathcal{M} \left(\nabla c_\Phi - \frac{4}{\xi} c_\Phi (1 - c_\Phi) \mathbf{n}_{\text{inter}} \right) \right], \tag{25}$$

where the phase field c_Φ represents the percentage of the flow phase, the mobility $\mathcal{M}$ controls the degree of interface splitting, ξ denotes the interface width, and $\mathbf{n}_{\text{inter}}$ corresponds to the interface normal. This method is highly general and versatile and can produce results comparable to industry-standard CFD simulations.

In a similar vein, Li *et al.* [205] introduced a kinetic approach to multiphase fluids. Their scheme incorporates an accurate collision model and is able to robustly capture intricate and visually-appealing behaviors, such as the injection of gas into the liquid (Fig. 34).

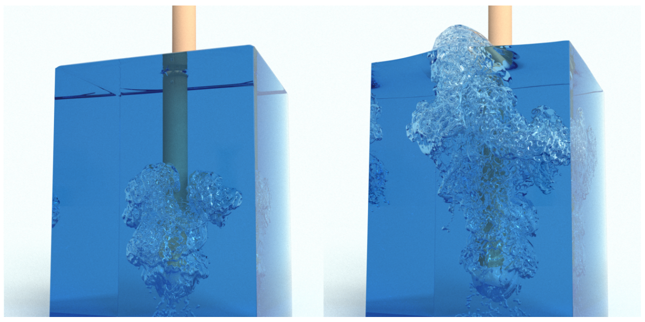


Fig. 34 Injection of gas into a glass of water [205].

8 Fine detail enhancement

Capturing high-frequency details of fluid surfaces, such as vortices, waves, and turbulence, is key to enhancing the realism or artistry of fluid simulations. The use of higher-order advection numerical methods or finer discretization can alleviate the numerical dissipation problems inherent in fluid simulation. However, this creates very high memory and computation time costs. To cope with this, in the last decade, several methods have been designed to specifically add fine details to a coarse fluid simulation. We group these into three classes: reduced-dimensional simulation on the fluid surface only (Sec. 8.1), dynamic methods to combat numerical dissipation (Sec. 8.2), and data-driven methods (Sec. 8.3).

8.1 Reduced-dimensional simulation on the fluid surface only

These techniques decouple the surface simulation from the volume simulation, allowing a secondary model with high-resolution surface features to be added to a coarse (thus fast to compute) volume fluid model. We further split methods in this class into embedding techniques, 2D water wave simulation, and surface tracking and reconstruction as follows.

Embedding techniques The Closest Point Method (CPM) [206] is a numerical method for solving PDEs on surfaces. Unlike 2D surface parametrization, CPM typically uses a 3D Cartesian grid to discretize narrow spatial bands around the surface, which allows it to scale according to the complexity of the surface rather than the volume.

Auer *et al.* [207] used CPM to simulate fluid effects on static surfaces in real time. Auer and Westermann [208] followed up with a semi-Lagrangian CPM that alleviated some technical limitations of previous applications of CPM to deformed surfaces. Their method is unconditionally stable for surface deformation. Kim *et al.* [209] used CPM to explicitly perform high-resolution wave simulations on the liquid surface. They used the iWave algorithm [210] to produce more realistic water waves than the traditional wave equation, which can be expressed as

$$\frac{\partial^2 H}{\partial t^2} = -g\sqrt{-\nabla^2}H, \tag{26}$$

where H is the fluid height, g is the gravity constant, and $\sqrt{-\nabla^2}$ is a fractional Laplacian operator.

Mercier *et al.* [211] added a sub-grid wave model to particle-based liquid simulations to enhance such simulations with additional turbulence. Goldade *et al.* [212] worked to eliminate sub-grid errors in underlying surfaces and reduced artifacts in narrow bands around surfaces. Morgenroth *et al.* [213] used CPM to efficiently compute high-resolution 2D simulations on rough surfaces. Their method is similar to that of [208] but adds mass and momentum conservation and can produce interesting effects, such as oil films on water surfaces and thermal convection on a hemisphere.

2D water wave simulation To reduce computational complexity while retaining surface detail, some researchers have investigated the simulation of water waves on fluid surfaces. Water wave simulations are independent of degrees of freedom, so high-frequency visual detail can be created without increasing the simulation resolution.

In response to the inability of the shallow water equations (SWE) to capture motion details such as wakes, Pan *et al.* [214] used a 2D discrete vortex method to capture the wake behind a moving rigid body. Their method requires only a small number of wake particles and is sufficiently fast for real-time applications. However, this method fails to handle the complex wake patterns caused by vortex stretching and tiling in 3D flow. Azencot *et al.* [215] used a scalar vorticity function on 2D domains to describe the vortex behavior of fluid surfaces, greatly simplifying the analysis and simulation of fluids. Later, Azencot *et al.* [216] simulated the complex behavior between multiple waves, including annihilation, recreation, splitting, and merging, by solving the EPDiff [217] on arbitrary triangle meshes using an explicit structure-preserving numerical scheme.

A key challenge in wave simulation is handling the coupling between waves and obstacles. Canabal *et al.* [218] generated rich water waves using a dispersion kernel as the spatially variant filter and further simulated interactions between waves and static or moving obstacles by modulating this dispersion kernel. The dispersion relation under the Airy wave theory [219] defines the propagation speed of each wave u_c as

$$u_c = \sqrt{\left(\frac{g}{w} + \frac{\gamma}{\rho} w\right) \tanh(wH)} \tag{27}$$

where w is the wave number, g is the gravitational constant, ρ and γ are the density and surface tension of the fluid, respectively, and H is the fluid height. Jeschke and Wojtan [220] simulated the movement and interaction of a large amount of waves by a wavefront tracking algorithm with multivalued function interpolation. Their method can model the dispersion, refraction, reflection, and diffraction of waves well, but it only handles scenes with static obstacles. Later, they introduced the concept of wave packets [221], which can handle the interaction of water waves and moving objects. They used an improved Lagrangian particle method to simulate the diffusion of water waves to add more visual detail. However, this method cannot be extended to moving 3D fluid simulations of surfaces. The same problem was addressed by Skrivan *et al.* [222], who decoupled the wave resolution from the simulated resolution using Lagrangian wave packets. This method significantly increases the visible detail on the fluid surface as a post-processing step.

Creating large open-water animations and adding wave detail is a common requirement for a variety of interactive and offline applications. Implementing this requires visual quality *vs* computational resources to be carefully balanced. Nielsen *et al.* [223] proposed a wave synthesis technique based on the Fourier transform to enhance the details of wave animation. However, wave–obstacle interactions are difficult to incorporate into the spectral solver. Keeler and Bridson [224] proposed an efficient surface-only simulation of deep ocean waves and used a new indirect boundary integral equation to deal with wave–solid boundary interactions. The Method of Fundamental Solutions (MFS) was also used to generate realistic waves behind moving obstacles. Schreck *et al.* [225] proposed a novel discretization for MFS using wavelets and achieved naturally-looking wave interactions with complex boundaries (Fig. 35). Their method achieved impressive results on a large-scale ocean scene. Jeschke *et al.* successively developed two interactive systems for the simulation of large ocean scenes that can handle detailed wave features [226] and coupled interactions with complex terrain [227], respectively. Recently, Schreck and Wojtan [228] proposed a coupled method of 3D liquid simulation and 2D wave propagation to simulate infinitely large bodies of water and fine surface wave detail. An empirically-driven error compensation method was also used to remove coupling errors from the simulation to achieve a seamless transition between 2D and 3D.

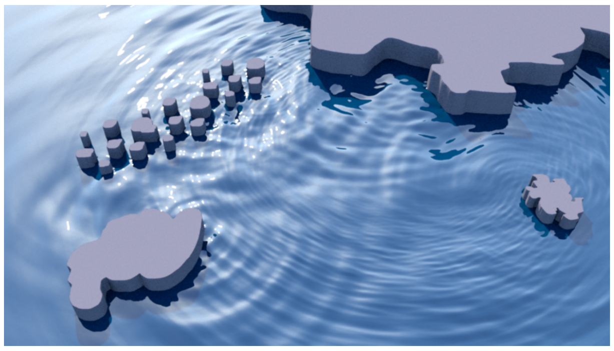


Fig. 35 Water waves can accurately interact with complex boundaries [225].

Surface tracking and reconstruction Tracking and reconstructing fluid surfaces is important for generating realistic animation effects. This is difficult to achieve due to the complex shapes and frequent topological changes of fluids.

For Eulerian fluid simulations, robust handling of surface triangle mesh splitting and merging can remove visual artifacts to preserve important surface features. Bojsen-Hansen *et al.* [229] proposed a method for tracking the topological evolution of surfaces that can solve the wave equations on lower-resolution fluid surfaces to synthesize high-frequency details. Later, Bojsen-Hansen and Wojtan [230] presented a novel physics-based surface fairing method that solved the physical and topological artifacts arising from coupling high-resolution surface trackers with low-resolution fluid simulations by introducing an error metric and surface correction force. Edwards and Bridson [231] presented a new approach to adaptive fluid simulation. They tracked explicit triangulated mesh surfaces and used the p-adaptive Discontinuous Galerkin (DG) method [232] within detailed cut-cells near the surface. By using coarse-grid fluid simulations, the treatment of dynamics is guaranteed to be physically consistent while reducing computational costs. Chentanez *et al.* [233] devised a grid-free surface tracking method that deals with topological changes by removing overlapping triangles and performing effective triangulation of the generated holes. This method can be used in both mesh-based and particle-based simulations.

Inaccurate detection of free surface particles in particle-based fluid simulations can lead to unrealistic artifacts. Also, irregularly distributed particles can make the reconstructed surface bumpy. To address the fact that particles do not keep connectivity information, Yu *et al.* [234] proposed periodically projecting surface meshes to match implicit surfaces defined by fluid particles. This method allows the simulation of high-resolution surface waves without the limitation of particle resolution. Later, Yu and Turk [235] proposed a method for reconstructing surfaces in particle-based fluid simulations. They utilized a stretched anisotropic smooth kernel to represent each simulated particle, resulting in a greatly improved surface quality (Fig. 36). Sandim *et al.* [236] proposed a fast free surface detection method that only requires the positions of particles to identify surface particles without using kernel functions or normal vectors. This method is applicable to cases with non-uniform particle distributions and complex free surface deformations. Dagenais *et al.* [237] used an explicit mesh projection method based on signed distance fields to preserve surface detail and introduced a new topology matching operation to maintain consistency between explicit surface and particle behavior.

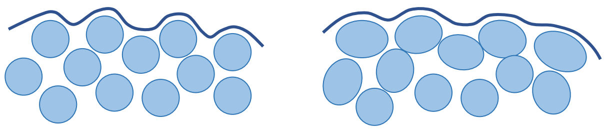


Fig. 36 Schematic diagram of isotropic and anisotropic particles. In contrast to isotropic particle fluids (left), anisotropic particle fluids (right) have a smoother surface.

8.2 Dynamical methods for reducing numerical dissipation

The advective-projection method leads to *numerical dissipation*, which results in kinetic energy decay and suppression of motion, such as vortices and turbulence. Bulk enhancement methods aim to improve the whole fluid volume rather than only its free surface. We further group such methods that aim to improve system energy conservation and detail preservation by reducing numerical dissipation in vorticity confinement, vortex-based methods, and various variants of dynamics solvers, as described next.

Vorticity confinement methods. Such methods are based on the principle of vorticity conservation, which adds a vorticity control term to restrain the diffusion of vortices, thus simulating fluid dynamics problems, such as turbulence and vortex streets, without dissipation. Fedkiw *et al.* [238] first applied the vorticity confinement method to smoke simulations by adding an additional force field to maintain the airflow vorticity. Second-order vorticity confinement (VC2) further improved this method by ensuring momentum conservation. The confinement term of VC2, $\mathbf{F}_{\text{conf}}$, is given by

$$\mathbf{F}_{\text{conf}} = -\Delta x \nabla \times (\alpha \omega - \zeta \mathbf{m}), \tag{28}$$

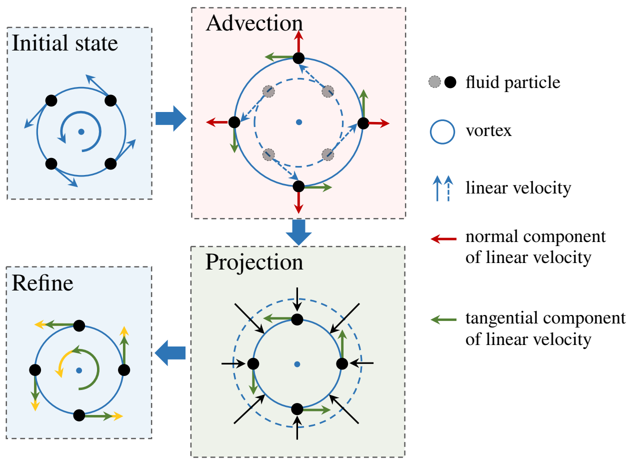

Initial state
Advection
Projection
Refine
fluid particle
vortex
linear velocity
normal component of linear velocity
tangential component of linear velocity

Fig. 37 Schematic diagram of the causes of vorticity dissipation and methods of refinement.

where Δx is the grid size, α and ζ are the positive and negative diffusion coefficients, respectively, ω is the angular velocity, and $\mathbf{m}$ is the harmonic mean of the local vorticity stencil. Lentine *et al.* [239] improved the vorticity confinement model by allocating global momentum to ensure momentum conservation. Jang *et al.* [240] applied the multi-level vorticity confinement method to simulate water turbulence to capture large- and small-scale vortices and complex flow details. He and Lau [241] proposed adaptive adjustment of the positive diffusion term to balance constraints, which broadened the stability conditions of the VC2 method and made it capable of generating highly turbulent flows. However, the vorticity constraint method can only enhance existing vortices or turbulence and may not be effective for other types of flows, such as laminar flow. Moreover, computational costs can increase significantly due to additional constraints, especially for large-scale complex fluid scenarios.

Vortex-based methods The potential vorticity field can be effectively modeled using vortex-based methods. These methods simulate the vorticity of the velocity field rather than the velocity field itself, so this automatically guarantees a divergence-free velocity field and removes numerical dissipation. Most such methods are Lagrangian and model the vorticity form ω_v of the Navier–Stokes equations as

$$\frac{\partial \omega_v}{\partial t} + (\mathbf{u} \cdot \nabla)\omega_v = (\omega_v \cdot \nabla)\mathbf{u} + \mu \nabla^2 \omega_v, \tag{29}$$

$$\nabla \cdot \omega_v = 0. \tag{30}$$

This formulation represents the vorticity distribution as a superposition of singularities.

Zhang *et al.* [242] proposed a new scheme named IVOCK, which aims to solve the errors and energy loss caused by the self-advection step through compensating for vorticity error. However, this method is only applicable to fluid simulation on uniform grids. Liu *et al.* [243] extended this idea to particle-based turbulent detail simulation (Fig. 37). Recently, Xiong *et al.* [244] proposed a vortex segment method to simulate flows with strong anisotropic vortical features. Compared with existing Lagrangian vortex particle methods, this method can more vividly model complex phenomena, such as the splitting and reconnection of two vortex tubes or vortex shedding near a solid boundary.

The main challenge of vortex methods is the handling of *fluid–solid coupling* and creating vortices at this coupling boundary. For this, Golas *et al.* [245] proposed a combination of Eulerian simulation and vortex singularity bases. By using Lagrangian vortex elements inside the fluid and enforcing boundary conditions in the Eulerian mesh, robust interaction of free surfaces and non-rigid obstacles can be achieved. Zhang *et al.* [246] used a FLIP approach to solve the Navier–Stokes equation using viscous particle strength exchange, handling the momentum transformation at the solid boundary effectively. Liao *et al.* [247] proposed a new wall-bounded turbulent smoke simulation method, which introduced particle–particle interactions to traditional vortex filament mesh calculations to accurately capture the vortices and thin turbulence generated by smoke–obstacle interactions.

Variants of dynamics solvers Additional methods improve on current advection–projection solvers or extend classical dynamical methods to achieve detail enhancement. In the advection–projection step, detail and energy preservation are greatly improved by the introduction of detail capture and shape correction techniques [248], the use of energy-preserving reflection operators [249], and feature mapping with convectors [250].

Yang *et al.* [251] first introduced the Clebsch wave function as a system scaling variable to evolve Eulerian flow fields, which significantly improved the ability to generate and sustain vorticity in simulations of various gases and liquids. Later, to solve the numerical instability of Clebsch's method near dynamic interfaces, Xiong *et al.* [252] proposed a new wave function correction scheme and extrapolation algorithm, which achieved detailed simulation of various vortex structures on free surfaces. Recently, Feng *et al.* [253] proposed a numerical method for solving the Navier–Stokes equations based on the pulse gauge transformation, which can generate rich vortical details by treating the fluid pulse as an auxiliary variable.

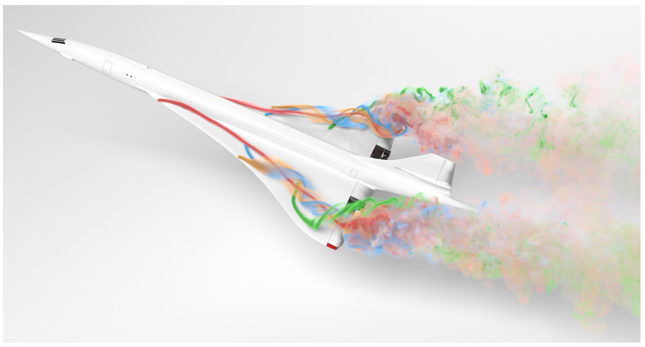


Fig. 38 Airflow over the delta wing of an aircraft showing a realistic and complex vortex distribution [256].

Liu *et al.* [254] and Li *et al.* [255] created two turbulence simulation methods using an adaptive multi-relaxation scheme and statistical mechanics, respectively. Later, Lyu *et al.* [256] further improved the boundary treatment of dynamic solids, enabling the simulation of fluid–solid coupling between thin structures and turbulent fluids (Fig. 38).

The micropolar fluid model is an extension of the classical Navier–Stokes equation, which takes into account not only the linear but also angular velocities of the fluid particles, thus enhancing the eddy and turbulence details of the fluid. Bender *et al.* [257] first used the micropolar fluid model to simulate the turbulence phenomenon of non-viscous fluids. Subsequently, they post-processed the foam phenomenon on this basis to significantly improve the realism of the visual effect.

8.3 Data-driven methods for detail enhancement

Texture synthesis Geared toward the production of artistic effects based on fluid elements, texture synthesis is the technique of choice for adding detail to surfaces. As a post-processing method, texture synthesis achieves detailed surface features through patch or style transfer based on deep learning.

Patch-based texture synthesis maps image textures or simulated features to the target flow field, improving the appearance of the source simulation by matching the target dataset features. Jamriška *et al.* [258] used per-pixel best-fit search to achieve rich visual effects through 2D input image appearance transfer. However, this method is limited to image space synthesis and is difficult to extend to 3D fluid surfaces. Gagnon *et al.* [259] proposed a temporally coherent patch-based texture synthesis method to handle scenes with significant deformation and topological changes. This approach aims to maintain a Poisson disk distribution of patches on a free surface to find the optimal parameter values and locations of time-varying patches. For the ghosting problem of overlapping patches in [259], Gagnon *et al.* [260] followed up with a solution scheme based on patch erosion. They used feature-aware erosion to remove patch distortion textures to ensure realism of the mapping.

In contrast to surface texture synthesis, *deep neural networks* perform various stylization tasks on volumetric data by extracting 2D image features. Sato *et al.* [261] proposed a style transfer method that migrates high-resolution turbulent details to low-resolution flow fields, which speeds up the fine surface detail simulation almost 30-fold. In addition, they used an optimized texture synthesis method to solve the problem of discontinuity at the patch boundary. Kim *et al.* [262] first proposed a transport-based neural style transfer algorithm that enables automatic conversion of the semantic structure of 2D images into 3D smoke simulations. The method achieves complex artistic effects by optimizing the transport of smoke to the desired stylized velocity field at each time step. However, this method cannot handle the transfer of color information. Therefore, the authors further redefined the method in a Lagrangian setting to ensure better temporal consistency and support for color stylization [263]. Unlike stylization methods that focus on fluid simulation shapes, Guo *et al.* [264] proposed a Stylizing Kernel Prediction Network (SKPN) aimed at stylizing physical color appearances. The method can easily generate the user's desired color appearance without complex parameter tuning.

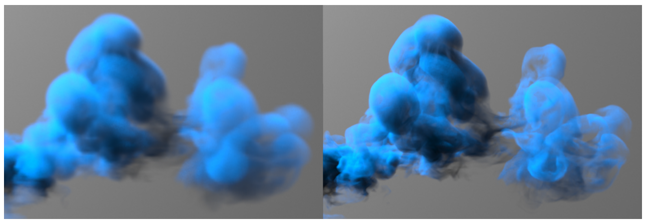


Fig. 39 Using TempoGAN [265] to generate high-resolution smoke volumes from low-resolution inputs.

Upsampling methods generate super-resolution Generating super-resolution simulations from low-resolution inputs is challenging. With the popularity of machine learning techniques, recovering fine details of fluids through super-resolution reconstruction techniques or upsampling methods has received increasing attention.

Ten years ago, Zhang and Ma [266] proposed a spatio-temporal extrapolation technique that enables high-resolution flow features on coarse grids. Chu and Thuerey [267] enhanced the turbulence detail of the smoke simulation on the coarse grid by using local patch descriptors. Um *et al.* [268] proposed a deep neural network to capture small-scale splashed droplet details from low-resolution liquid simulations. Xie *et al.* [265] used a conditional generative adversarial network with a temporal discriminator to directly generate advected quantities with highly detailed and temporally coherent features for smoke simulation (Fig. 39). CNNs have been used to create matching models to correct the shape of low-resolution smoke simulations [269] and estimate physical parameters to guide the reconstruction of high-resolution velocity fields [270]. Bai *et al.* [271] used a dictionary-based neural network for fluid upsampling. However, the choice of training set was somewhat limited, and the spatial and temporal consistency of the results could not be guaranteed. They next significantly improved the prediction quality of the network by adding filtering to the training process [272]. With the development of Deep Neural Networks (DNNs) in recent years, Roy *et al.* [273] proposed a DNN-based method for improving the resolution of coarse particle liquid simulations.

9 Fluid control

The wealth of methods for fluid simulation surveyed so far leads to a key question: How do we *control* such simulations? Although a method can technically produce highly accurate results, its ultimate target is to enable its users to steer the method toward the desired results. We next group and discuss fluid control methods in three classes using different control perspectives: scenario editing (Sec. 9.1), artificial effects (Sec. 9.2), and media-directed formation (Sec. 9.3).

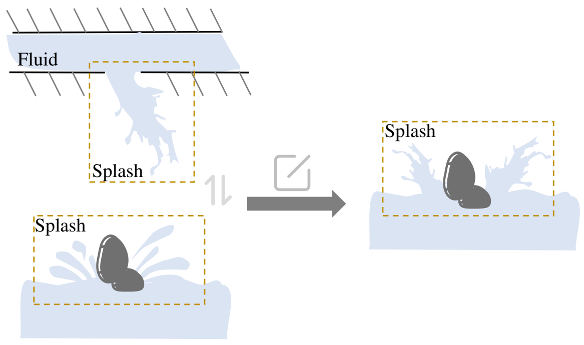


Fig. 40 Schematic diagram of scenario editing based on changing an existing simulation to achieve desired effects.

9.1 Scenario Editing

Scenario editing steers the fluid by generating new simulation scenarios based on existing simulation results without losing the characteristics of the original simulations (Fig. 40). On the positive side, such control is technically the closest to how a simulation works, so it can steer the fluid most directly. On the negative side, this control requires the end users' advanced skills and understanding of the underlying simulation and overall fluid flow technicalities. We further divide scenario editing into three sub-types based on the implementation approach: target-guided editing, adjustable editing, and camera-based editing.

Target-guided editing. Such methods modify an existing fluid simulation to match a given target, *e.g.*, a higher resolution. Gregson *et al.* [274] connected low-resolution smoke capture with its velocity field. They treated the pressure projection as a proximal operator and tracked the fluid by estimating its velocity. Through advection, their method obtained a high-resolution re-simulated smoke. Forootaninia and Narain [275] successfully guided high-resolution smoke flow by replacing its low-frequency component with a given guiding field. Generally, the guiding task is seen as an optimization problem that minimizes errors. This optimization problem was solved efficiently by using a fast primal-dual method [276]. To achieve a more desirable artistic effect, it is essential to guide smoke animation in such a way that it aligns with one or multiple target density keyframes provided by the artist. To address this control problem, Pan and Manocha [277] formulated it as a space-time optimization. They employed an Alternating Direction Method of Multipliers (ADMM) optimizer [278] to derive a dense sequence that forces the smoke to meet the desired target shape. Tang *et al.* [279] proposed an advanced method that effectively addresses both the issues of unconstrained optimization and high-dimensionality of the parameter space simultaneously.

Adjustable editing. These methods aim to control, edit, or resize fluid simulations during implementation. Raveendran *et al.* [280] focused on control with an emphasis on the liquid surface and proposed a method for creating high-quality fluid animations that provides the animator with multiple control levels. Later, Raveendran *et al.* [92] proposed a smooth blending method to interpolate between two or more existing pre-computed liquid simulations. With their method, one can generate hundreds of different plausible results at interactive rates with potential applications in games and virtual reality. Sato *et al.* [281] proposed a smoke blending method to help animators synthesize the desired fluid animation. Velocities at the boundaries are interpolated by minimizing an energy function. This approach significantly reduces computational costs by reusing existing flow data instead of creating realistic fluid animations by numerical simulation. *Fluid carving* is another way to edit fluid simulations. By utilizing seam carving, efficient and effective resizing of 4D fluid simulation data can be achieved [282]. Flynn *et al.* [283] proposed a lattice-guided seam computation method that can overcome the limitation of rectangular boundary and reduced calculation time.

Camera-based editing. In the context of large-scale scene simulation, due to computational cost considerations, it is often necessary to perform coarse-grained simulations of the entire world and subsequently integrate finer-grained details into the scene. To integrate two simulated fluid scenes seamlessly, Bojsen-Hansen and Wojtan [284] presented a fluid modification approach with "non-reflecting" boundary handling. They extended the simple Perfectly-Matched Layers (PMLs) method to handle coupling inflow/outflow boundaries with varying spatial and temporal conditions. The boundary can be modified easily during the simulation, and it handles the multi-resolution combination. Stomanuykhin and Selle [285] introduced the Flow-Animated Boundary (FAB) method, in which the boundary can have a custom shape and vary over time, with materials outside the boundary dynamically removed using volume flux.

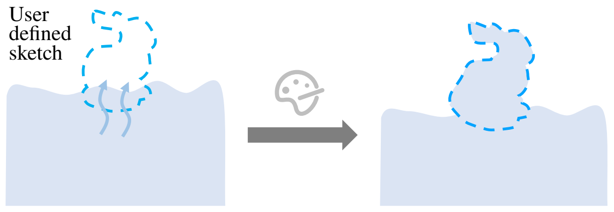


Fig. 41 Schematic diagram of artificial effects: Artist-directed control to achieve non-physical effects.

9.2 Artificial effects

One often needs to artificially edit and control fluids to achieve specific artistic effects. Different from scenario editing, *artificial effects* add new characteristics to a simulation by artificially guiding the formation of fluid shapes or movements during the process (Fig. 41). Due to the complex motion of fluids, keyframe animation, which involves controlling the flow of fluids to match keyframes, is a commonly used method for fluid control to reduce unrealistic effects in simulations. However, manually designed frames often lack volume preservation and exhibit excessive smoothness, resulting in the loss of simulation details.

Pan *et al.* [286] proposed a local control method instead of globally manipulating the entire fluid, allowing users to edit and control fluid shapes in specific regions using a brush-like tool. However, controlling the simulation process between keyframes is challenging. To address this issue, Lu *et al.* [287] drew inspiration from skeletal animation techniques. They introduced a method that controls fluid motion by manipulating a point cloud with rigid body motion and incompressible deformations, subsequently performing skinning operations on the point cloud. Similar to [287], Yan *et al.* [288] applied conditional generative adversarial networks to generate fluid splashes based on simple user-defined sketch input (Fig. 42). Control particles with attractive forces provide an efficient way to reproduce complex motion using pre-computed templates [289]. In this work, a set of shape-constraint particles were seeded and a repulsion force field was computed to control the shape of the final result.

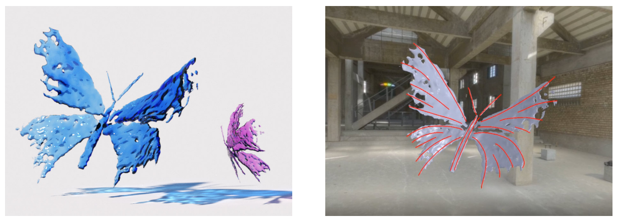

(a) Sketch drawn in a virtual reality environment **(b)** Front view of the model

Fig. 42 Example of artificial effects: Fluid splashed on a butterfly shape generated by user sketching [288].

9.3 Media-directed formation

Pure physics simulations often suffer from significant computational time requirements, making them impractical for real-world production applications. *Media-directed formation* aims to set up simulation scenarios based on real-life videos or images of the fluid, reproducing real-world scenes (Fig. 43). These methods estimate fluid properties such as volume, density, motion, and style from visual data.

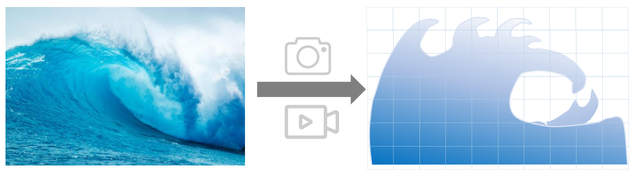


Fig. 43 Schematic diagram of media-directed formation; images or videos are used to reproduce real-world scenes.

Okabe *et al.* [290] focused on reconstructing a detailed 3D model of a fluid volume, such as smoke or liquid, from sparse multi-view images. This involved sparse reconstruction and appearance transfer to capture the underlying structure and enhance the visual fidelity of the reconstructed fluid volume. Unlike [290], Eckert *et al.* [291] estimated both the density and motion of a fluid from a single view or sequence of images without the need for multiple views. Nie *et al.* [292] proposed a fluid reconstruction and editing model to generate particle-based simulations based on monocular videos. Using the SPH method with external forces, they could obtain a simulated fluid volume under the guidance of a pre-processed water surface.

10 Special fluids

10.1 Highly viscous fluids

Viscosity is an attribute that measures the ability of a fluid to resist deformation at a given rate. With viscosity, moving fluids will generate internal stress responding to the deformation, which causes energy dissipation of the fluids and affects their behavior. For low-viscosity fluids like water, inertial forces are dominant. For high-viscosity fluids like molasses and chocolate sauce, viscosity leads to special phenomena like bulking and coiling. Moreover, for different kinds of fluids, the viscous characteristics can vary significantly with their shear rate (see Fig. 44). The simulation of high-viscosity fluids has attracted recent interest in computer graphics.

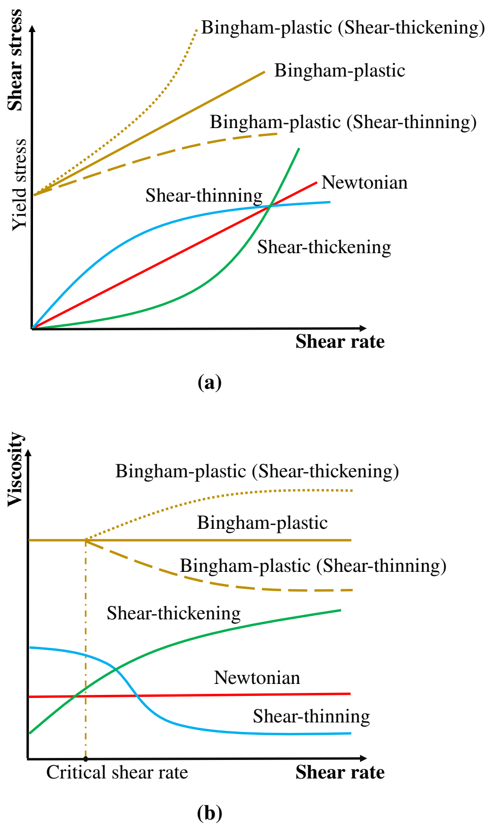


Fig. 44 Diagrams illustrating the relationship between shear rate and shear stress, as well as shear rate and viscosity, for various highly viscous fluids, with design inspiration taken from [293]. (a) Plots of shear stress versus shear rate for different highly viscous fluids. (b) Plots of viscosity versus shear rate for different highly viscous fluids.

Newtonian fluids Following Newton's viscosity law, the viscosity of a Newtonian fluid (incompressible and isotropic) can be expressed by a material parameter μ called dynamic viscosity. Using this, the inner viscous stress tensor $\mathbf{T}_{\text{vis}}$ is computed as

$$\mathbf{T}_{\text{vis}} = 2\mu\mathbf{E}, \tag{31}$$

where $\mathbf{E}$ is a symmetric strain rate tensor that describes the shear strain rate. This equation indicates that the viscous stresses of Newtonian fluids are linearly correlated to the local strain rate at every point. Using the spatial derivatives of the velocity field, the strain rate tensor $\mathbf{E}$ can be defined as

$$\mathbf{E} = \frac{1}{2}(\nabla\mathbf{u} + (\nabla\mathbf{u})^T). \tag{32}$$

Substituting Eqn. (31) and Eqn. (32) into the viscosity force field formulation $\mathbf{f}_{\text{vis}} = \nabla \cdot \mathbf{T}_{\text{vis}}$ and adding the incompressible condition $\nabla \cdot \mathbf{u} = 0$, the viscosity force field $\mathbf{f}_{\text{vis}}$ can be computed as

$$\mathbf{f}_{\text{vis}} = \mu\nabla^2\mathbf{u}. \tag{33}$$

This gives the viscosity term in the momentum equation, corresponding to the viscosity term in the form of kinematic viscosity in Eqn. (6).

Discretizing the viscosity term is a challenging problem for SPH-based methods. Takahashi *et al.* [294] proposed an implicit Euler integration to solve the viscosity term separately, using two SPH first derivatives to discretize the strain tensor and the divergence of the stress tensor, respectively. This method supports a larger range of viscosity and time step values, but a second-ring neighbor computation is required, which impacts efficiency. Peer and Teschner [295] decomposed the velocity gradient into three tensors: spin rate, expansion rate, and shear rate. A user-defined viscosity parameter modifies the shear component, which describes the dissipation due to viscosity. This leads to a target velocity gradient that is used to reconstruct the final velocity field with a first-order Taylor approximation. Because the velocity gradient field is decomposed, the linear system of the velocity field can be solved separately. However, shear viscosity does physically affect the rotation component in the velocity gradient because of the tangential component in rotation. Peer *et al.* [296] extended this method using a vorticity diffusion scheme. The spin rate tensor in the target velocity gradient is modified by solving a vorticity diffusion process, which uses the viscosity parameter in [295] such that vorticity damping is introduced to achieve more realistic effects. Instead of using the strain rate, Weiler *et al.* [297] introduced an implicit viscosity solver based on the Laplacian of the velocity field in Eqn. (33). With a symmetric form of the approximation discretization of the viscosity term [298], an implicit linear system for a new velocity field can be obtained.

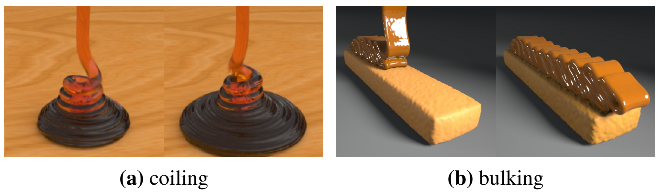


Fig. 45 Coiling and bulking effects of highly viscous fluids [299].

Fig. 46 Changing the viscosity to achieve the effect of different viscous fluids, such as cream, jam, and chocolate sauce [300].

The above-mentioned solvers separate the solving of pressure and viscosity, which reduces accuracy and cannot generate free surface details. Larinov *et al.* [299] introduced a unified pressure–viscosity solver based on implicit variational unsteady Stokes flow problems for grid-based methods, where the inertial force is considered to improve accuracy and achieve a wider viscosity range (Fig. 45). Combining the semi-implicit equation of correlation pressure (SIMPLE) method with SPH, Liu *et al.* [300] used the result of the pressure Poisson equation to improve the pressure in the viscosity solver in an iterative process, which converges to a globally optimal solution (Fig. 46).

Even with a stable solver, mimicking the viscosity coefficient of the target fluid is essential to realistically simulate highly viscous fluids. Takahashi and Lin [99] proposed a framework to find the required viscosity parameter from real videos of highly viscous fluids by minimizing an objective function that evaluates the difference between the silhouettes extracted from video frames and those obtained from the simulation.

Non-Newtonian fluids Such fluids do not follow Newton's viscosity law but rather show a non-linear relation between shear stress and strain rate. For example, the viscosity of a shear-thickening or dilatant fluid (*e.g.*, starch paste) increases when the shear rate increases; the opposite happens for a shear-thinning or pseudoplastic fluid (*e.g.*, ketchup). Some non-Newtonian fluids have properties of solids, such as Bingham plastic fluids like toothpaste. Hence, it is impossible to use a constant viscosity for non-Newtonian fluids, and appropriate constitutive models are required to simulate such fluids.

The Carreau–Yasuda model is a well-known method to simulate non-Newtonian fluids by defining a shear-rate-related viscosity μ as

$$\mu = \mu_\infty + (\mu_0 - \mu_\infty)[1 + (R\dot{\epsilon})^a]^{(n-1)/a}, \quad (34)$$

where μ_0 is the zero-shear viscosity, μ_∞ is the infinite viscosity, $\dot{\epsilon}$ is the shear rate, R is the relaxation time that scales the shear rate, n is a power law index, and a models the transition smoothness between the Newtonian plateau and power law regime. When $n = 1$, this model becomes a Newtonian fluid with dynamic viscosity μ_0. For shear-thinning fluids ($n < 1$), as the strain rate increases, viscosity will vary from μ_0 to μ_∞. For shear-thickening fluids ($n > 1$), the viscosity increases as the shear rate increases.

Bingham plastic fluids are another typical non-Newtonian fluid that behaves as a rigid body at low stress but flow as a Newtonian viscous fluid once the yield stress is exceeded. For many viscoplastic fluids, the stress curve of the flowing part is nonlinear when the shear rate exceeds a critical value. To capture shear thickening/thinning, μ is modeled by the Herschel–Bulkley model given by

$$\mu(\dot{\epsilon}) = \begin{cases} k\dot{\epsilon}^{n-1} + \sigma_y^0 \dot{\epsilon}^{-1}, & \dot{\epsilon} > \dot{\epsilon}_0 \\ \mu_0, & \dot{\epsilon} \leqslant \dot{\epsilon}_0 \end{cases} \quad (35)$$

where k is the consistency coefficient, σ_y^0 is the yield stress, and $\dot{\epsilon}_0$ is the critical shear rate. Similar to Eqn. (34), when $n = 1$, this model describes the ideal Bingham plastic $n > 1$ models shear thickening Bingham plastic, and $n < 1$ models the shear thinning Bingham plastic fluid.

Recent work used the above two models to simulate non-Newtonian fluids. Zhu *et al.* [293] simulated various co-dimensional features of different non-Newtonian fluids, *e.g*, shear thinning and thickening for Bingham plastics, and elasto-plastics. The Carreau–Yasuda model for non-Newtonian fluids was used on a multi-level-set model; semi-implicit methods were used for elasticity and variable viscosity. On the rims of thin fluid sheets, viscosity had an improved treatment to yield twisting motion. Yue *et al.* [301] used the non-Newtonian Herschel–Bulkley model to simulate dense foams composed of microscopic bubbles using MPM. They also proposed a particle resampling method for MPM and a tearing model to simulate tearing/connectivity recovery by explicitly handling the weakening regions detected from space. Mixtures of non-Newtonian fluids were studied by Nagasawa *et al.* [302]. Using the Herschel–Bulkley model, a nonlinear blending model that satisfies the five blending laws [303], along with mass conservation, was proposed to capture non-Newtonian fluid mixture behavior. For viscoelastoplastic materials, a constraint-based method [304] extended position-based dynamics to simulate elastoplastic and highly viscous fluids by recasting a constitutive model of viscoelasticity, which defined governing equations for a conforming tensor.

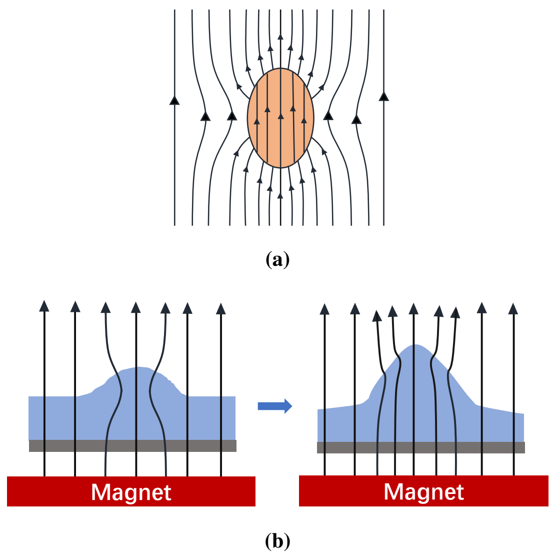


Fig. 47 Schematic representation of ferrofluids in interaction with a magnetic field. (a) Superposition of a uniform vertical magnetic field and the magnetic field originating from an ellipsoid magnetization. There is a pronounced discontinuity in the magnetic field on the surface, most significantly at the extremities where there is a sharp escalation in the field strength. (b) A surface perturbation in the magnetized ferrofluid, resulting in a localized concentration of magnetic induction lines. The ferrofluid is drawn towards this bump due to a heightened field strength, which consequently enhances the gradient, amplifying the perturbation and leading to the formation of a spike.

10.2 Ferrofluids

The dynamic interactions of ferrofluids — liquid media responsive to external magnetic fields — have emerged as an area of considerable interest within the computer graphics research community. These magnetically active fluids, originally conceived by NASA to facilitate fuel transfer in spacecraft under microgravity conditions, derive their magnetic properties from the incorporation of nanoscale magnetic particles. Upon exposure to an external magnetic field, these dispersed particles within the ferrofluid polarize, thereby generating a distinct internal magnetic field. This induced magnetic field, working synergistically with the external one, is pivotal to the magnetization process of the ferrofluid, as shown in Fig. 47. The computer graphics simulation of these captivating fluids, however, had not received significant attention until the very recent pioneering work of Michels *et al.*, which directed attention towards this specialized field [305]. We reference several key insights from their contributions in the subsequent discussion.

The interrelationships of these spatial magnetic fields, both internally produced and externally imposed, conform to the well-established principles outlined in Maxwell's equations:

$$\begin{aligned} \nabla \cdot \mathbf{B} &= 0, \\ \nabla \times \mathbf{H} &= \mathbf{J} + \frac{\partial \mathbf{D}}{\partial t}, \end{aligned} \tag{36}$$

where $\mathbf{B}$ is the magnetic flux density describing the spatial magnetic field, $\mathbf{H}$ is the total magnetic field intensity, $\mathbf{J}$ is the free current density, and $\mathbf{D}$ is the electric displacement field. The ferrofluids further discussed in this section comply with the following model:

$$\begin{aligned} \mathbf{B} &= \mu_0^E(\mathbf{H} + \mathbf{M}^E) \\ \mathbf{H} &= \mathbf{H}_{\text{ext}} + \mathbf{H}_{\text{int}}, \end{aligned} \tag{37}$$

where μ_0^E is the vacuum permeability, $\mathbf{M}^E$ is the magnetization field describing the density of the magnetic moment induced by the total magnetic field, $\mathbf{H}_{\text{ext}}$ is the external magnetic field, and $\mathbf{H}_{\text{int}}$ is the internal magnetic field generated by the ferrofluid. The magnetization field is a function of the known external magnetic field and internal magnetic field, and the current internal magnetic field can be obtained by solving Poisson's equation with the assumptions of $\nabla \times \mathbf{H}_{\text{ext}} = \mathbf{J}$ and $\frac{\partial \mathbf{D}}{\partial t} = 0$ in Eqn. (36), as the free current density $\mathbf{J}$ is not influenced by $\mathbf{H}_{\text{int}}$, and the electric displacement produced by the flow of ferrofluid is not strong enough to influence the system.

Under the effect of magnetic force, fluid particles gather on tiny bumps near the surface where the synthesized magnetic field is stronger and pull the fluid to form spikes with attractive visuals. The spike shapes are also influenced by gravity and surface tension.

The past few years have witnessed the proposal of various methods for simulating ferrofluids within the field of computer graphics. Huang *et al.* [305] presented the smooth magnets method, which utilizes Lagrangian particles embedded with magnetic nanoparticles to discretize fluids, as shown in Fig. 48. This method was the first in the field of computer graphics to address the first-principle-based macroscopic simulation of ferrofluids. Their proposed magnetization model, along with the magnetic field's Poisson equation, can be discretized using a smooth kernel function akin to that used in SPH.

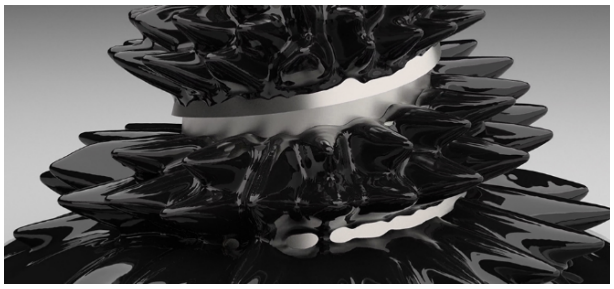


Fig. 48 A ferrofluid climbs up a steel helix under a strong external upward magnetic field to create surface spikes [305].

Approaching from a Lagrangian perspective, the volumetric Kelvin force model is utilized to characterize the magnetic force interactions between particles. Standard Smoothed Particle Hydrodynamics (SPH) computations employ particles, enabling the enforcement of incompressibility and surface tension within ferrofluids. However, the implementation of a Kelvin force model engenders an unanticipated outward-directed force on the surface, prompting particles to exhibit levitation near this surface. To address this phenomenon, Shao *et al.* [306] introduced a modification, replacing the Kelvin force model with a current loop model, thereby creating an inward force. This alteration permitted the integration of the magnetic model into Implicit SPH models. Consequently, it enhanced system stability and facilitated the use of larger time-step increments.

From an Eulerian perspective, Ni *et al.* [307] presented a level-set method for simulating various magnetic bodies, including ferrofluids. The interplay between the magnetic field and mechanical system was addressed as an interfacial issue, and a weighted average of the internal and external magnetic fields was calculated to manage discontinuities. The resulting magnetic force, coupled with surface tension, was integrated into the Navier–Stokes equations to direct the dynamics of the ferrofluids.

Advancing the research further, Huang and Michels [308] introduced the concept of surface-only ferrofluids, a singular study to date, tested successfully against real ferrofluids. Unlike previous approaches that incorporated magnetic force as an additional term in the momentum equation, their method infuses the discontinuity of magnetic pressure into the Dirichlet boundary condition within the pressure-projection of the Galerkin Boundary Element Method (BEM)-based surface-only liquid solver [183], specifically at the fluid–air interface. This pioneering surface scheme enhances the Helmholtz decomposition step in surface-only fluid solvers via a more precise analytic integration process. Additionally, it augments the accuracy of the pressure projection step within surface-only fluid solvers through a Galerkin BEM.

From the hybrid discretization perspective, Sun *et al.* [309] utilized the MPM structure to further simulate nonlinear magnetic substances in pursuit of a more general magnetic description. They used the physically-realistic Langevin's nonlinear magnetization model to bound the magnetic force between magnetic micro elements without additional numerical approximation. Following the concept of MPM, this method uses particles to carry microscopic magnetic quantities and solves Poisson's equation of the magnetic fields and Kelvin force on Cartesian grids. Without integrating the surface tension, this method cannot form stable spikes when simulating ferrofluids. However, thanks to the versatility of MPM, it can achieve a unified simulation and coupling of different magnetic materials.

10.3 Thin films

Thin films and bubbles are fascinating phenomena that have received special attention. A common example is a soap bubble floating in air. Bubbles produced from pure water are usually few, small, and disappear quickly due to gravity, pressure, and strong surface tension. To produce more, larger, and longer-lasting bubbles, surfactants are added to the water, *e.g.*, fatty acids common in soaps. With surfactants interspersed among water molecules, surface tension is reduced so that larger bubbles appear. The tensile deformation of the film will recover due to the difference in surface tension working like elasticity — the so-called Marangoni effect (Fig . 49). While the Marangoni effect models the resilience given by inconsistent surface tension, the Young–Laplace equation describes the capillary pressure difference Δp caused by surface tension between air and the fluid as

$$\Delta p = -\gamma \nabla \cdot \mathbf{n}, \tag{38}$$

where γ is the surface tension, and $\mathbf{n}$ is the surface outward-pointing normal. Eqn. (38) relates the pressure difference to film shape.

Batty *et al.* [310] developed discrete viscous sheets by building on (Lagrangian) elastic thin shells. This reduced-dimensional technique describes the sheets using triangular meshes with local thickness and used the area-based surface tension derived from the mid-surface of the (thin) shell. Wang *et al.* [311] enhanced this to capture the surface tension flow using moving-least-squares particles. The mixed Lagrangian–Eulerian approach models not only volumetric phenomena (3D) but also those arising from thin shells (2D), filaments (1D), and even individual points (0D). Surface tension (and other forces) are handled in a unified way across all (co)dimensions, including codimensional transitions. This enables complex scenarios requiring careful (surface) tension treatment, such as two water jets colliding and forming a thin sheet, to be simulated. Similarly to [311], the codimensional surface tension flow of Zhu *et al.* [312] also relies on simplicial complexes and transitions between elements of different (co)dimensions, covering thin fluid sheets, filaments, and surface tension effects.

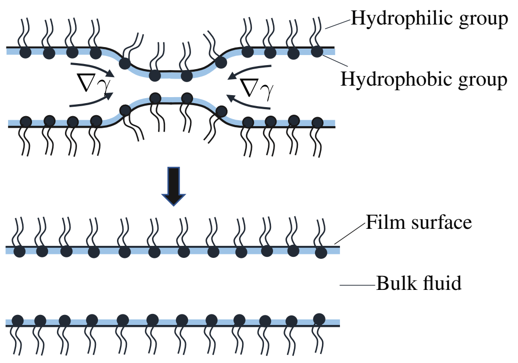


Fig. 49 Schematic diagram of thin film. Surfactants gather on both sides of the film. The tensile deformation yields an inconsistent surfactant concentration, which leads to the surface tension gradient shown in the figure. The film recovers its shape due to this gradient.

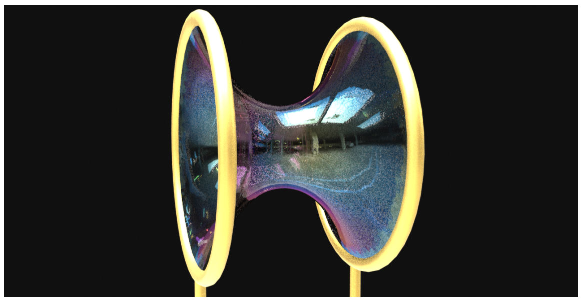


Fig. 50 Film catenoid formed between two rings connected with soap film [313].

Wang *et al.* [313] extended the work of [311] to yield a thin-film SPH fluid model (Fig. 50). Films are modeled as (surface) particles of codimension one with local thickness estimates. This particle setup interacts with a Lagrangian flow simulation using a thin film shape description. This physically couples aggressive surface deformations and strong tangential flows. A process of transformation from codimension one to codimension zero is used to simulate rupture.

Focusing on viscous thin films, Vantzos *et al.* [314] proposed a numerical scheme to simulate the thin film equation (modeled as a height function) on a planar domain, including gravity and other forces. They added a novel quadratic term to the governing equation to stabilize flow while maintaining visual fidelity. Their scheme is fully local; thus, it allows an efficient GPU implementation that leads to real-time simulations, such as that of honey flowing through honeycombs.

Da *et al.* [315] studied soap films and foams whose dynamics are captured by a Lagrangian vortex sheet model with an emphasis on circulation. Surfaces are represented using multi-material triangular meshes supporting topological changes, and their tension forces lead to a circulation update rule based on mean curvature.

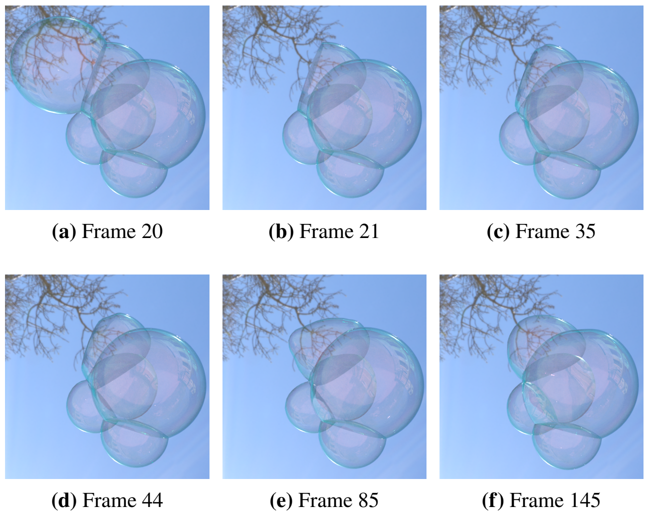

(a) Frame 20 **(b)** Frame 21 **(c)** Frame 35 **(d)** Frame 44 **(e)** Frame 85 **(f)** Frame 145

Fig. 51 Cluster of bubbles [316]. After the top-left bubble bursts at frame 21, the geometry of the remaining bubbles gradually transitions to the next equilibrium state, following Plateau's laws.

By Plateau's laws, a steady-state film consists of constant mean curvature parts and minimal surfaces (vanishing mean curvature). Ishida *et al.* [316] used this to model evolving foams via hyperbolic geometric flow, a type of mean curvature flow (Fig. 51).

Most existing soap film models [315, 316] assume that a film is infinitesimally thin and has no influence on its evolution. Ishida *et al.* [317] extended such methods to use film thickness, modeled on non-manifold meshes, as a reduced degree of freedom in the Navier–Stokes equations and derive the motion equations. This provides an incompressible fluid solver for 2.5D films.

In addition to dynamic effects, bubbles also produce rich color effects due to the light interference caused by uneven film thickness. Besides soap bubbles, other fluid dynamics simulations are also possible on a thin film for additional visual effects.

Two further methods focused on *spherical bubbles* and the fluid around them. Hill and Henderson [318] efficiently simulated fluids on a spherical surface. They handled poles/singularities of spherical coordinates, which would otherwise render the motion equations complex if used naively on the sphere. Their method also enables vector and scalar controls for art-directed spherical flows. Huang *et al.* [319] focused on the chemical–mechanical simulation of soap film flows on spherical bubbles using lubrication theory. Considering the Marangoni effect and the capillary pressure difference, the stress condition at the film surface is given by

$$\mathbf{T_c} \cdot \mathbf{n} = (-\gamma \nabla \cdot \mathbf{n} - p_a)\mathbf{n} + \nabla_s \gamma, \tag{39}$$

where $\mathbf{T_c}$ is Cauchy's stress tensor, $\mathbf{n}$ is the outward unit normal vector at the surface, p_a is the air pressure, and ∇_s is the 2D gradient operator. The surface tension γ is defined by a linear model:

$$\gamma(\Gamma) = \gamma_0 - \gamma_r \Gamma, \tag{40}$$

where γ_0 is the surface tension of pure water, Γ is the surfactant concentration, and γ_r is a constant that describes the Marangoni elasticity of the film. The surfactant concentration Γ is advected by an advection-diffusion equation. Huang *et al.*'s advection scheme used spherical coordinates, using local frames removed some artifacts of [318]. This last method also proposed a physically accurate shader for real-time rendering under environmental lighting.

Recently, Deng *et al.* [320] introduced MELP (Moving Eulerian–Lagrange Particles), a novel mesh-free method for incompressible fluids on moving foams and thin films. Their approach, including multi-MELP for interfacial flow, is able to model both large-scale surface deformations and detailed flows.

11 Conclusion

Physics-based fluid simulation has been successful in games, film, and animation. Current advances enable high levels of control in production and have shown increasing acceptance and potential in virtual and augmented reality and other real-time graphics-intensive applications. Recent advances in physics-based fluid simulation methods rely on a complex mix of computational efficiency, realism, controllability, and ability to simulate diverse scenarios. This survey presented an in-depth overview of these methods over the last decade. We discussed the different goals in this field, techniques proposed to address these goals, and challenges of these techniques.

Our survey found seven major themes present in approximately 300 fluid simulation papers from the computer graphics community in the last decade: advanced computational approaches, interaction with materials, multiphase simulations, gas–liquid interfaces, enhancing fine details, simulation control, and special fluids. These themes formed the structure of our survey to outline important developments in this period and community. We also surveyed existing implementations used to compare and assess the quality of fluid simulations of various types; see also Appendix A.

Several open challenges and directions for future work have emerged while analyzing the above seven themes, as outlined next.

Advanced computational approaches. Adaptive solutions and GPU parallelization are ubiquitous in computer graphics, but there is much room for developing such new methods for physics-based fluids. For instance, adaptivity usually makes an implementation complex and difficult to apply to GPU hardware. Spatial and temporal resolution are often related aspects. In space discretization, interactions between grids or particles at different scales may cause instability and fidelity loss, which can also create energy diffusion. Hence, in addition to improving computational efficiency and reducing overhead, proposing methods to reduce such unwanted effects on a wide range of resolution scales is important for future research. Using neural networks to learn fluid dynamic behavior will be a hot research topic in the future, as it has strong prospects for real-time simulation and industrial control. However, only summarizing physical laws from a large amount of training data lacks underlying logical support. As such, more attention is likely to be paid next to how to inject physical prior knowledge into deep learning models.

Fluid coupling with multi-materials. The key challenge for such simulations is to keep accuracy, stability, and efficiency when coupling multiple materials in one scenario at one time. Using different solvers *vs* materials limited the diversity of fluid animation in the past. Current research has moved from merging multiple solvers to developing monolithic ones. Monolithic solvers can simulate different materials and their interaction in a single framework, which can eliminate stability issues. However, such solvers currently demand high computational resources. Thus, an open challenge is to mix hardware and algorithm designs to better support such solvers.

Multiphase liquids. Future work can explore many interesting aspects. One challenge is that current methods for incompressible fluid simulation do not handle high-density ratios well. Linear systems become ill-conditioned under high-density contrasts, and Jacobi-like solvers fail to converge. Currently, many parameters of mixed-fluid are manually adjusted. How to control parameters more intuitively to achieve the desired visual results is worth further study. Modeling temperature, chemical reactions, elasticity blending, and optical blending are equally important open aspects in this area.

Liquid–gas interaction. Challenges for liquid–gas interaction include simulating the gas–liquid phase transition and modeling its effect on surface tension, supporting non-manifold thin film structure, handling the transition between different codimensionalities, producing realistic surface colors for bubbles, reducing the simulation complexity while preserving accuracy for liquid–air coupling, and adding fine splash details that are not limited by particle size. Achieving richer gas–liquid interaction phenomena while ensuring stability and efficiency is a subject of ongoing and future long-term research.

Fine detail enhancement. Many existing detail enhancement techniques can achieve detailed fluid surfaces without using high-resolution discretization. Realistic and fine-grained appearance representations are the primary pursuit in this direction. Energy conservation and detail preservation in accordance with physical laws are also important goals. How to improve the efficiency and scalability of detail enhancement is an open challenge for the end goal of providing real-time, interactive, and generalizable tools for artists. Future work will likely use deep learning techniques with innovative explorations in style transfer, high-resolution reconstruction, and detail generation.

Fluid control. Issues still exist in fluid control: high memory and computation costs, sensitive parameters, and manual feature labeling. A key difficulty of control methods is to achieve precise control. Achieving precise control, along with high accuracy and efficient computation, is — regarding the other topics studied in this survey — one of the grand challenges of fluid simulation research.

Special fluids. The challenge of simulating fluids with special forms lies in unifying governing equations and other physical characteristics in a consolidated system, which means integrating different solvers together. The targeted formulation of a monolithic-style solver is required to perform high-performance simulation results.

We hope this survey helps to introduce the theoretical concepts underpinning physics-based fluid simulation and their practical implementation to serve as a guide for researchers and practitioners as well as facilitate future works to exploit on the basis of recent developments.

A Practical resources

For beginners, how to quickly build their own fluid models and achieve convincing rendering results may be of the most concern and interest. In this appendix, we recommend several popular libraries, frameworks, and other software tools to help beginners get their foot in the door and get excited about learning. In addition, these resources are also useful to benchmark third-party fluid simulations.

A.1 Focus on simulation:

OpenMaelstrom is an open-source library for the simulation and rendering of fluids based on the SPH method. It provides many pressure solvers (IISPH, DFSPH, and Interlinked SPH for strong fluid-rigid coupling) and boundary handling methods. Moreover, it features spatial adaptation and full GPU support.

MantaFlow includes a wide range of Navier–Stokes solver variants, and its parallelized C++ solver core, Python scene definition interface, and plugin system allow for quick prototyping and testing of new algorithms. An advantage of *MantaFlow* is that it can be easily integrated into Blender for end-to-end fluid simulation and rendering. In addition, *MantaFlow* can be coupled with TensorFlow, which helps the development of traditional physical simulation combined with deep learning techniques.

PhysIKA (Physics-based Interactive Kinematics Architecture) is a node-based architecture targeted at real-time simulation of versatile physical materials. Currently, it supports the simulation of a wide range of physical phenomena, including fluids, elastic objects, and fractures. *PhysIKA* is highly modularized and can also help the research community develop novel algorithms.

PositionBasedDynamics supports the physically-based simulation of mechanical effects for elastic rods, deformable solids, rigid bodies, and fluids. *SPlisHSPlasH*, by the same author as *PositionBasedDynamics*, simulates complex fluid effects based on SPH methods. It includes several SPH solvers (including WCSPH, PCISPH, IISPH, and DFSPH) and provides different methods to simulate viscosity, surface tension, vorticity, and multiphase fluid interaction.

PhysBAM is a multiphysics simulation library capable of simulating rigid and deformable bodies, compressible and incompressible fluids, coupled solids and fluids, fracture, fire, smoke, hair, cloth, muscles, and many other natural phenomena. The *PhysBAM* library has a component called OpenGL Viewer, which displays and analyzes 1D, 2D, and 3D simulation data generated from *PhysBAM* projects to facilitate the fast verification and debugging of simulation methods.

SOFA is an open-source framework targeting real-time simulation, with an emphasis on medical simulation. It is based on approximately 15 years of research in physics simulation. *SOFA* is being used in many different projects, such as solid mechanics for the simulation of brain, ear, bones, heart, and liver, as well as fluid dynamics for the simulation of fat filling and blood flow in aneurysms.

FleX is a particle-based simulation technique for real-time visual effects. It uses a unified particle representation for all object types and enables new effects in which different simulated substances can interact with each other seamlessly. It supports the simulation of rigid bodies, deformable objects, phase transitions, fluids, gases, and other phenomena. The goal of *FleX* is to use the power of GPUs to bring the capabilities of offline applications to real-time computer graphics.

Realflow is a professional commercial fluid dynamics simulation software. It can be well connected with other 3D software, such as Maya, 3ds MAX, and Cinema 4D. It is powerful enough to simulate the fluid effects typically seen in high-speed macro photography and to simulate viscous liquids, such as cream, chocolate, oil, honey, and many others. Moreover, it allows the simulation of sand, snow, ice, and many other materials. It also provides a highly-optimized CPU and GPU particle solver where different types of materials are simulated within the same framework and are able to interact with each other.

A.2 Focus on modeling and rendering:

Highly accurate fluid simulation results require highly accurate rendering to convey the obtained simulation details. End-to-end systems cover most aspects of a graphics pipeline, including modeling, animation, and actual rendering. Below are some commonly used software for modeling and rendering.

Houdini is a flexible node-based workflow following a dataflow computing model and allowing users to reuse computing nodes or even entire (sub)networks. *Houdini* has its own programming language, VEX, to deal with geometry for free development. The downside of *Houdini* is slow rendering.

Blender is an efficient 3D modeling, rendering, and animation software. Its key advantages are hundreds of open-source add-ons and extensive Python API. Every tool is available for scripting and customization.

Both *MAYA* and *3ds MAX* are Autodesk products that enable modeling, mapping, binding, animation, rendering, and more. *MAYA*'s strength lies in animation and its special effects, mostly used for film and television animation; *3ds MAX* has the advantage of being easy to learn and model with. Both Maya and *3ds MAX* are highly professional, accelerate workflows, and provide stunning visuals.
With the rise of the metaverse concept, the NVIDIA *Omniverse* simulation platform is poised to launch the next wave of digitalization. NVIDIA Omniverse is an extensible open platform built for virtual collaboration and physically accurate real-time simulation. The advantage of *Omniverse* is that it facilitates real-time collaboration between users and applications, simplifying workflows by updating, iterating, and changing in real time without having to prepare data. It could change the way designers around the world collaborate and become the foundation of the metaverse.

Acknowledgements

In this report, we have included several illustrations from previous papers. The authors of these papers have agreed for us to publish these illustrations in this report, which is gratefully acknowledged.

This research was funded by National Natural Science Foundation of China (Nos.62376025, 62332017, U22A2022), Horizon 2020-Marie Sklodowska-Curie Action-Individual Fellowships (No.895941), National Key Research and Development Program of China (No.2022ZD0118001) and Guangdong Basic and Applied Basic Research Foundation (Nos.2023A1515030177, 2022A1515110350, 2021A1515012285).

Declaration of competing interest

The authors have no competing interests to declare that are relevant to the content of this article.

References

[1] Reeves WT, Blau R. Approximate and Probabilistic Algorithms for Shading and Rendering Structured Particle Systems. *SIGGRAPH Comput. Graph.*, 1985, 19(3): 313–322, doi:10.1145/325165.325250.

[2] Stam J. Stable Fluids. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '99, 1999, 121–128, doi:10.1145/311535.311548.

[3] Bridson R. *Fluid Simulation for Computer Graphics*. 2008.

[4] Koschier D, Bender J, Solenthaler B, Teschner M. Smoothed Particle Hydrodynamics Techniques for the Physics Based Simulation of Fluids and Solids. *Eurographics 2019 - Tutorials*, 2019, doi:10.2312/EGT.20191035.

[5] Jiang C, Schroeder C, Teran J, Stomakhin A, Selle A. The Material Point Method for Simulating Continuum Materials. In *ACM SIGGRAPH 2016 Courses*, SIGGRAPH '16, 2016, doi:10.1145/2897826.2927348.

[6] Landau LD, Lifshitz EM. *Fluid Mechanics*, volume 6. 2013.

[7] Sito T. *Moving Innovation: A History of Computer Animation*. 2013.

[8] Wejchert J, Haumann D. Animation Aerodynamics. In *Proceedings of the 18th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '91, 1991, 19–22, doi:10.1145/122718.122719.

[9] Stam J, Fiume E. Turbulent Wind Fields for Gaseous Phenomena. In *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '93, 1993, 369–376, doi:10.1145/166117.166163.

[10] Desbrun M, Gascuel MP. Smoothed Particles: A New Paradigm for Animating Highly Deformable Bodies. In *Proceedings of the Eurographics Workshop on Computer Animation and Simulation '96*, 1996, 61–76, doi:10.1007/978-3-7091-7486-9_5.

[11] Foster N, Metaxas D. Realistic Animation of Liquids. *Graphical Models and Image Processing*, 1996, 58(5): 471–483, doi:10.1006/gmip.1996.0039.

[12] Harlow FH. The particle-in-cell method for numerical solution of problems in fluid dynamics. Technical report, 1962, doi: 10.2172/4769185.

[13] Brackbill J, Ruppel H. FLIP: A method for adaptively zoned, particle-in-cell calculations of fluid flows in two dimensions. *Journal of Computational Physics*, 1986, 65(2): 314–343, doi:10.1016/0021-9991(86)90211-1.

[14] Zhu Y, Bridson R. Animating Sand as a Fluid. *ACM Trans. Graph.*, 2005, 24(3): 965–972, doi:10.1145/1073204.1073298.

[15] Harlow FH, Welch JE. Numerical Calculation of Time-Dependent Viscous Incompressible Flow of Fluid with Free Surface. *The Physics of Fluids*, 1965, 8(12): 2182–2189, doi:10.1063/1.1761178.

[16] Lucy LB. A numerical approach to the testing of the fission hypothesis. *The astronomical journal*, 1977, 82: 1013–1024, doi:10.1086/112164.

[17] Gingold RA, Monaghan JJ. Smoothed particle hydrodynamics: theory and application to non-spherical stars. *Monthly Notices of the Royal Astronomical Society*, 1977, 181(3): 375–389, doi:10.1093/mnras/181.3.375.

[18] Becker M, Teschner M. Weakly Compressible SPH for Free Surface Flows. In *Proceedings of the 2007 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '07, 2007, 209–217.

[19] Solenthaler B, Pajarola R. Predictive-Corrective Incompressible SPH. *ACM Trans. Graph.*, 2009, 28(3), doi: 10.1145/1531326.1531346.

[20] Ihmsen M, Cornelis J, Solenthaler B, Horvath C, Teschner M. Implicit Incompressible SPH. *IEEE Transactions on Vi-*

sualization and Computer Graphics, 2014, 20(3): 426–435, doi:10.1109/TVCG.2013.105.

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

[22] Müller M, Heidelberger B, Hennix M, Ratcliff J. Position Based Dynamics. *J. Vis. Comun. Image Represent.*, 2007, 18(2): 109–118, doi:10.1016/j.jvcir.2007.01.005.

[23] Macklin M, Müller M. Position Based Fluids. *ACM Trans. Graph.*, 2013, 32(4), doi:10.1145/2461912.2461984.

[24] Harlow FH. The particle-in-cell computing method for fluid dynamics. *Methods Comput. Phys.*, 1964, 3: 319–343.

[25] Sulsky D, Zhou SJ, Schreyer HL. Application of a particle-in-cell method to solid mechanics. *Computer Physics Communications*, 1995, 87(1): 236–252, doi:10.1016/0010-4655(94)00170-7, particle Simulation Methods.

[26] Jiang C, Schroeder C, Selle A, Teran J, Stomakhin A. The Affine Particle-in-Cell Method. *ACM Trans. Graph.*, 2015, 34(4), doi:10.1145/2766996.

[27] Fu C, Guo Q, Gast T, Jiang C, Teran J. A Polynomial Particle-in-Cell Method. *ACM Trans. Graph.*, 2017, 36(6), doi:10.1145/3130800.3130878.

[28] Hu Y, Fang Y, Ge Z, Qu Z, Zhu Y, Pradhana A, Jiang C. A Moving Least Squares Material Point Method with Displacement Discontinuity and Two-Way Rigid Body Coupling. *ACM Trans. Graph.*, 2018, 37(4), doi:10.1145/3197517.3201293.

[29] Manteaux PL, Wojtan C, Narain R, Redon S, Faure F, Cani MP. Adaptive Physically Based Models in Computer Graphics. *Computer Graphics Forum*, 2017, 36(6): 312–337, doi:10.1111/cgf.12941.

[30] Koike T, Morishima S, Ando R. Asynchronous Eulerian Liquid Simulation. *Computer Graphics Forum*, 2020, 39(2): 1–8, doi:10.1111/cgf.13907.

[31] Courant R, Friedrichs K, Lewy H. Über die partiellen Differenzengleichungen der mathematischen Physik. *Mathematische annalen*, 1928, 100(1): 32–74, doi:10.1007/BF01448839.

[32] Sun Y, Shinar T, Schroeder C. Effective time step restrictions for explicit MPM simulation. *Computer Graphics Forum*, 2020, 39(8): 55–67, doi:10.1111/cgf.14101.

[33] Goswami P, Batty C. Regional Time Stepping for SPH. In E Galin, M Wand, editors, *Eurographics 2014 - Short Papers*, 2014, doi:10.2312/egsh.20141011.

[34] Fang Y, Hu Y, Hu SM, Jiang C. A Temporally Adaptive Material Point Method with Regional Time Stepping. *Computer Graphics Forum*, 2018, 37(8): 195–204, doi:10.1111/cgf.13524.

[35] Reinhardt S, Huber M, Eberhardt B, Weiskopf D. Fully Asynchronous SPH Simulation. In *Proceedings of the ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, SCA '17, 2017, doi:10.1145/3099564.3099571.

[36] Losasso F, Gibou F, Fedkiw R. Simulating Water and Smoke with an Octree Data Structure. *ACM Trans. Graph.*, 2004, 23(3): 457–462, doi:10.1145/1015706.1015745.

[37] Setaluri R, Aanjaneya M, Bauer S, Sifakis E. SPGrid: A Sparse Paged Grid Structure Applied to Adaptive Smoke Simulation. *ACM Trans. Graph.*, 2014, 33(6), doi:10.1145/2661229.2661269.

[38] Goldade R, Wang Y, Aanjaneya M, Batty C. An Adaptive Variational Finite Difference Framework for Efficient Symmetric Octree Viscosity. *ACM Trans. Graph.*, 2019, 38(4), doi:10.1145/3306346.3322939.

[39] Ando R, Batty C. A Practical Octree Liquid Simulator with Adaptive Surface Resolution. *ACM Trans. Graph.*, 2020, 39(4), doi:10.1145/3386569.3392460.

[40] Shao H, Huang L, Michels DL. A Fast Unsmoothed Aggregation Algebraic Multigrid Framework for the Large-Scale Simulation of Incompressible Flow. *ACM Trans. Graph.*, 2022, 41(4), doi:10.1145/3528223.3530109.

[41] Ferstl F, Westermann R, Dick C. Large-Scale Liquid Simulation on Adaptive Hexahedral Grids. *IEEE Transactions on Visualization and Computer Graphics*, 2014, 20(10): 1405–1417, doi:10.1109/TVCG.2014.2307873.

[42] Aanjaneya M, Gao M, Liu H, Batty C, Sifakis E. Power Diagrams and Sparse Paged Grids for High Resolution Adaptive Liquids. *ACM Trans. Graph.*, 2017, 36(4), doi:10.1145/3072959.3073625.

[43] Xiao Y, Chan S, Wang S, Zhu B, Yang X. An Adaptive Staggered-Tilted Grid for Incompressible Flow Simulation. *ACM Trans. Graph.*, 2020, 39(6), doi:10.1145/3414685.3417837.

[44] Gao Y, Li CF, Ren B, Hu SM. View-Dependent Multiscale Fluid Simulation. *IEEE Transactions on Visualization and Computer Graphics*, 2013, 19(2): 178–188, doi:10.1109/TVCG.2012.117.

[45] English RE, Qiu L, Yu Y, Fedkiw R. Chimera Grids for Water Simulation. In *Proceedings of the 12th ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '13, 2013, 85–94, doi:10.1145/2485895.2485897.

[46] Li W, Bai K, Liu X. Continuous-Scale Kinetic Fluid Simulation. *IEEE Transactions on Visualization and Computer Graphics*, 2019, 25(9): 2694–2709, doi:10.1109/TVCG.2018.2859931.

[47] Zhu B, Lu W, Cong M, Kim B, Fedkiw R. A New Grid Structure for Domain Extension. *ACM Trans. Graph.*, 2013, 32(4), doi:10.1145/2461912.2461999.

[48] Ibayashi H, Wojtan C, Thuerey N, Igarashi T, Ando R. Simulating Liquids on Dynamically Warping Grids. *IEEE Transactions on Visualization and Computer Graphics*, 2020, 26(6): 2288–2302, doi:10.1109/TVCG.2018.2883628.

[49] Adams B, Pauly M, Keiser R, Guibas LJ. Adaptively Sampled Particle Fluids. *ACM Trans. Graph.*, 2007, 26(3): 48–es, doi:10.1145/1276377.1276437.

[50] Orthmann J, Kolb A. Temporal Blending for Adaptive SPH.

Computer Graphics Forum, 2012, 31(8): 2436–2449, doi: 10.1111/j.1467-8659.2012.03186.x.

[51] Winchenbach R, Hochstetter H, Kolb A. Infinite Continuous Adaptivity for Incompressible SPH. *ACM Trans. Graph.*, 2017, 36(4), doi:10.1145/3072959.3073713.

[52] Zhai X, Hou F, Qin H, Hao A. Fluid Simulation with Adaptive Staggered Power Particles on GPUs. *IEEE Transactions on Visualization and Computer Graphics*, 2020, 26(6): 2234–2246, doi:10.1109/TVCG.2018.2886322.

[53] Winchenbach R, Akhunov R, Kolb A. Semi-Analytic Boundary Handling below Particle Resolution for Smoothed Particle Hydrodynamics. *ACM Trans. Graph.*, 2020, 39(6), doi: 10.1145/3414685.3417829.

[54] Winchenbach R, Kolb A. Optimized Refinement for Spatially Adaptive SPH. *ACM Trans. Graph.*, 2021, 40(1), doi:10.114 5/3363555.

[55] Winchenbach R, Hochstetter H, Kolb A. Constrained Neighbor Lists for SPH-based Fluid Simulations. In L Kavan, C Wojtan, editors, *Eurographics/ ACM SIGGRAPH Symposium on Computer Animation*, 2016, doi:10.2312/sca.20161222.

[56] Winchenbach R, Kolb A. Multi-Level Memory Structures for Simulating and Rendering Smoothed Particle Hydrodynamics. *Computer Graphics Forum*, 2020, 39(6): 527–541, doi:10.1111/cgf.14090.

[57] Greengard L, Rokhlin V. A fast algorithm for particle simulations. *Journal of Computational Physics*, 1987, 73(2): 325–348, doi:10.1016/0021-9991(87)90140-9.

[58] Zhang X, Bridson R. A PPPM Fast Summation Method for Fluids and Beyond. *ACM Trans. Graph.*, 2014, 33(6), doi:10.1145/2661229.2661261.

[59] Angelidis A. Multi-Scale Vorticle Fluids. *ACM Trans. Graph.*, 2017, 36(4), doi:10.1145/3072959.3073606.

[60] Nakanishi R, Nascimento F, Campos R, Pagliosa P, Paiva A. RBF Liquids: An Adaptive PIC Solver Using RBF-FD. *ACM Trans. Graph.*, 2020, 39(6), doi:10.1145/3414685.3417794.

[61] Wu K, Truong N, Yuksel C, Hoetzlein R. Fast Fluid Simulations with Sparse Volumes on the GPU. *Computer Graphics Forum*, 2018, 37(2): 157–167, doi:10.1111/cgf.13350.

[62] Chen Y, Li W, Fan R, Liu X. GPU Optimization for High-Quality Kinetic Fluid Simulation. *IEEE Transactions on Visualization and Computer Graphics*, 2022, 28(9): 3235–3251, doi:10.1109/TVCG.2021.3059753.

[63] Ando R, Thurey N, Tsuruno R. Preserving Fluid Sheets with Adaptively Sampled Anisotropic Particles. *IEEE Transactions on Visualization and Computer Graphics*, 2012, 18(8): 1202–1214, doi:10.1109/TVCG.2012.87.

[64] Ando R, Thürey N, Wojtan C. Highly Adaptive Liquid Simulations on Tetrahedral Meshes. *ACM Trans. Graph.*, 2013, 32(4), doi:10.1145/2461912.2461982.

[65] Yue Y, Smith B, Chen PY, Chantharayukhonthorn M, Kamrin K, Grinspun E. Hybrid Grains: Adaptive Coupling of Discrete and Continuum Simulations of Granular Media. *ACM Trans. Graph.*, 2018, 37(6), doi:10.1145/3272127.3275095.

[66] Chentanez N, Müller M, Kim TY. Coupling 3D Eulerian, Heightfield and Particle Methods for Interactive Simulation of Large Scale Liquid Phenomena. *IEEE Transactions on Visualization and Computer Graphics*, 2015, 21(10): 1116–1128, doi:10.1109/TVCG.2015.2449303.

[67] Ferstl F, Ando R, Wojtan C, Westermann R, Thuerey N. Narrow Band FLIP for Liquid Simulations. *Computer Graphics Forum*, 2016, 35(2): 225–232, doi:10.1111/cgf.12825.

[68] Sato T, Wojtan C, Thuerey N, Igarashi T, Ando R. Extended Narrow Band FLIP for Liquid Simulations. *Computer Graphics Forum*, 2018, 37(2): 169–177, doi:10.1111/cgf.13351.

[69] Huang L, Qu Z, Tan X, Zhang X, Michels DL, Jiang C. Ships, Splashes, and Waves on a Vast Ocean. *ACM Trans. Graph.*, 2021, 40(6), doi:10.1145/3478513.3480495.

[70] Museth K. VDB: High-Resolution Sparse Volumes with Dynamic Topology. *ACM Trans. Graph.*, 2013, 32(3), doi: 10.1145/2487228.2487235.

[71] Gao M, Wang X, Wu K, Pradhana A, Sifakis E, Yuksel C, Jiang C. GPU Optimization of Material Point Methods. *ACM Trans. Graph.*, 2018, 37(6), doi:10.1145/3272127.3275044.

[72] Chu J, Zafar NB, Yang X. A Schur Complement Preconditioner for Scalable Parallel Fluid Simulation. *ACM Trans. Graph.*, 2017, 36(5), doi:10.1145/3092818.

[73] Hu Y, Li TM, Anderson L, Ragan-Kelley J, Durand F. Taichi: A Language for High-Performance Computation on Spatially Sparse Data Structures. *ACM Trans. Graph.*, 2019, 38(6), doi:10.1145/3355089.3356506.

[74] Hu Y, Liu J, Yang X, Xu M, Kuang Y, Xu W, Dai Q, Freeman WT, Durand F. QuanTaichi: A Compiler for Quantized Simulations. *ACM Trans. Graph.*, 2021, 40(4), doi: 10.1145/3450626.3459671.

[75] Liu H, Mitchell N, Aanjaneya M, Sifakis E. A Scalable Schur-Complement Fluids Solver for Heterogeneous Compute Platforms. *ACM Trans. Graph.*, 2016, 35(6), doi:10.1145/29 80179.2982430.

[76] Wang X, Qiu Y, Slattery SR, Fang Y, Li M, Zhu SC, Zhu Y, Tang M, Manocha D, Jiang C. A Massively Parallel and Scalable Multi-GPU Material Point Method. *ACM Trans. Graph.*, 2020, 39(4), doi:10.1145/3386569.3392442.

[77] Biddiscombe J, Soumagne J, Oger G, Guibert D, Piccinali JG. Parallel Computational Steering for HPC Applications Using HDF5 Files in Distributed Shared Memory. *IEEE Transactions on Visualization and Computer Graphics*, 2012, 18(6): 852–864, doi:10.1109/TVCG.2012.63.

[78] Mashayekhi O, Shah C, Qu H, Lim A, Levis P. Automatically Distributing Eulerian and Hybrid Fluid Simulations in the Cloud. *ACM Trans. Graph.*, 2018, 37(2), doi:10.1145/3173 551.

[79] Shah C, Hyde D, Qu H, Levis P. Distributing and Load Balancing Sparse Fluid Simulations. *Computer Graphics Forum*, 2018, 37(8): 35–46, doi:10.1111/cgf.13510.

[80] Qu H, Mashayekhi O, Shah C, Levis P. Accelerating Distributed Graphical Fluid Simulations with Micro-partitioning.

Computer Graphics Forum, 2020, 39(1): 375–388, doi: 10.1111/cgf.13809.

[81] Treuille A, Lewis A, Popović Z. Model reduction for real-time fluids. *ACM Transactions on Graphics*, 2006, 25(3): 826–834, doi:10.1145/1141911.1141962.

[82] Stanton M, Sheng Y, Wicke M, Perazzi F, Yuen A, Narasimhan S, Treuille A. Non-Polynomial Galerkin Projection on Deforming Meshes. *ACM Trans. Graph.*, 2013, 32(4), doi: 10.1145/2461912.2462006.

[83] Kim T, Delaney J. Subspace Fluid Re-Simulation. *ACM Trans. Graph.*, 2013, 32(4), doi:10.1145/2461912.2461987.

[84] De Witt T, Lessig C, Fiume E. Fluid Simulation Using Laplacian Eigenfunctions. *ACM Trans. Graph.*, 2012, 31(1), doi:10.1145/2077341.2077351.

[85] Liu B, Mason G, Hodgson J, Tong Y, Desbrun M. Model-Reduced Variational Fluid Simulation. *ACM Trans. Graph.*, 2015, 34(6), doi:10.1145/2816795.2818130.

[86] Zhai X, Hou F, Qin H, Hao A. Inverse Modelling of Incompressible Gas Flow in Subspace. *Computer Graphics Forum*, 2017, 36(6): 100–111, doi:10.1111/cgf.12861.

[87] Cui Q, Sen P, Kim T. Scalable Laplacian Eigenfluids. *ACM Trans. Graph.*, 2018, 37(4), doi:10.1145/3197517.3201352.

[88] Cui Q, Langlois T, Sen P, Kim T. Spiral-Spectral Fluid Simulation. *ACM Trans. Graph.*, 2021, 40(6), doi:10.1145/3478513.3480536.

[89] Mercier O, Nowrouzezahrai D. Local Bases for Model-reduced Smoke Simulations. *Computer Graphics Forum*, 2020, 39(2): 9–22, doi:10.1111/cgf.13908.

[90] Panuelos J, Goldade R, Batty C. Efficient Unified Stokes using a Polynomial Reduced Fluid Model. In DL Michels, editor, *Eurographics/ ACM SIGGRAPH Symposium on Computer Animation - Posters*, 2020, doi:10.2312/sca.20201214.

[91] Ladický L, Jeong S, Solenthaler B, Pollefeys M, Gross M. Data-Driven Fluid Simulations Using Regression Forests. *ACM Trans. Graph.*, 2015, 34(6), doi:10.1145/2816795.2818129.

[92] Raveendran K, Wojtan C, Thuerey N, Turk G. Blending Liquids. *ACM Trans. Graph.*, 2014, 33(4), doi:10.1145/2601097.2601126.

[93] Thuerey N. Interpolations of Smoke and Liquid Simulations. *ACM Trans. Graph.*, 2016, 36(1), doi:10.1145/2956233.

[94] Oh YJ, Lee IK. Two-step Temporal Interpolation Network Using Forward Advection for Efficient Smoke Simulation. *Computer Graphics Forum*, 2021, 40(2): 355–365, doi:10.1111/cgf.142638.

[95] Gao Y, Zhang Q, Li S, Hao A, Qin H. Accelerating Liquid Simulation With an Improved Data-Driven Method. *Computer Graphics Forum*, 2020, 39(6): 180–191, doi:10.1111/cgf.14010.

[96] Xiao X, Zhou Y, Wang H, Yang X. A Novel CNN-Based Poisson Solver for Fluid Simulation. *IEEE Transactions on Visualization and Computer Graphics*, 2020, 26(3): 1454–1465, doi:10.1109/TVCG.2018.2873375.

[97] Wiewel S, Becher M, Thuerey N. Latent Space Physics: Towards Learning the Temporal Evolution of Fluid Flow. *Computer Graphics Forum*, 2019, 38(2): 71–82, doi:10.1111/cgf.13620.

[98] Wiewel S, Kim B, Azevedo VC, Solenthaler B, Thuerey N. Latent Space Subdivision: Stable and Controllable Time Predictions for Fluid Flow. *Computer Graphics Forum*, 2020, 39(8): 15–25, doi:10.1111/cgf.14097.

[99] Takahashi T, Lin MC. Video-Guided Real-to-Virtual Parameter Transfer for Viscous Fluids. *ACM Trans. Graph.*, 2019, 38(6), doi:10.1145/3355089.3356551.

[100] Eckert ML, Um K, Thuerey N. ScalarFlow: A Large-Scale Volumetric Data Set of Real-World Scalar Transport Flows for Computer Animation and Machine Learning. *ACM Trans. Graph.*, 2019, 38(6), doi:10.1145/3355089.3356545.

[101] Becker M, Tessendorf H, Teschner M. Direct Forcing for Lagrangian Rigid-Fluid Coupling. *IEEE Transactions on Visualization and Computer Graphics*, 2009, 15(3): 493–503, doi:10.1109/TVCG.2008.107.

[102] Yang L, Li S, Hao A, Qin H. Realtime Two-Way Coupling of Meshless Fluids and Nonlinear FEM. *Computer Graphics Forum*, 2012, 31(7): 2037–2046, doi:10.1111/j.1467-8659.2012.03196.x.

[103] Schechter H, Bridson R. Ghost SPH for Animating Water. *ACM Trans. Graph.*, 2012, 31(4), doi:10.1145/2185520.2185557.

[104] He X, Liu N, Wang G, Zhang F, Li S, Shao S, Wang H. Staggered Meshless Solid-Fluid Coupling. *ACM Trans. Graph.*, 2012, 31(6), doi:10.1145/2366145.2366168.

[105] Akinci N, Ihmsen M, Akinci G, Solenthaler B, Teschner M. Versatile Rigid-Fluid Coupling for Incompressible SPH. *ACM Trans. Graph.*, 2012, 31(4), doi:10.1145/2185520.2185558.

[106] Macklin M, Müller M, Chentanez N, Kim TY. Unified Particle Physics for Real-Time Applications. *ACM Trans. Graph.*, 2014, 33(4), doi:10.1145/2601097.2601152.

[107] Cornelis J, Ihmsen M, Peer A, Teschner M. IISPH-FLIP for incompressible fluids. *Computer Graphics Forum*, 2014, 33(2): 255–262, doi:10.1111/cgf.12324.

[108] Peer A, Gissler C, Band S, Teschner M. An Implicit SPH Formulation for Incompressible Linearly Elastic Solids. *Computer Graphics Forum*, 2018, 37(6): 135–148, doi: 10.1111/cgf.13317.

[109] Takahashi T, Lin MC. A Multilevel SPH Solver with Unified Solid Boundary Handling. *Computer Graphics Forum*, 2016, 35(7): 517–526, doi:10.1111/cgf.13048.

[110] Takahashi T, Dobashi Y, Nishita T, Lin MC. An Efficient Hybrid Incompressible SPH Solver with Interface Handling for Boundary Conditions. *Computer Graphics Forum*, 2018, 37(1): 313–324, doi:10.1111/cgf.13292.

[111] Shao X, Zhou Z, Magnenat-Thalmann N, Wu W. Stable and Fast Fluid–Solid Coupling for Incompressible SPH. *Computer Graphics Forum*, 2015, 34(1): 191–204, doi:10.1111/cgf.12467.

[112] Band S, Gissler C, Ihmsen M, Cornelis J, Peer A, Teschner M. Pressure Boundaries for Implicit Incompressible SPH. *ACM Trans. Graph.*, 2018, 37(2), doi:10.1145/3180486.

[113] Gissler C, Peer A, Band S, Bender J, Teschner M. Interlinked SPH Pressure Solvers for Strong Fluid-Rigid Coupling. *ACM Trans. Graph.*, 2019, 38(1), doi:10.1145/3284980.

[114] Truong N, Yuksel C, Watcharopas C, Levine JA, Kirby RM. Particle Merging-and-Splitting. *IEEE Transactions on Visualization and Computer Graphics*, 2021: 1–1, doi:10.1109/TVCG.2021.3093776.

[115] Vines M, Houston B, Lang J, Lee WS. Vortical Inviscid Flows with Two-Way Solid-Fluid Coupling. *IEEE Transactions on Visualization and Computer Graphics*, 2014, 20(2): 303–315, doi:10.1109/TVCG.2013.95.

[116] Fujisawa M, Miura KT. An Efficient Boundary Handling with a Modified Density Calculation for SPH. *Computer Graphics Forum*, 2015, 34(7): 155–162, doi:10.1111/cgf.12754.

[117] Chang Y, Liu S, He X, Li S, Wang G. Semi-analytical Solid Boundary Conditions for Free Surface Flows. *Computer Graphics Forum*, 2020, 39(7): 131–141, doi:10.1111/cgf.14132.

[118] Koschier D, Bender J. Density Maps for Improved SPH Boundary Handling. In *Proceedings of the ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, SCA ’17, 2017, doi:10.1145/3099564.3099565.

[119] Bender J, Kugelstadt T, Weiler M, Koschier D. Implicit Frictional Boundary Handling for SPH. *IEEE Transactions on Visualization and Computer Graphics*, 2020, 26(10): 2982–2993, doi:10.1109/TVCG.2020.3004245.

[120] Clausen P, Wicke M, Shewchuk JR, OB́rien JF. Simulating Liquids and Solid-Liquid Interactions with Lagrangian Meshes. *ACM Trans. Graph.*, 2013, 32(2), doi:10.1145/2451236.2451243.

[121] Azevedo VC, Oliveira MM. Efficient Smoke Simulation on Curvilinear Grids. *Computer Graphics Forum*, 2013, 32(7): 235–244, doi:10.1111/cgf.12231.

[122] Teng Y, Levin DIW, Kim T. Eulerian Solid-Fluid Coupling. *ACM Trans. Graph.*, 2016, 35(6), doi:10.1145/2980179.2980229.

[123] Takahashi T, Lin MC. A Geometrically Consistent Viscous Fluid Solver with Two-Way Fluid-Solid Coupling. *Computer Graphics Forum*, 2019, 38(2): 49–58, doi:10.1111/cgf.13618.

[124] Chentanez N, Mueller-Fischer M. A Multigrid Fluid Pressure Solver Handling Separating Solid Boundary Conditions. *IEEE Transactions on Visualization and Computer Graphics*, 2012, 18(8): 1191–1201, doi:10.1109/TVCG.2012.86.

[125] Weber D, Mueller-Roemer J, Stork A, Fellner D. A Cut-Cell Geometric Multigrid Poisson Solver for Fluid Simulation. *Computer Graphics Forum*, 2015, 34(2): 481–491, doi:10.1111/cgf.12577.

[126] Azevedo VC, Batty C, Oliveira MM. Preserving Geometry and Topology for Fluid Flows with Thin Obstacles and Narrow Gaps. *ACM Trans. Graph.*, 2016, 35(4), doi:10.1145/2897824.2925919.

[127] Zarifi O, Batty C. A Positive-Definite Cut-Cell Method for Strong Two-Way Coupling between Fluids and Deformable Bodies. In *Proceedings of the ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, SCA ’17, 2017, doi:10.1145/3099564.3099572.

[128] Chen YL, Meier J, Solenthaler B, Azevedo VC. An Extended Cut-Cell Method for Sub-Grid Liquids Tracking with Surface Tension. *ACM Trans. Graph.*, 2020, 39(6), doi:10.1145/3414685.3417859.

[129] Tao M, Batty C, Ben-Chen M, Fiume E, Levin DIW. VEMPIC: Particle-in-Polyhedron Fluid Simulation for Intricate Solid Boundaries. *ACM Trans. Graph.*, 2022, 41(4), doi:10.1145/3528223.3530138.

[130] Gao M, Tampubolon AP, Jiang C, Sifakis E. An Adaptive Generalized Interpolation Material Point Method for Simulating Elastoplastic Materials. *ACM Trans. Graph.*, 2017, 36(6), doi:10.1145/3130800.3130879.

[131] Fang Y, Qu Z, Li M, Zhang X, Zhu Y, Aanjaneya M, Jiang C. IQ-MPM: An Interface Quadrature Material Point Method for Non-Sticky Strongly Two-Way Coupled Nonlinear Solids and Fluids. *ACM Trans. Graph.*, 2020, 39(4), doi:10.1145/3386569.3392438.

[132] Cao Y, Chen Y, Li M, Yang Y, Zhang X, Aanjaneya M, Jiang C. An Efficient B-Spline Lagrangian/Eulerian Method for Compressible Flow, Shock Waves, and Fracturing Solids. *ACM Trans. Graph.*, 2022, 41(5), doi:10.1145/3519595.

[133] Aanjaneya M. An Efficient Solver for Two-way Coupling Rigid Bodies with Incompressible Flow. *Computer Graphics Forum*, 2018, 37(8): 59–68, doi:10.1111/cgf.13512.

[134] Lai J, Chen Y, Gu Y, Batty C, Wan JW. Fast and Scalable Solvers for the Fluid Pressure Equations with Separating Solid Boundary Conditions. *Computer Graphics Forum*, 2020, 39(2): 23–33, doi:10.1111/cgf.13909.

[135] Takahashi T, Batty C. Monolith: A Monolithic Pressure-Viscosity-Contact Solver for Strong Two-Way Rigid-Rigid Rigid-Fluid Coupling. *ACM Trans. Graph.*, 2020, 39(6), doi:10.1145/3414685.3417798.

[136] Ruan L, Liu J, Zhu B, Sueda S, Wang B, Chen B. Solid-Fluid Interaction with Surface-Tension-Dominant Contact. *ACM Trans. Graph.*, 2021, 40(4), doi:10.1145/3450626.3459862.

[137] Akbay M, Nobles N, Zordan V, Shinar T. An Extended Partitioned Method for Conservative Solid-Fluid Coupling. *ACM Trans. Graph.*, 2018, 37(4), doi:10.1145/3197517.3201345.

[138] Lee M, Hyde D, Li K, Fedkiw R. A Robust Volume Conserving Method for Character-Water Interaction. In *Proceedings of the 18th Annual ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA ’19, 2019, doi:10.1145/3309486.3340244.

[139] Brandt C, Scandolo L, Eisemann E, Hildebrandt K. The Reduced Immersed Method for Real-Time Fluid-Elastic Solid

Interaction and Contact Simulation. *ACM Trans. Graph.*, 2019, 38(6), doi:10.1145/3355089.3356496.

[140] Rungjiratananon W, Kanamori Y, Nishita T. Wetting Effects in Hair Simulation. *Computer Graphics Forum*, 2012, 31(7): 1993–2002, doi:10.1111/j.1467-8659.2012.03191.x.

[141] Chen Z, Kim B, Ito D, Wang H. Wetbrush: GPU-Based 3D Painting Simulation at the Bristle Level. *ACM Trans. Graph.*, 2015, 34(6), doi:10.1145/2816795.2818066.

[142] Fei YR, Maia HT, Batty C, Zheng C, Grinspun E. A Multi-Scale Model for Simulating Liquid-Hair Interactions. *ACM Trans. Graph.*, 2017, 36(4), doi:10.1145/3072959.3073630.

[143] Fei YR, Batty C, Grinspun E, Zheng C. A Multi-Scale Model for Coupling Strands with Shear-Dependent Liquid. *ACM Trans. Graph.*, 2019, 38(6), doi:10.1145/3355089.3356532.

[144] Lee M, Hyde D, Bao M, Fedkiw R. A Skinned Tetrahedral Mesh for Hair Animation and Hair-Water Interaction. *IEEE Transactions on Visualization and Computer Graphics*, 2019, 25(3): 1449–1459, doi:10.1109/TVCG.2018.2808972.

[145] Huber M, Eberhardt B, Weiskopf D. Boundary Handling at Cloth–Fluid Contact. *Computer Graphics Forum*, 2015, 34(1): 14–25, doi:10.1111/cgf.12455.

[146] Jiang C, Gast T, Teran J. Anisotropic Elastoplasticity for Cloth, Knit and Hair Frictional Contact. *ACM Trans. Graph.*, 2017, 36(4), doi:10.1145/3072959.3073623.

[147] Fei YR, Batty C, Grinspun E, Zheng C. A Multi-Scale Model for Simulating Liquid-Fabric Interactions. *ACM Trans. Graph.*, 2018, 37(4), doi:10.1145/3197517.3201392.

[148] Wang X, Liu S, Tong Y. Stain Formation on Deforming Inelastic Cloth. *IEEE Transactions on Visualization and Computer Graphics*, 2018, 24(12): 3214–3224, doi:10.1109/TVCG.2017.2789203.

[149] Zheng Y, Chen Y, Fei G, Dorsey J, Wu E. Simulation of Textile Stains. *IEEE Transactions on Visualization and Computer Graphics*, 2019, 25(7): 2471–2481, doi:10.1109/TVCG.2018.2832039.

[150] Patkar S, Chaudhuri P. Wetting of Porous Solids. *IEEE Transactions on Visualization and Computer Graphics*, 2013, 19(9): 1592–1604, doi:10.1109/TVCG.2013.8.

[151] Vantzos O, Azencot O, Wardeztky M, Rumpf M, Ben-Chen M. Functional Thin Films on Surfaces. *IEEE Transactions on Visualization and Computer Graphics*, 2017, 23(3): 1179–1192, doi:10.1109/TVCG.2016.2605083.

[152] Ren B, Yuan T, Li C, Xu K, Hu SM. Real-Time High-Fidelity Surface Flow Simulation. *IEEE Transactions on Visualization and Computer Graphics*, 2018, 24(8): 2411–2423, doi:10.1109/TVCG.2017.2720672.

[153] Yang T, Chang J, Lin MC, Martin RR, Zhang JJ, Hu SM. A Unified Particle System Framework for Multi-Phase, Multi-Material Visual Simulations. *ACM Trans. Graph.*, 2017, 36(6), doi:10.1145/3130800.3130882.

[154] Yan X, Jiang YT, Li CF, Martin RR, Hu SM. Multiphase SPH Simulation for Interactive Fluids and Solids. *ACM Trans. Graph.*, 2016, 35(4), doi:10.1145/2897824.2925897.

[155] Tampubolon AP, Gast T, Klár G, Fu C, Teran J, Jiang C, Museth K. Multi-Species Simulation of Porous Sand and Water Mixtures. *ACM Trans. Graph.*, 2017, 36(4), doi:10.1145/3072959.3073651.

[156] Gao M, Pradhana A, Han X, Guo Q, Kot G, Sifakis E, Jiang C. Animating Fluid Sediment Mixture in Particle-Laden Flows. *ACM Trans. Graph.*, 2018, 37(4), doi:10.1145/3197517.3201309.

[157] He X, Wang H, Wu E. Projective Peridynamics for Modeling Versatile Elastoplastic Materials. *IEEE Transactions on Visualization and Computer Graphics*, 2018, 24(9): 2589–2599, doi:10.1109/TVCG.2017.2755646.

[158] Takahashi T, Batty C. FrictionalMonolith: A Monolithic Optimization-Based Approach for Granular Flow with Contact-Aware Rigid-Body Coupling. *ACM Trans. Graph.*, 2021, 40(6), doi:10.1145/3478513.3480539.

[159] Gao Y, Li S, Hao A, Qin H. Simulating Multi-Scale, Granular Materials and Their Transitions With a Hybrid Euler-Lagrange Solver. *IEEE Transactions on Visualization and Computer Graphics*, 2021, 27(12): 4483–4494, doi:10.1109/TVCG.2021.3107597.

[160] Solenthaler B, Pajarola R. Density Contrast SPH Interfaces. In *Proceedings of the 2008 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '08, 2008, 211–218, doi:10.2312/SCA/SCA08/211-218.

[161] Alduán I, Tena A, Otaduy MA. DYVERSO: A Versatile Multi-Phase Position-Based Fluids Solution for VFX. *Computer Graphics Forum*, 2017, 36(8): 32–44, doi:10.1111/cgf.12992.

[162] Yan X, Li CF, Chen XS, Hu SM. MPM simulation of interacting fluids and solids. *Computer Graphics Forum*, 2018, 37(8): 183–193, doi:10.1111/cgf.13523.

[163] Misztal MK, Erleben K, Bargteil A, Fursund J, Christensen BB, Andreas Bærentzen J, Bridson R. Multiphase Flow of Immiscible Fluids on Unstructured Moving Meshes. *IEEE Transactions on Visualization and Computer Graphics*, 2014, 20(1): 4–16, doi:10.1109/TVCG.2013.97.

[164] Da F, Batty C, Grinspun E. Multimaterial Mesh-Based Surface Tracking. *ACM Trans. Graph.*, 2014, 33(4), doi:10.1145/2601097.2601146.

[165] Li X, He X, Liu X, Zhang JJ, Liu B, Wu E. Multiphase Interface Tracking with Fast Semi-Lagrangian Contouring. *IEEE Transactions on Visualization and Computer Graphics*, 2016, 22(8): 1973–1986, doi:10.1109/TVCG.2015.2476788.

[166] Yang M, Ye J, Ding F, Zhang Y, Yan DM. A Semi-Explicit Surface Tracking Mechanism for Multi-Phase Immiscible Liquids. *IEEE Transactions on Visualization and Computer Graphics*, 2019, 25(10): 2873–2885, doi:10.1109/TVCG.2018.2864283.

[167] Ren B, Li C, Yan X, Lin MC, Bonet J, Hu SM. Multiple-Fluid SPH Simulation Using a Mixture Model. *ACM Trans. Graph.*, 2014, 33(5), doi:10.1145/2645703.

[168] Jiang Y, Li C, Deng S, Hu SM. A Divergence-free Mixture

Model for Multiphase Fluids. *Computer Graphics Forum*, 2020, 39(8): 69–77, doi:10.1111/cgf.14102.

[169] Yang T, Chang J, Ren B, Lin MC, Zhang JJ, Hu SM. Fast Multiple-Fluid Simulation Using Helmholtz Free Energy. *ACM Trans. Graph.*, 2015, 34(6), doi:10.1145/2816795.2818117.

[170] Chen XS, Li CF, Cao GC, Jiang YT, Hu SM. A Moving Least Square Reproducing Kernel Particle Method for Unified Multiphase Continuum Simulation. *ACM Trans. Graph.*, 2020, 39(6), doi:10.1145/3414685.3417809.

[171] Ren B, Xu B, Li C. Unified Particle System for Multiple-Fluid Flow and Porous Material. *ACM Trans. Graph.*, 2021, 40(4), doi:10.1145/3450626.3459764.

[172] Jiang Y, Lan Y. A Dynamic Mixture Model for Non-equilibrium Multiphase Fluids. *Computer Graphics Forum*, 2021, 40(7): 85–95, doi:10.1111/cgf.14403.

[173] Ren B, He W, Li C, Chen X. Incompressibility Enforcement for Multiple-fluid SPH Using Deformation Gradient. *IEEE Transactions on Visualization and Computer Graphics*, 2021: 1–1, doi:10.1109/TVCG.2021.3062643.

[174] Im J, Park H, Kim JH, Kim CH. A Particle-Grid Method for Opaque Ice Formation. *Computer Graphics Forum*, 2013, 32(2pt3): 371–377, doi:10.1111/cgf.12057.

[175] He X, Wang H, Zhang F, Wang H, Wang G, Zhou K, Wu E. Simulation of Fluid Mixing with Interface Control. In *Proceedings of the 14th ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, SCA '15, 2015, 129–135, doi:10.1145/2786784.2786791.

[176] Xue T, Su H, Han C, Jiang C, Aanjaneya M. A Novel Discretization and Numerical Solver for Non-Fourier Diffusion. *ACM Trans. Graph.*, 2020, 39(6), doi:10.1145/3414685.3417863.

[177] Su H, Xue T, Han C, Jiang C, Aanjaneya M. A Unified Second-Order Accurate in Time MPM Formulation for Simulating Viscoelastic Liquids with Phase Change. *ACM Trans. Graph.*, 2021, 40(4), doi:10.1145/3450626.3459820.

[178] Stomakhin A, Schroeder C, Jiang C, Chai L, Teran J, Selle A. Augmented MPM for Phase-Change and Varied Materials. *ACM Trans. Graph.*, 2014, 33(4), doi:10.1145/2601097.2601176.

[179] Hochstetter H, Kolb A. Evaporation and Condensation of SPH-Based Fluids. In *Proceedings of the ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, SCA '17, 2017, doi:10.1145/3099564.3099580.

[180] Li W, Liu D, Desbrun M, Huang J, Liu X. Kinetic-Based Multiphase Flow Simulation. *IEEE Transactions on Visualization and Computer Graphics*, 2021, 27(7): 3318–3334, doi:10.1109/TVCG.2020.2972357.

[181] Wang H, Mucha PJ, Turk G. Water Drops on Surfaces. *ACM Trans. Graph.*, 2005, 24(3): 921–929, doi:10.1145/1073204.1073284.

[182] Zhang Y, Wang H, Wang S, Tong Y, Zhou K. A Deformable Surface Model for Real-Time Water Drop Animation. *IEEE Transactions on Visualization and Computer Graphics*, 2012, 18(8): 1281–1289, doi:10.1109/TVCG.2011.141.

[183] Da F, Hahn D, Batty C, Wojtan C, Grinspun E. Surface-Only Liquids. *ACM Trans. Graph.*, 2016, 35(4), doi:10.1145/2897824.2925899.

[184] Akinci N, Akinci G, Teschner M. Versatile Surface Tension and Adhesion for SPH Fluids. *ACM Trans. Graph.*, 2013, 32(6), doi:10.1145/2508363.2508395.

[185] Orthmann J, Hochstetter H, Bader J, Bayraktar S, Kolb A. Consistent Surface Model for SPH-Based Fluid Transport. In *Proceedings of the 12th ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '13, 2013, 95–103, doi:10.1145/2485895.2485902.

[186] Yang T, Martin RR, Lin MC, Chang J, Hu SM. Pairwise Force SPH Model for Real-Time Multi-Interaction Applications. *IEEE Transactions on Visualization and Computer Graphics*, 2017, 23(10): 2235–2247, doi:10.1109/TVCG.2017.2706289.

[187] He X, Wang H, Zhang F, Wang H, Wang G, Zhou K. Robust Simulation of Sparsely Sampled Thin Features in SPH-Based Free Surface Flows. *ACM Trans. Graph.*, 2015, 34(1), doi:10.1145/2682630.

[188] Hyde DAB, Gagniere SW, Marquez-Razon A, Teran J. An Implicit Updated Lagrangian Formulation for Liquids with Large Surface Energy. *ACM Trans. Graph.*, 2020, 39(6), doi:10.1145/3414685.3417845.

[189] Chen J, Kala V, Marquez-Razon A, Gueidon E, Hyde DAB, Teran J. A Momentum-Conserving Implicit Material Point Method for Surface Tension with Contact Angles and Spatial Gradients. *ACM Trans. Graph.*, 2021, 40(4), doi:10.1145/3450626.3459874.

[190] Patkar S, Aanjaneya M, Karpman D, Fedkiw R. A Hybrid Lagrangian-Eulerian Formulation for Bubble Generation and Dynamics. In *Proceedings of the 12th ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '13, 2013, 105–114, doi:10.1145/2485895.2485912.

[191] Cho J, Ko HS. Geometry-Aware Volume-of-Fluid Method. *Computer Graphics Forum*, 2013, 32(2pt3): 379–388, doi:10.1111/cgf.12058.

[192] Goldade R, Aanjaneya M, Batty C. Constraint Bubbles and Affine Regions: Reduced Fluid Models for Efficient Immersed Bubbles and Flexible Spatial Coarsening. *ACM Trans. Graph.*, 2020, 39(4), doi:10.1145/3386569.3392455.

[193] Padilla M, Chern A, Knöppel F, Pinkall U, Schröder P. On Bubble Rings and Ink Chandeliers. *ACM Trans. Graph.*, 2019, 38(4), doi:10.1145/3306346.3322962.

[194] Langlois TR, Zheng C, James DL. Toward Animating Water with Complex Acoustic Bubbles. *ACM Trans. Graph.*, 2016, 35(4), doi:10.1145/2897824.2925904.

[195] Popinet S. Gerris: a tree-based adaptive solver for the incompressible Euler equations in complex geometries. *Journal of Computational Physics*, 2003, 190(2): 572–600, doi:10.1016/S0021-9991(03)00298-5.

[196] Busaryev O, Dey TK, Wang H, Ren Z. Animating Bubble Interactions in a Liquid Foam. *ACM Trans. Graph.*, 2012, 31(4), doi:10.1145/2185520.2185559.

[197] Kim JH, Lee J, Cha S, Kim CH. Efficient Representation of Detailed Foam Waves by Incorporating Projective Space. *IEEE Transactions on Visualization and Computer Graphics*, 2017, 23(9): 2056–2068, doi:10.1109/TVCG.2016.2609429.

[198] Wretborn J, Flynn S, Stomakhin A. Guided Bubbles and Wet Foam for Realistic Whitewater Simulation. *ACM Trans. Graph.*, 2022, 41(4), doi:10.1145/3528223.3530059.

[199] Boyd L, Bridson R. MultiFLIP for Energetic Two-Phase Fluid Simulation. *ACM Trans. Graph.*, 2012, 31(2), doi: 10.1145/2159516.2159522.

[200] Ando R, Thuerey N, Wojtan C. A Stream Function Solver for Liquid Simulations. *ACM Trans. Graph.*, 2015, 34(4), doi:10.1145/2766935.

[201] Nielsen MB, Østerby O. A Two-Continua Approach to Eulerian Simulation of Water Spray. *ACM Trans. Graph.*, 2013, 32(4), doi:10.1145/2461912.2461918.

[202] Jones R, Southern R. Physically-Based Droplet Interaction. In *Proceedings of the ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, SCA '17, 2017, doi: 10.1145/3099564.3099573.

[203] Yang L, Li S, Hao A, Qin H. Hybrid Particle-grid Modeling for Multi-scale Droplet/Spray Simulation. *Computer Graphics Forum*, 2014, 33(7): 199–208, doi:10.1111/cgf.12488.

[204] Guo Y, Liu X, Xu X. A Unified Detail-Preserving Liquid Simulation by Two-Phase Lattice Boltzmann Modeling. *IEEE Transactions on Visualization and Computer Graphics*, 2017, 23(5): 1479–1491, doi:10.1109/TVCG.2016.2532335.

[205] Li W, Ma Y, Liu X, Desbrun M. Efficient Kinetic Simulation of Two-Phase Flows. *ACM Trans. Graph.*, 2022, 41(4), doi: 10.1145/3528223.3530132.

[206] Ruuth SJ, Merriman B. A simple embedding method for solving partial differential equations on surfaces. *Journal of Computational Physics*, 2008, 227(3): 1943–1961, doi: 10.1016/j.jcp.2007.10.009.

[207] Auer S, Macdonald CB, Treib M, Schneider J, Westermann R. Real-time fluid effects on surfaces using the closest point method. *Computer Graphics Forum*, 2012, 31(6): 1909–1923, doi:10.1111/j.1467-8659.2012.03071.x.

[208] Auer S, Westermann R. A Semi-Lagrangian Closest Point Method for Deforming Surfaces. *Computer Graphics Forum*, 2013, 32(7): 207–214, doi:10.1111/cgf.12228.

[209] Kim T, Tessendorf J, Thürey N. Closest Point Turbulence for Liquid Surfaces. *ACM Trans. Graph.*, 2013, 32(2), doi: 10.1145/2451236.2451241.

[210] Tessendorf J, et al.. Simulating ocean water. *Simulating nature: realistic and interactive techniques. SIGGRAPH*, 2001, 1(2): 5.

[211] Mercier O, Beauchemin C, Thuerey N, Kim T, Nowrouzezahrai D. Surface Turbulence for Particle-Based Liquid Simulations. *ACM Trans. Graph.*, 2015, 34(6), doi: 10.1145/2816795.2818115.

[212] Goldade R, Batty C, Wojtan C. A Practical Method for High-Resolution Embedded Liquid Surfaces. *Computer Graphics Forum*, 2016, 35(2): 233–242, doi:10.1111/cgf.12826.

[213] Morgenroth D, Reinhardt S, Weiskopf D, Eberhardt B. Efficient 2D Simulation on Moving 3D Surfaces. *Computer Graphics Forum*, 2020, 39(8): 27–38, doi:10.1111/cgf.14098.

[214] Pan Z, Huang J, Tong Y, Bao H. Wake Synthesis For Shallow Water Equation. *Computer Graphics Forum*, 2012, 31(7): 2029–2036, doi:10.1111/j.1467-8659.2012.03195.x.

[215] Azencot O, Weißmann S, Ovsjanikov M, Wardetzky M, Ben-Chen M. Functional Fluids on Surfaces. *Computer Graphics Forum*, 2014, 33(5): 237–246, doi:10.1111/cgf.12449.

[216] Azencot O, Vantzos O, Ben-Chen M. An explicit structure-preserving numerical scheme for EPDiff. *Computer Graphics Forum*, 2018, 37(5): 107–119, doi:10.1111/cgf.13495.

[217] Holm DD, Schmah T, Stoica C. *Geometric mechanics and symmetry: from finite to infinite dimensions*, volume 12. 2009.

[218] Canabal JA, Miraut D, Thuerey N, Kim T, Portilla J, Otaduy MA. Dispersion Kernels for Water Wave Simulation. *ACM Trans. Graph.*, 2016, 35(6), doi:10.1145/2980179.2982415.

[219] Airy GB. *Tides and waves*. 1845.

[220] Jeschke S, Wojtan C. Water Wave Animation via Wavefront Parameter Interpolation. *ACM Trans. Graph.*, 2015, 34(3), doi:10.1145/2714572.

[221] Jeschke S, Wojtan C. Water Wave Packets. *ACM Trans. Graph.*, 2017, 36(4), doi:10.1145/3072959.3073678.

[222] Skrivan T, Soderstrom A, Johansson J, Sprenger C, Museth K, Wojtan C. Wave Curves: Simulating Lagrangian Water Waves on Dynamically Deforming Surfaces. *ACM Trans. Graph.*, 2020, 39(4), doi:10.1145/3386569.3392466.

[223] Nielsen MB, Söderström A, Bridson R. Synthesizing waves from animated height fields. *ACM Transactions on Graphics (TOG)*, 2013, 32(1): 1–9, doi:10.1145/2421636.2421638.

[224] Keeler T, Bridson R. Ocean waves animation using boundary integral equations and explicit mesh tracking. *SCA 2014 - Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 2014: 11–19, doi: 10.1145/2614217.2614245.

[225] Schreck C, Hafner C, Wojtan C. Fundamental Solutions for Water Wave Animation. *ACM Trans. Graph.*, 2019, 38(4), doi:10.1145/3306346.3323002.

[226] Jeschke S, Skřivan T, Müller-Fischer M, Chentanez N, Macklin M, Wojtan C. Water Surface Wavelets. *ACM Trans. Graph.*, 2018, 37(4), doi:10.1145/3197517.3201336.

[227] Jeschke S, Hafner C, Chentanez N, Macklin M, Müller-Fischer M, Wojtan C. Making Procedural Water Waves Boundary-aware. *Computer Graphics Forum*, 2020, 39(8): 47–54, doi: 10.1111/cgf.14100.

[228] Schreck C, Wojtan C. Coupling 3D Liquid Simulation with 2D Wave Propagation for Large Scale Water Surface Animation

Using the Equivalent Sources Method. *Computer Graphics Forum*, 2022, 41(2): 343–353, doi:10.1111/cgf.14478.

[229] Bojsen-Hansen M, Li H, Wojtan C. Tracking Surfaces with Evolving Topology. *ACM Trans. Graph.*, 2012, 31(4), doi:10.1145/2185520.2185549.

[230] Bojsen-Hansen M, Wojtan C. Liquid Surface Tracking with Error Compensation. *ACM Trans. Graph.*, 2013, 32(4), doi:10.1145/2461912.2461991.

[231] Edwards E, Bridson R. Detailed Water with Coarse Grids: Combining Surface Meshes and Adaptive Discontinuous Galerkin. *ACM Trans. Graph.*, 2014, 33(4), doi:10.1145/2601097.2601167.

[232] Arnold DN, Brezzi F, Cockburn B, Marini LD. Unified analysis of discontinuous Galerkin methods for elliptic problems. *SIAM journal on numerical analysis*, 2002, 39(5): 1749–1779, doi:10.1137/S0036142901384162.

[233] Chentanez N, Müller M, Macklin M, Kim TY. Fast Grid-Free Surface Tracking. *ACM Trans. Graph.*, 2015, 34(4), doi:10.1145/2766991.

[234] Yu J, Wojtan C, Turk G, Yap C. Explicit Mesh Surfaces for Particle Based Fluids. *Computer Graphics Forum*, 2012, 31(2pt4): 815–824, doi:10.1111/j.1467-8659.2012.03062.x.

[235] Yu J, Turk G. Reconstructing Surfaces of Particle-Based Fluids Using Anisotropic Kernels. *ACM Trans. Graph.*, 2013, 32(1), doi:10.1145/2421636.2421641.

[236] Sandim M, Cedrim D, Nonato LG, Pagliosa P, Paiva A. Boundary Detection in Particle-based Fluids. *Computer Graphics Forum*, 2016, 35(2): 215–224, doi:10.1111/cgf.12824.

[237] Dagenais F, Gagnon J, Paquette E. Detail-Preserving Explicit Mesh Projection and Topology Matching for Particle-Based Fluids. *Computer Graphics Forum*, 2017, 36(8): 444–457, doi:10.1111/cgf.13091.

[238] Fedkiw R, Stam J, Jensen HW. Visual simulation of smoke. In *Proc. ACM SIGGRAPH*, 2001, 15–22, doi:10.1145/383259.383260.

[239] Lentine M, Aanjaneya M, Fedkiw R. Mass and momentum conservation for fluid simulation. In *Proceedings of the 2011 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 2011, 91–100, doi:10.1145/2019406.2019419.

[240] Jang T, Kim H, Bae J, Seo J, Noh J. Multilevel vorticity confinement for water turbulence simulation. *The Visual Computer*, 2010, 26(6–8): 873–881, doi:10.1007/s00371-010-0487-1.

[241] He S, Lau RWH. Synthetic Controllable Turbulence Using Robust Second Vorticity Confinement. *Computer Graphics Forum*, 2013, 32(1): 27–35, doi:10.1111/j.1467-8659.2012.03228.x.

[242] Zhang X, Bridson R, Greif C. Restoring the Missing Vorticity in Advection-Projection Fluid Solvers. *ACM Trans. Graph.*, 2015, 34(4), doi:10.1145/2766982.

[243] Liu S, Wang X, Ban X, Xu Y, Zhou J, Kosinka J, Telea AC. Turbulent Details Simulation for SPH Fluids via Vorticity Refinement. *Computer Graphics Forum*, 2021, 40(1): 54–67, doi:10.1111/cgf.14095.

[244] Xiong S, Tao R, Zhang Y, Feng F, Zhu B. Incompressible Flow Simulation on Vortex Segment Clouds. *ACM Trans. Graph.*, 2021, 40(4), doi:10.1145/3450626.3459865.

[245] Golas A, Narain R, Sewall J, Krajcevski P, Dubey P, Lin M. Large-Scale Fluid Simulation Using Velocity-Vorticity Domain Decomposition. *ACM Trans. Graph.*, 2012, 31(6), doi:10.1145/2366145.2366167.

[246] Zhang X, Li M, Bridson R. Resolving Fluid Boundary Layers with Particle Strength Exchange and Weak Adaptivity. *ACM Trans. Graph.*, 2016, 35(4), doi:10.1145/2897824.2925910.

[247] Liao X, Si W, Yuan Z, Sun H, Qin J, Wang Q, Heng PA. Animating Wall-Bounded Turbulent Smoke via Filament-Mesh Particle-Particle Method. *IEEE Transactions on Visualization and Computer Graphics*, 2018, 24(3): 1260–1273, doi:10.1109/TVCG.2017.2665551.

[248] Wu X, Yang X, Yang Y. A Novel Projection Technique with Detail Capture and Shape Correction for Smoke Simulation. *Computer Graphics Forum*, 2013, 32(2pt4): 389–397, doi:10.1111/cgf.12059.

[249] Zehnder J, Narain R, Thomaszewski B. An Advection-Reflection Solver for Detail-Preserving Fluid Simulation. *ACM Trans. Graph.*, 2018, 37(4), doi:10.1145/3197517.3201324.

[250] Nabizadeh MS, Wang S, Ramamoorthi R, Chern A. Covector Fluids. *ACM Trans. Graph.*, 2022, 41(4), doi:10.1145/3528223.3530120.

[251] Yang S, Xiong S, Zhang Y, Feng F, Liu J, Zhu B. Clebsch Gauge Fluid. *ACM Trans. Graph.*, 2021, 40(4), doi:10.1145/3450626.3459866.

[252] Xiong S, Wang Z, Wang M, Zhu B. A Clebsch Method for Free-Surface Vortical Flow Simulation. *ACM Trans. Graph.*, 2022, 41(4), doi:10.1145/3528223.3530150.

[253] Feng F, Liu J, Xiong S, Yang S, Zhang Y, Zhu B. Impulse Fluid Simulation. *IEEE Transactions on Visualization and Computer Graphics*, 2022: 1–1, doi:10.1109/TVCG.2022.3149466.

[254] Liu X, Pang WM, Qin J, Fu CW. Turbulence Simulation by Adaptive Multi-Relaxation Lattice Boltzmann Modeling. *IEEE Transactions on Visualization and Computer Graphics*, 2014, 20(2): 289–302, doi:10.1109/TVCG.2012.303.

[255] Li W, Chen Y, Desbrun M, Zheng C, Liu X. Fast and Scalable Turbulent Flow Simulation with Two-Way Coupling. *ACM Trans. Graph.*, 2020, 39(4), doi:10.1145/3386569.3392400.

[256] Lyu C, Li W, Desbrun M, Liu X. Fast and Versatile Fluid-Solid Coupling for Turbulent Flow Simulation. *ACM Trans. Graph.*, 2021, 40(6), doi:10.1145/3478513.3480493.

[257] Bender J, Koschier D, Kugelstadt T, Weiler M. A Micropolar Material Model for Turbulent SPH Fluids. In *Proceedings of the ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, SCA '17, 2017, doi:10.1145/3099564.3099578.

[258] Jamriška O, Fišer J, Asente P, Lu J, Shechtman E, Sýkora D. LazyFluids: Appearance Transfer for Fluid Animations. *ACM Trans. Graph.*, 2015, 34(4), doi:10.1145/2766983.

[259] Gagnon J, Guzmán JE, Vervondel V, Dagenais F, Mould D, Paquette E. Distribution Update of Deformable Patches for Texture Synthesis on the Free Surface of Fluids. *Computer Graphics Forum*, 2019, 38(7): 491–500, doi:10.1111/cgf.13855.

[260] Gagnon J, Guzmán JE, Mould D, Paquette E. Patch Erosion for Deformable Lapped Textures on 3D Fluids. *Computer Graphics Forum*, 2021, 40(2): 367–374, doi:10.1111/cgf.142639.

[261] Sato S, Dobashi Y, Kim T, Nishita T. Example-Based Turbulence Style Transfer. *ACM Trans. Graph.*, 2018, 37(4), doi:10.1145/3197517.3201398.

[262] Kim B, Azevedo VC, Gross M, Solenthaler B. Transport-Based Neural Style Transfer for Smoke Simulations. *ACM Trans. Graph.*, 2019, 38(6), doi:10.1145/3355089.3356560.

[263] Kim B, Azevedo VC, Gross M, Solenthaler B. Lagrangian Neural Style Transfer for Fluids. *ACM Trans. Graph.*, 2020, 39(4), doi:10.1145/3386569.3392473.

[264] Guo J, Li M, Zong Z, Liu Y, He J, Guo Y, Yan LQ. Volumetric Appearance Stylization with Stylizing Kernel Prediction Network. *ACM Trans. Graph.*, 2021, 40(4), doi:10.1145/3450626.3459799.

[265] Xie Y, Franz E, Chu M, Thuerey N. TempoGAN: A Temporally Coherent, Volumetric GAN for Super-Resolution Fluid Flow. *ACM Trans. Graph.*, 2018, 37(4), doi:10.1145/3197517.3201304.

[266] Zhang Y, Ma KL. Spatio-Temporal Extrapolation for Fluid Animation. *ACM Trans. Graph.*, 2013, 32(6), doi:10.1145/2508363.2508401.

[267] Chu M, Thuerey N. Data-Driven Synthesis of Smoke Flows with CNN-Based Feature Descriptors. *ACM Trans. Graph.*, 2017, 36(4), doi:10.1145/3072959.3073643.

[268] Um K, Hu X, Thuerey N. Perceptual Evaluation of Liquid Simulation Methods. *ACM Trans. Graph.*, 2017, 36(4), doi:10.1145/3072959.3073633.

[269] Xiao X, Wang H, Yang X. A CNN-based Flow Correction Method for Fast Preview. *Computer Graphics Forum*, 2019, 38(2): 431–440, doi:10.1111/cgf.13649.

[270] Li C, Qiu S, Wang C, Qin H. Learning Physical Parameters and Detail Enhancement for Gaseous Scene Design Based on Data Guidance. *IEEE Transactions on Visualization and Computer Graphics*, 2021, 27(10): 3867–3880, doi:10.1109/TVCG.2020.2991217.

[271] Bai K, Li W, Desbrun M, Liu X. Dynamic Upsampling of Smoke through Dictionary-Based Learning. *ACM Trans. Graph.*, 2020, 40(1), doi:10.1145/3412360.

[272] Bai K, Wang C, Desbrun M, Liu X. Predicting High-Resolution Turbulence Details in Space and Time. *ACM Trans. Graph.*, 2021, 40(6), doi:10.1145/3478513.3480492.

[273] Roy B, Poulin P, Paquette E. Neural UpFlow: A Scene Flow Learning Approach to Increase the Apparent Resolution of Particle-Based Liquids. *Proc. ACM Comput. Graph. Interact. Tech.*, 2021, 4(3), doi:10.1145/3480147.

[274] Gregson J, Ihrke I, Thuerey N, Heidrich W. From Capture to Simulation: Connecting Forward and Inverse Problems in Fluids. *ACM Trans. Graph.*, 2014, 33(4), doi:10.1145/2601097.2601147.

[275] Forootaninia Z, Narain R. Frequency-Domain Smoke Guiding. *ACM Trans. Graph.*, 2020, 39(6), doi:10.1145/3414685.3417842.

[276] Inglis T, Eckert ML, Gregson J, Thuerey N. Primal-Dual Optimization for Fluids. *Computer Graphics Forum*, 2017, 36(8): 354–368, doi:10.1111/cgf.13084.

[277] Pan Z, Manocha D. Efficient Solver for Spacetime Control of Smoke. *ACM Trans. Graph.*, 2017, 36(5), doi:10.1145/3016963.

[278] Boyd S, Parikh N, Chu E, Peleato B, Eckstein J, et al.. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine learning*, 2011, 3(1): 1–122, doi:10.1561/2200000016.

[279] Tang J, C Azevedo V, Cordonnier G, Solenthaler B. Honey, I Shrunk the Domain: Frequency-aware Force Field Reduction for Efficient Fluids Optimization. *Computer Graphics Forum*, 2021, 40(2): 339–353, doi:10.1111/cgf.142637.

[280] Raveendran K, Thuerey N, Wojtan C, Turk G. Controlling Liquids Using Meshes. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '12, 2012, 255–264.

[281] Sato S, Dobashi Y, Nishita T. Editing Fluid Animation Using Flow Interpolation. *ACM Trans. Graph.*, 2018, 37(5), doi:10.1145/3213771.

[282] Flynn S, Egbert P, Holladay S, Morse B. Fluid Carving: Intelligent Resizing for Fluid Simulation Data. *ACM Trans. Graph.*, 2019, 38(6), doi:10.1145/3355089.3356572.

[283] Flynn S, Hart D, Morse B, Holladay S, Egbert P. Generalized Fluid Carving with Fast Lattice-Guided Seam Computation. *ACM Trans. Graph.*, 2021, 40(6), doi:10.1145/3478513.3480544.

[284] Bojsen-Hansen M, Wojtan C. Generalized Non-Reflecting Boundaries for Fluid Re-Simulation. *ACM Trans. Graph.*, 2016, 35(4), doi:10.1145/2897824.2925963.

[285] Stomakhin A, Selle A. Fluxed Animated Boundary Method. *ACM Trans. Graph.*, 2017, 36(4), doi:10.1145/3072959.3073597.

[286] Pan Z, Huang J, Tong Y, Zheng C, Bao H. Interactive Localized Liquid Motion Editing. *ACM Trans. Graph.*, 2013, 32(6), doi:10.1145/2508363.2508429.

[287] Lu JM, Chen XS, Yan X, Li CF, Lin M, Hu SM. A Rigging-Skinning Scheme to Control Fluid Simulation. *Computer Graphics Forum*, 2019, 38(7): 501–512, doi:10.1111/cgf.13856.

[288] Yan G, Chen Z, Yang J, Wang H. Interactive Liquid Splash

Modeling by User Sketches. *ACM Trans. Graph.*, 2020, 39(6), doi:10.1145/3414685.3417832.

[289] Schoentgen A, Poulin P, Darles E, Meseure P. Particle-based Liquid Control using Animation Templates. *Computer Graphics Forum*, 2020, 39(8): 79–88, doi:10.1111/cgf.14103.

[290] Okabe M, Dobashi Y, Anjyo K, Onai R. Fluid Volume Modeling from Sparse Multi-View Images by Appearance Transfer. *ACM Trans. Graph.*, 2015, 34(4), doi:10.1145/2766958.

[291] Eckert ML, Heidrich W, Thuerey N. Coupled Fluid Density and Motion from Single Views. *Computer Graphics Forum*, 2018, 37(8): 47–58, doi:10.1111/cgf.13511.

[292] Nie X, Hu Y, Su Z, Shen X. Fluid Reconstruction and Editing from a Monocular Video based on the SPH Model with External Force Guidance. *Computer Graphics Forum*, 2021, 40(6): 62–76, doi:10.1111/cgf.14203.

[293] Zhu B, Lee M, Quigley E, Fedkiw R. Codimensional Non-Newtonian Fluids. *ACM Trans. Graph.*, 2015, 34(4), doi:10.1145/2766981.

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

[295] Peer A, Ihmsen M, Cornelis J, Teschner M. An Implicit Viscosity Formulation for SPH Fluids. *ACM Trans. Graph.*, 2015, 34(4), doi:10.1145/2766925.

[296] Peer A, Teschner M. Prescribed Velocity Gradients for Highly Viscous SPH Fluids with Vorticity Diffusion. *IEEE Transactions on Visualization and Computer Graphics*, 2017, 23(12): 2656–2662, doi:10.1109/TVCG.2016.2636144.

[297] Weiler M, Koschier D, Brand M, Bender J. A Physically Consistent Implicit Viscosity Solver for SPH Fluids. *Computer Graphics Forum*, 2018, 37(2): 145–155, doi:10.1111/cgf.13349.

[298] Monaghan JJ. Smoothed particle hydrodynamics. *Reports on progress in physics*, 2005, 68(8): 1703, doi:10.1088/0034-4885/68/8/R01.

[299] Larionov E, Batty C, Bridson R. Variational Stokes: A Unified Pressure-Viscosity Solver for Accurate Viscous Liquids. *ACM Trans. Graph.*, 2017, 36(4), doi:10.1145/3072959.3073628.

[300] Liu S, He X, Wang W, Wu E. Adapted SIMPLE Algorithm for Incompressible SPH Fluids With a Broad Range Viscosity. *IEEE Transactions on Visualization and Computer Graphics*, 2022, 28(9): 3168–3179, doi:10.1109/TVCG.2021.3055789.

[301] Yue Y, Smith B, Batty C, Zheng C, Grinspun E. Continuum Foam: A Material Point Method for Shear-Dependent Flows. *ACM Trans. Graph.*, 2015, 34(5), doi:10.1145/2751541.

[302] Nagasawa K, Suzuki T, Seto R, Okada M, Yue Y. Mixing Sauces: A Viscosity Blending Model for Shear Thinning Fluids. *ACM Trans. Graph.*, 2019, 38(4), doi:10.1145/3306346.3322947.

[303] Rusin M. The structure of nonlinear blending models. *Chemical Engineering Science*, 1975, 30(8): 937–944, doi:10.1016/0009-2509(75)80060-1.

[304] Barreiro H, García-Fernández I, Alduán I, Otaduy MA. Conformation Constraints for Efficient Viscoelastic Fluid Simulation. *ACM Trans. Graph.*, 2017, 36(6), doi:10.1145/3130800.3130854.

[305] Huang L, Hädrich T, Michels DL. On the Accurate Large-Scale Simulation of Ferrofluids. *ACM Trans. Graph.*, 2019, 38(4), doi:10.1145/3306346.3322973.

[306] Shao H, Huang L, Michels D. A Current Loop Model for the Fast Simulation of Ferrofluids. *IEEE Transactions on Visualization and Computer Graphics*, 2022, PP: 1–12, doi:10.1109/TVCG.2022.3211414.

[307] Ni X, Zhu B, Wang B, Chen B. A Level-Set Method for Magnetic Substance Simulation. *ACM Trans. Graph.*, 2020, 39(4), doi:10.1145/3386569.3392445.

[308] Huang L, Michels DL. Surface-Only Ferrofluids. *ACM Trans. Graph.*, 2020, 39(6), doi:10.1145/3414685.3417799.

[309] Sun Y, Ni X, Zhu B, Wang B, Chen B. A Material Point Method for Nonlinearly Magnetized Materials. *ACM Trans. Graph.*, 2021, 40(6), doi:10.1145/3478513.3480541.

[310] Batty C, Uribe A, Audoly B, Grinspun E. Discrete Viscous Sheets. *ACM Trans. Graph.*, 2012, 31(4), doi:10.1145/2185520.2185609.

[311] Wang H, Jin Y, Luo A, Yang X, Zhu B. Codimensional Surface Tension Flow Using Moving-Least-Squares Particles. *ACM Trans. Graph.*, 2020, 39(4), doi:10.1145/3386569.3392487.

[312] Zhu B, Quigley E, Cong M, Solomon J, Fedkiw R. Codimensional Surface Tension Flow on Simplicial Complexes. *ACM Trans. Graph.*, 2014, 33(4), doi:10.1145/2601097.2601201.

[313] Wang M, Deng Y, Kong X, Prasad AH, Xiong S, Zhu B. Thin-Film Smoothed Particle Hydrodynamics Fluid. *ACM Trans. Graph.*, 2021, 40(4), doi:10.1145/3450626.3459864.

[314] Vantzos O, Raz S, Ben-Chen M. Real-Time Viscous Thin Films. *ACM Trans. Graph.*, 2018, 37(6), doi:10.1145/3272127.3275086.

[315] Da F, Batty C, Wojtan C, Grinspun E. Double Bubbles sans Toil and Trouble: Discrete Circulation-Preserving Vortex Sheets for Soap Films and Foams. *ACM Trans. Graph.*, 2015, 34(4), doi:10.1145/2767003.

[316] Ishida S, Yamamoto M, Ando R, Hachisuka T. A Hyperbolic Geometric Flow for Evolving Films and Foams. *ACM Trans. Graph.*, 2017, 36(6), doi:10.1145/3130800.3130835.

[317] Ishida S, Synak P, Narita F, Hachisuka T, Wojtan C. A Model for Soap Film Dynamics with Evolving Thickness. *ACM Trans. Graph.*, 2020, 39(4), doi:10.1145/3386569.3392405.

[318] Hill DJ, Henderson RD. Efficient Fluid Simulation on the Surface of a Sphere. *ACM Trans. Graph.*, 2016, 35(2), doi:10.1145/2879177.

[319] Huang W, Iseringhausen J, Kneiphof T, Qu Z, Jiang C, Hullin MB. Chemomechanical Simulation of Soap Film Flow on Spherical Bubbles. *ACM Trans. Graph.*, 2020, 39(4), doi:10.1145/3386569.3392094.

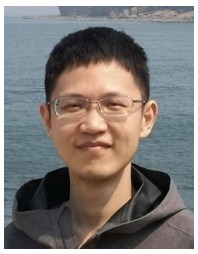
[320] Deng Y, Wang M, Kong X, Xiong S, Xian Z, Zhu B. A Moving Eulerian-Lagrangian Particle Method for Thin Film and Foam Simulation. *ACM Trans. Graph.*, 2022, 41(4), doi:10.1145/3528223.3530174.

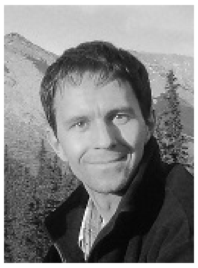
Author biography

Xiaokun Wang is an associate professor in Intelligence Science and Technology, University of Science and Technology Beijing, China. He received a Ph.D. degree in Computer Science and Technology from the University of Science and Technology Beijing, in 2017. He is currently working at the National Centre for Computer Animation at Bournemouth University funded by the EU's Horizon 2020 Marie Curie Individual Fellowship. His research interests include computer graphics, virtual reality and human-computer interaction.

Yanrui Xu is a PhD student at Bernoulli Institute, University of Groningen and the School of Intelligence Science and Technology, University of Science and Technology Beijing. He received his Master's degree from the University of Science and Technology Beijing in 2020. His research interests include physical-based fluid simulation.

Sinuo Liu is a postdoc in School of Computer Science, Peking University, China. She received a Ph.D. degree in Computer Science and Technology from University of Science and Technology Beijing, in 2021. Her research interests include computer graphics and virtual reality.


Bo Ren is an associate professor at College of Computer Science, Nankai University. He received his Ph.D. degree from Tsinghua University in 2015. His research interests lie in computer graphics, computer vision and artificial intelligence. Current researches involve learning-based/physically-based simulation, 3D scene geometry reconstruction and analysis.


Jiří Kosinka is an associate professor at Bernoulli Institute, University of Groningen. He received his Ph.D. degree from the Faculty of Mathematics and Physics, Charles University in 2006. His research interests include Geometric Modelling, Computer Aided (Geometric) Design, Computer Graphics and Image Processing.

Alexandru Telea is professor of visual data analysis at the Information and Computing Science Department, Utrecht University, the Netherlands. His research interests cover multiscale shape and image processing, information visualization (with a focus on high-dimensional and relational data), and image-based data visualization.

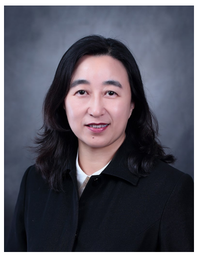
Jiamin Wang is a PhD student at the School of Intelligence Science and Technology, University of Science and Technology Beijing. She received an undergraduate Diploma in University of Science and Technology Beijing, in 2021. Her research field is computer graphics, especially physically-based fluid simulation.

Chongming Song is a postgraduate student at the School of Intelligence Science and Technology, University of Science and Technology Beijing. She received an undergraduate Diploma in University of Science and Technology Beijing, in 2021. Her research field is computer graphics, especially physically-based fluid simulation.

Jian Chang is a professor in the National Centre Computer Animation at Bournemouth University, United Kingdom. He received his Ph.D. degree in Computer Graphics from Bournemouth University in 2007. His research interests include physics based modeling (deformation & fluid), motion synthesis, virtual reality (surgery simulation), and novel HCI (eye tracking, gesture control, and haptics).

Chenfeng Li is a Professor in Civil Engineering, Swansea University. He received his Ph.D. degree in Computational Engineering from Swansea University in 2007. His research centres largely on Computational Solid Mechanics, Computational Fluid Dynamics, Physics-based Data Mining and Visual Computing, to develop computational solutions for technical challenges in Civil Engineering, Geo-mechanics, Oil & Gas reservoir, Material, and Manufacturing sectors.

Jian Jun Zhang is a professor and director of the UK National Centre for Computer Animation Research at Bournemouth University. His research interests include 3D computer animation, virtual human modeling, virtual reality and physics-based simulation.



Xiaojuan Ban is a professor at the School of Intelligence Science and Technology, University of Science and Technology Beijing. She is the leader of the Artificial Intelligence and 3D Visualization Group at University of Science and Technology Beijing, China. She received her master's degree in computer application and Ph.D. degree in control theory and control engineering from the University of Science and Technology Beijing. Her research interests include computer graphics, artificial intelligence, human-computer interaction, big data analysis, and 3D visualization.