



An efficient FLIP and shape matching coupled method for fluid–solid and two-phase fluid simulations

Yang Gao¹ · Shuai Li² · Hong Qin³ · Yinghao Xu¹ · Aimin Hao¹

Published online: 11 June 2018

© Springer-Verlag GmbH Germany, part of Springer Nature 2018

Abstract

Solid dynamic deformation and multiphase fluid coupling driven by numerical simulation have manifested their significance for many graphics applications during the past 2 decades. For example, the fluid implicit particle (FLIP) method and shape matching constraint based on position-based dynamics (PBD) have demonstrated their unique graphics strength in fluid and solid animation, respectively. In this paper, we propose a novel integrated approach supporting the seamless unification of FLIP and dynamic shape matching. We devise new algorithms to tackle existing difficulties when handling new phenomena such as high-fidelity fluid–solid interactions, solid deformations, melting and immiscible fluid coupling. The key innovation of this paper is a unified Lagrangian framework that seamlessly blends FLIP- and PBD-based shape matching constraints toward the natural yet flexible coupling between fluid and deformable solid. Within our integrated framework, it enables many complicated fluid–solid phenomena with ease. We conduct various kinds of experiments, all the results demonstrate the advantages of our unified hybrid approach toward visual fidelity, computational efficiency, numerical stability, and application versatility.

Keywords FLIP · Shape matching · Position-based dynamics · Solid deformation · Immiscible fluid coupling

Electronic supplementary material The online version of this article (<https://doi.org/10.1007/s00371-018-1569-8>) contains supplementary material, which is available to authorized users.

✉ Shuai Li
lishuaiouc@126.com

Yang Gao
gaoyang2963@163.com

Hong Qin
qin@cs.stonybrook.edu

Yinghao Xu
xuyinghao@buaa.edu.cn

Aimin Hao
ham_buaa@163.com

¹ State Key Laboratory of Virtual Reality Technology and Systems, Beihang University, Beijing, China

² State Key Laboratory of Virtual Reality Technology and Systems, Beihang University Qingdao Research Institute, Beihang University, Beijing, China

³ Department of Computer Science, Stony Brook University, Stony Brook, USA

1 Introduction

Visual-based simulations of fluids and deformable solids have been widely studied in graphics. In particular, some applications involve many interesting phenomena relevant to fluid–solid coupling and interactions, such as solid motion and deformation, fluid coupling with rigid/soft bodies, multi-phase fluids. However, certain difficulties still prevail and call for novel algorithms and techniques. In this paper, we mainly focus on the complicated dynamic interactions between fluids, solids and multiphase liquid. Our research is based on the original simple framework [16], and we have improved the boundary condition and developed the shape matching constraint. Compared to our previous work, our new techniques can support more smooth interface between fluid and solid and more complex phenomena, and our improved constraint enables two immiscible two-phase fluid, which cannot be realized by certain methods in [16].

A lot of state-of-the-art methods for visually appealing fluid animation have offered many choices to researchers. For example, smoothed particle hydrodynamics (SPH) methods [12,13,19,32,37] enable great works, and PBD method could borrow the concept of a density estimator from SPH

to realize fluid simulation. When PBD method is used for fluid simulation, fluid particles use additional density constraints with SPH kernels to imitate incompressibility of fluid, and inevitably introduces the time-consuming computation of global neighborhood search and inherent errors into PBD fluid, while FLIP solves the Navier–Stokes equations using grids instead of particles, and only updates the changes of particles' velocities to avoid iteration errors. In recent years, the fluid implicit particle (FLIP) method becomes a very popular particle–grid coupling method, which is good at handling incompressible fluid with complex boundaries [9,13,14]. Although many great ideas have been proposed for fluid simulation based on FLIP or FLIP-coupled methods, FLIP-based fluid–solid interactions and two-phase fluid animations have not been studied as widely as SPH and PBD methods. We examine the FLIP method, which is widely used to simulate high-quality fluid effects because of its less numerical dissipation and better numerical stability [40].

In contrast, PBD methods with various constraints have the advantage for deformable objects simulation with high-level stability. In this paper, we will extend and unify the incompressible FLIP method and shape matching constraint [6,30] to uniformly accommodate multiple phases with ease, including deformation bodies, fluid–solid coupling. Of which, the distributions of all phases (fluids and solids) are uniformly represented by FLIP particles. The dynamics of our multiphase system is governed by a variety of shape matching constraints, which will collectively serve as the constraining condition of the PBD method. As all the materials are represented with FLIP particles, the shape matching constraints are dynamically coupled in a synchronized way as the particles' number and the positions of solid/deformable objects change (either geometrically or topologically). Our salient contributions can be summarized as follows:

- We propose to uniformly model the behaviors of all the involved materials based on the same set of variables in FLIP-driven Navier–Stokes equations, which can greatly reduce the computational burden.
- We detail a coupled FLIP-shape matching framework, which enables to simultaneously simulate a much wider range of fluid–solid phenomena, such as solid deformations, fluid–solid interactions, melting.
- We devise an efficient shape matching constraint to facilitate the interactions between immiscible fluid phases, which ensure the vivid visual simulation and stable numerical computation.

2 Related works

Since this paper mainly focuses on fluid–solid interactions, to keep the review most relevant to our work, we briefly summarize previous works as follows.

2.1 FLIP-based fluid simulation

FLIP method is introduced to computer graphics by Zhu et al. [40], and then it is extended to simulate splashing water [14,18], preserve fluid sheet [4], conduct fluid–solid coupling [33], combine with particle methods [13], model multi-scale droplet/spray [39], etc. For example, Ando and Selino et al. [4,13], respectively, proposed methods to improve the particle distribution of FLIP method. Boyd and Bridson [9] extended FLIP method to model two-phase flows, named as MultiFLIP, which separates velocity fields with a combined divergence-free formulation to enforce overall incompressibility.

2.2 PBD-based simulation

PBD is used to handle position-level constraints based on iterative Gauss–Seidel solver [6,31]. Many works employ PBD for the simulation of deformable objects. For example, Müller et al. [30] introduced a geometric constraint to PBD for deformable object simulation, which serves as the basic framework of our solid simulation. Bender et al. [7] proposed a continuum-based strain energy formulation to solve the constraint function. Tournier et al. [35] formulated a compliant constraint to avoid instabilities due to linearization, which enables the unification of elasticity and constraints. Meanwhile, PBD is also extended for fluid simulation. Macklin et al. [26] proposed a set of positional constraints to enforce constant density for PBD-based fluid simulation, and then they used clamped nonnegative densities to improve the constraint [27], their unified particle representation for all materials and the improved XPBD method [25] achieved good performance when simulating solid dynamics and fluid–solid coupling.

Compared to PBD-based fluid which uses density constraints of SPH [26,27] to ensure incompressibility, our FLIP-based framework combines divergence-free formulation to enforce overall incompressibility, which has less dissipations. And the constraints for soft bodies directly perform on FLIP particles, with very little time consumption and make the hydrodynamics algorithm has power to simulate solids dynamic.

2.3 Fluid–solid coupling

Keiser et al. [20] simulated fluid flow interaction with deformable solids with a unified Navier–Stokes equation, which can also accommodate phase transition. Lenaerts et al. [22] and Lenaerts [21] successfully simulated porous materials and water coupling. Allard et al. [3] and Yang et al. [38], respectively, proposed FEM-based solid and particle-based fluid coupling methods. Akinci et al. [1] proposed a novel boundary sampling method for incompressible SPH fluids,

which can support complex solid–fluid interaction efficiently. And then they extended their rigid–fluid coupling method [2] for elastic–fluid boundary handling by sampling the solids with boundary particles, both of the two works achieved good performances. Shao et al. [34] coupled SPH with lattice shape matching to simulate the interactions between fluid and soft body. Clausen et al. [11] and Misztal et al. [29] employed tetrahedral meshes to simulate materials (ranging from stiff solids to visco-plastic or inviscid fluids), as well as the interactions and phase changes among them. In this paper, we will also focus on fluid coupling with stiffness-changeable materials (ranging from stiff solids to visco-plastic and immiscible fluid), which is somewhat similar to [29]. However, we propose new hybrid particle-grid method (FLIP-Shape matching), which differs from unstructured moving meshes in [29], so that we can better simulate deformable bodies and handle large-scale phenomena. For fluid–solid coupling simulations with unified particle representations, both Carlson et al. [10] and Batty et al. [5] achieved good performances. However, with controllable parameters of shape matching, we can simulate the interactions between fluid and soft bodies with varying stiffness, which is more flexible than other similar approaches.

In summary, compared with the aforementioned works, we blend the FLIP- and PBD-based shape matching constraint toward the flexible coupling between fluid and deformable solid, using FLIP particles to represent all materials instead of PBD particles. Meanwhile, the shape matching constraint can represent the position and orientation variables, linear and angular velocities of objects, which cannot be realized by FLIP particles alone. That is the very reason that we intend to introduce the shape matching constraint into our FLIP framework.

3 FLIP and shape matching models

Our hybrid framework is built upon MultiFLIP and shape matching constraint of PBD, of which, MultiFLIP is used for fluid dynamics and shape matching is only used to handle solid dynamics, which has few influences to FLIP solver; thus, we can support large-scale scenario with sufficient details in a robust and fast way. For the sake of completeness, we now briefly review the basic ideas of them.

3.1 MultiFLIP model

Simulating fluid dynamics essentially needs to solve Navier–Stokes equations (N–S equations), which conserve both mass and momentum:

$$\frac{\partial \rho}{\partial t} + \rho \nabla \cdot \mathbf{u} = 0, \quad (1)$$

$$\rho \left(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} \right) = -\nabla p + \mu \nabla^2 \mathbf{u} + \mathbf{f}, \quad (2)$$

where ρ is the density, $\mathbf{u}$ is the velocity, p is the pressure, and $\mathbf{f}$ is the external force.

In FLIP, the fluid volume is discretized into grid cells, and then traditional Eulerian method is used to solve N–S equations. The velocity change on grid (rather than velocity itself) is interpolated over particles to avoid the numerical dissipation. Usually, FLIP velocity is blended with particle-in-cell (PIC) method to suppress the potential high-frequency noise [40] as:

$$\mathbf{v} = a \mathbf{v}_{\text{FLIP}} + (1 - a) \mathbf{v}_{\text{PIC}}, \quad (3)$$

where the parameter $a \in [0, 1]$ controls the amount of velocity diffusion during simulation, and it can be bridged to the fluid viscosity.

We choose MultiFLIP model as basic framework to simulate fluid and other materials, since it has low numerical dissipation, and can maintain a sharp and clear interface. The same as original MultiFLIP algorithm, our method also takes air domain into consideration to enforce overall incompressibility while maintaining a free-slip condition at the interface; however, the gas/air material is not our simulating target. Both of the liquid and solid/immiscible fluid are represented by fluid particles, and one kind of phases is performed by shape matching constraint to imitate deformable characteristics.

3.2 Constraints of PBD

PBD is a popular method for simulating deformable bodies in computer graphics and interactive applications because of its simplicity and robustness, it omits the velocity and acceleration, directly works on the position [6,31]. The objects to be simulated are represented by a set of particles and a set of constraints

$$C_i(\mathbf{x} + \Delta \mathbf{x}) = 0, \quad i = 1, \dots, n \quad (4)$$

$$C_j(\mathbf{x} + \Delta \mathbf{x}) \geq 0, \quad j = 1, \dots, n. \quad (5)$$

where $\mathbf{x}$ is the position. Constraint C is typically solved through Gauss–Seidel iteration sequentially using the linearization around $\mathbf{x}$,

$$C_i(\mathbf{x} + \Delta \mathbf{x}) \approx C_i(\mathbf{x}) + \nabla C_i(\mathbf{x}) \Delta \mathbf{x} = 0. \quad (6)$$

$\Delta \mathbf{x}$ is restricted along the constraint gradient, a Lagrange multiplier has been used such that the correction

$$\Delta \mathbf{x} = \lambda \mathbf{M}^{-1} \nabla C_i(\mathbf{x}) \quad (7)$$

where $\mathbf{M} = \text{diag}\{m_1, \dots, m_n\}$ is the mass matrix. Then positions are updated after each constraint is processed. After iterations, a new velocity is computed according to the total constraint delta $\Delta \mathbf{v} = \Delta \mathbf{x} / \Delta t$. Constraints are kinematic restrictions in the form of equation and inequalities that constrain the relative motion of bodies.

Different constraints can be used to simulate different materials. For example, distance constraint can be used for stretchable bodies, bending constraint can be used for inextensible surfaces, bilateral bending constraint is important to simulate bending material such as cloth.

3.3 Shape matching constraint

Shape matching can be taken as a kind of constraints to simulate deformable objects, which is introduced by Müller et al. [30]. This meshless approach is able to simulate visually plausible elastic and plastic deformations. Meanwhile, it is very efficient and unconditionally stable [24,28], and is easy to be implemented.

Given two sets of positions $\mathbf{x}_0$ and $\mathbf{x}_i$, one represents the initial position, and the other represents the predicted position. Shape matching aims to find the rotation matrix $\mathbf{R}$ and target position g_i , which collectively control the transformation from $\mathbf{x}_0$ to $\mathbf{x}_i$. The mass centers of the initial shape and the actual shape can be defined as:

$$\mathbf{x}_{\text{cm}}^0 = \frac{\sum_i m_i \mathbf{x}_i^0}{\sum_i m_i}, \quad \mathbf{x}_{\text{cm}}^i = \frac{\sum_i m_i \mathbf{x}_i}{\sum_i m_i}. \quad (8)$$

Here m_i is the mass of point $\mathbf{x}_i$. Finding the optimal rotation matrix $\mathbf{R}$ can be relaxed by further finding the minimized item of a optimal linear transformation $\mathbf{A}$: $\sum_i m_i (\mathbf{A} \mathbf{q}_i - \mathbf{p}_i)$, which sets the derivatives with respect to all the coefficients of $\mathbf{A}$ to be zero, yields

$$\mathbf{A} = \left(\sum_i m_i \mathbf{p}_i \mathbf{q}_i^T \right) \left(\sum_i m_i \mathbf{q}_i \mathbf{q}_i^T \right)^{-1} = \mathbf{A}_{pq} \mathbf{A}_{qq}, \quad (9)$$

and $\mathbf{q}_i = \mathbf{x}_i^0 - \mathbf{x}_{\text{cm}}^0$ and $\mathbf{p}_i = \mathbf{x}_i - \mathbf{x}_{\text{cm}}$ are the relative locations with respect to their mass centers. The second term $\mathbf{A}_{qq}$ is a symmetric matrix that involves scaling without rotation, and $\mathbf{A}_{pq}$ controls the optimal rotation $\mathbf{A}_{pq} = \mathbf{R} \mathbf{S}$ ($\mathbf{S} = \sqrt{\mathbf{A}_{pq}^T \mathbf{A}_{pq}}$). Thus, the rotation matrix becomes $\mathbf{R} = \mathbf{A}_{pq} \mathbf{S}^{-1}$. Finally, the target position can be calculated as:

$$g_i = \mathbf{R} (\mathbf{x}_i^0 - \mathbf{x}_{\text{cm}}^0) + \mathbf{x}_{\text{cm}}. \quad (10)$$

To simulate deformable bodies, some extensions, such as linear deformations and quadratic deformations, will be used (please refer to [6,30] for more details). It can imitate varying materials by setting $\alpha \in [0, 1]$. Solid bodies' dynamics

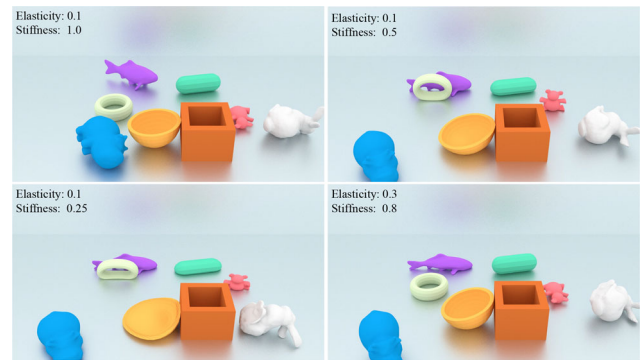


Fig. 1 Soft bodies' dynamic simulation. All the particles are computed by the FLIP algorithm and the shapes are maintained by the shape matching constraint. Objects in different scenarios but with different

are demonstrated in Fig. 1. In our method, we use quadratic deformation extension to enable the diverse deformation of solid body, such as twist and stretch [30]. So the optimal Eq. (9) can be converted to quadratic transformation:

$$\tilde{\mathbf{A}} = \left(\sum_i m_i \mathbf{p}_i \tilde{\mathbf{q}}_i^T \right) \left(\sum_i m_i \tilde{\mathbf{q}}_i \tilde{\mathbf{q}}_i^T \right)^{-1} = \tilde{\mathbf{A}}_{pq} \tilde{\mathbf{A}}_{qq}, \quad (11)$$

and $\tilde{\mathbf{q}} = [q_x, q_y, q_z, q_x^2, q_y^2, q_z^2, q_x q_y, q_y q_z, q_z q_x] \in \mathbb{R}^9$, and x, y, z represents the 3D directions, respectively. We use the combination $\beta \tilde{\mathbf{A}} + (1 - \beta) \tilde{\mathbf{R}}$ to compute the goal shape of solid, wherein $\tilde{\mathbf{R}} = [\mathbf{R}, \mathbf{0}, \mathbf{0}] \in \mathbb{R}^{3 \times 9}$, and β is an additional control parameter for linear deformation can be looked as elasticity.

4 Integrated framework

Both FLIP and PBD methods can satisfactorily simulate a variety of scenes, however, for some complicated phenomena such as fluid and deformable solid interaction, they fail to provide a pleasurable and convincing result without further improvement. The key of our approach is a unified Lagrangian framework that blends FLIP- and PBD-based shape matching constraint via natural and flexible coupling. At the numerical simulation level, we take advantages of MultiFLIP and shape matching. The dynamics of particles are solved by MultiFLIP solver, and the deformation of solid is handled by shape matching constraint. Our hybrid framework consists of three main components: (1) the uniform solution of all particles in a FLIP framework; (2) the coupling of FLIP simulation and shape matching constraint; (3) the correction of particles to ensure the accuracy and stability of FLIP solver. And the pipeline of our integrated framework is shown in Fig. 2, which illustrates it can support seamless unification of FLIP and shape matching.

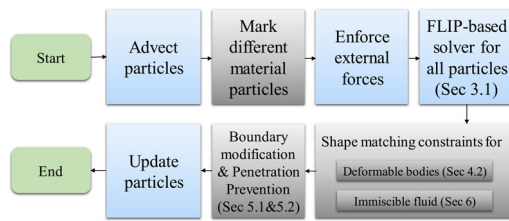


Fig. 2 The pipeline of our integrated framework. The blue squares represent the normal FLIP algorithms, and the gray ones represent our key improvements being incorporated into the FLIP framework

Algorithm 1 Implementation of our integrated framework for fluid–solid simulation.

```

1: Advect velocities of particles
2: Enforce external forces (gravity)
3: Verify fluid and solid particle flags via  $F_{\text{solid}}, F_{\text{fluid}}$ 
4: Map all the particles to grid  $\mathbf{u}_g^0 \leftarrow \mathbf{u}_p^0, \mathbf{x}_g^0 \leftarrow \mathbf{x}_p^0$ 
5: Compute level set  $\phi$  and velocity on grid  $\mathbf{u}_g$ 
6: Project  $\mathbf{u}_p \leftarrow \mathbf{u}_g, \mathbf{x}_p \leftarrow \mathbf{x}_g^0 + \mathbf{u}_g \Delta t$ 
7: if particle  $\in F_{\text{solid}}$  then
8:   Project shape matching constraint
9:   Compute target position  $\mathbf{g}$ 
10:  Update  $\mathbf{x}_p^* \leftarrow \alpha(\mathbf{g} - \mathbf{x}_p)$ 
11:  Update  $\mathbf{u}_p^* \leftarrow (\mathbf{u}_p, (\mathbf{x}_p^* - \mathbf{x}_p^0)/\Delta t)$ 
12: else Continue
13: Correct boundary conditions
14: Conduct the mechanism of penetration prevention
15: Update the velocities and positions of all particles
16: Update particles' flags

```

4.1 The unified algorithm

Take fluid–solid interaction for example, Algorithm 1 shows the main process within a time interval Δt . The texts marked in blue highlight our method's contributions, which improve the standard MultiFLIP simulation for hybrid fluid–solid simulation. During initialization, we mark different materials with different flags (e.g., $F_{\text{solid}}, F_{\text{fluid}}$). FLIP solver provides two positions ($\mathbf{x}_p^0$ and $\mathbf{x}_p$) for each solid particle, which can, respectively, be used as the original and predicted position for shape matching constraint. So we can apply the shape matching constraint to solid particles directly to simulate solid bodies movements (Fig. 1).

After solving the shape matching constraints, how to update target positions and how to correct velocities are the main concerns of a normal PBD method, in the interest of this paper's primary theme we will not detail these existing works, which can be found in [8] and [31].

4.2 Integrated formulations

When getting the particle velocity of solid, we compute a predicted position for this particle via

$$\mathbf{x}_p = \mathbf{x}_p^0 + \mathbf{u}_p \Delta t, \quad (12)$$

and $\mathbf{x}_p^0$ is the initial position, and $\mathbf{u}_g$ is the projected velocity from FLIP solver.

The target position is computed via

$$\mathbf{g} = \mathbf{C}_{\text{sm}} \times (\mathbf{x}_p^0 - \mathbf{x}_{\text{cm}}^0) + \mathbf{x}_{\text{cm}}, \quad (13)$$

where $\mathbf{x}_{\text{cm}}^0$ is the mass center, and the particle's position $\mathbf{x}_p$ is corresponding to space position $\mathbf{x}_i$ of shape matching constraint. Please note that, $\mathbf{C}_{\text{sm}}$ is not a unique function for shape matching constraints, there are several kinds of shape matching constraints, for the most basic form, $\mathbf{C}_{\text{sm}} = \mathbf{R}$, here $\mathbf{R}$ is rotation matrix. For linear form, $\mathbf{C}_{\text{sm}} = \beta \mathbf{A} + (1 - \beta) \mathbf{R}$. And in our case, we use the quadratic form combined with linear deformation as our constraint to compute target position, which enables twist and stretch:

$$\mathbf{C}_{\text{sm}} = \beta \tilde{\mathbf{A}} + (1 - \beta) \tilde{\mathbf{R}}. \quad (14)$$

The detailed definitions and derivation processes are described in Sect. 3.3, here particle's position $\mathbf{x}_p$ is corresponding to space position $\mathbf{x}_i$ of shape matching constraint. Thus, new particle position is computed as:

$$\mathbf{x}_p^* = \mathbf{x}_p^* + \alpha(\mathbf{g} - \mathbf{x}_p). \quad (15)$$

The computations in lines 10 and 11 of Algorithm 1 are the same as the traditional PBD method, wherein $\mathbf{x}_p^*$ will be updated toward the final position of each particle. And we update the new velocity $\mathbf{u}_p^*$ by a combination of FLIP velocity $\mathbf{u}_p$ and displacement–velocity:

$$\mathbf{u}_p^* = \frac{(\mathbf{u}_p + (\mathbf{x}_p^* - \mathbf{x}_p^0) / \Delta t)}{2}. \quad (16)$$

With the hybrid framework, we can simulate fluid interactions with various materials ranging from stiff solids to visco-plastic. However, since the new values are directly computed based on a PBD constraint, which involves no restrictions related to boundaries and FLIP fluid, the fluid particles may penetrate into solid, or the solid particles may move out of the defined boundary. Thus, boundary conditions need be added to solid and fluid particles to guarantee the stability, and the penetration prevention measurement should be introduced to guarantee physical reality and accuracy.

4.3 Interface tracking

To track the sharp interface between different materials, we, respectively, use two sets of independent marching-cube algorithms to capture the fluid and solid (or other material) surfaces. And we use the boundary particles of the solid to sample the surface of objects [1], which allows handling different shapes, such as lower-dimensional rigid bodies. The

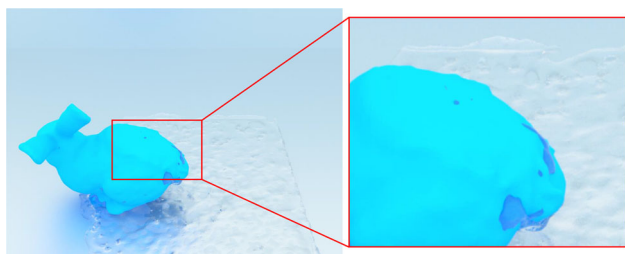


Fig. 3 Fluid interacts with solid and changes the solid shape of surface, while ensuring smoothness of the interface

flag of each particle indicates which system it belongs to, ensuring the accurate demarcation of different materials. As shown in Fig. 3, the deformable object changes its shape under the force effect of pouring water, while maintains a smooth surface between the fluid–solid interface.

5 Boundary handling and implementation details

With our hybrid framework, we can simulate fluid flows and imitate a solid simulator with shape matching constraint. However, since the naive coupling will suffer from serious numerical and stability problems [36], we will introduce our novel measurements into the hybrid framework to realize accurate simulations and rich applications.

5.1 Boundary handling for solid

In our hybrid solver, fluid grid interacting with boundary grid will rebound in an inverse direction. But for solid, since a set of particles are clustered together, if we take the same boundary conditions as fluid, local movements of the particles on boundary grids will lead to unrealistic global deformation, and affect the simulation stability. Thus, we define a new boundary condition for solid. For a set of solid particles, we allow transitory penetration into a virtual boundary grid to keep the global shape unchanged. The virtual boundary grid is defined as:

$$\begin{cases} \mathbf{P}_{\min} = \mathbf{X}_{\min} + \lambda h, \\ \mathbf{P}_{\max} = \mathbf{X}_{\max} - \lambda h. \end{cases} \quad (17)$$

Here $\mathbf{X}_{\max}$ is the maximum position of the boundary grid, and $\mathbf{X}_{\min}$ is the minimum position of the boundary grid, $\mathbf{P}_{\min}$ and $\mathbf{P}_{\max}$ are the virtual boundary locations, and h is the grid size of FLIP. $\lambda \in [0, 1]$ controls the shrinkage degree of virtual boundary. When $\lambda = 0$, the virtual boundary is equal to real boundary. In most of our experiments, we set $\lambda = 0.3$, which can effectively avoid penetrating into the real boundary and can handle the solid interactions well.

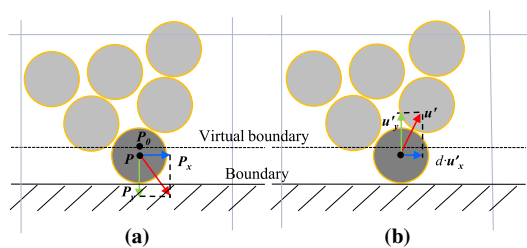


Fig. 4 Illustration of solid boundary conditions. When a solid particle penetrates into the virtual boundary (dotted line), we give it an inverse velocity depending on the position $\mathbf{P}$ and the position of virtual boundary grid $\mathbf{P}_0$ along normal direction, and multiply tangential velocity by a damping coefficient $d \in [0, 1]$ to imitate the effect of friction

As shown in Fig. 4a, $\mathbf{P}$ is the position of solid particle that penetrates into virtual boundary, and $\mathbf{P}_0$ is the position of the virtual boundary grid. In practice, when a solid hits the boundary, it will bounce along the normal direction and damp along the tangential direction under the force of friction. To imitate this process, we design a simplified model by introducing a damping coefficient d ($d \in [0, 1]$) in the tangential velocity. Since physics-based friction relates to many properties such as material, temperature, etc., which is beyond our research scope in this paper, the controllable damping parameter d for frictions can handle different boundaries with different frictions in a simple way, and effectively prevent solid from penetrating boundaries. When solid particle moves across the virtual boundary, we compute the inverse velocity of the whole solid (shown in Fig. 4b) by

$$\mathbf{u}' = \begin{cases} \mathbf{u}^* + \gamma \cdot N(\mathbf{P}_0 - \mathbf{P}), & \mathbf{P} < \mathbf{P}_{\min} \\ \mathbf{u}^* - \gamma \cdot N(\mathbf{P} - \mathbf{P}_0), & \mathbf{P} > \mathbf{P}_{\max} \end{cases} \quad (18)$$

Here γ is a bounce parameter that can be looked as the boundary elasticity, and N is the total number of solid particles, the right item of Eq. (18) can be looked as the velocity change after collision. Since solid particles' number relates to the mass of the solid body, the more particles a solid body has, the larger kinetic energy it carries.

5.2 Penetration prevention measurement

Inspired by the position correction idea used for smooth interface [9], we develop an additional algorithm to prevent particle penetration. First, after updating the particles' positions and velocities, we detect collision between solids and fluid particles. For each fluid particle P_i , we search solid particles that collide with P_i , and determine the collision positions. Second, we compute the relative velocity between the fluid and the solid particle by $\mathbf{v}_{\text{rel}} = \mathbf{v}_p - \mathbf{v}_c$, as shown in Fig. 5a. When the relative velocity points into the solid, i.e., $\mathbf{v}_{\text{rel}} \cdot \mathbf{n}_c$ is negative, penetration occurs. To prevent it we

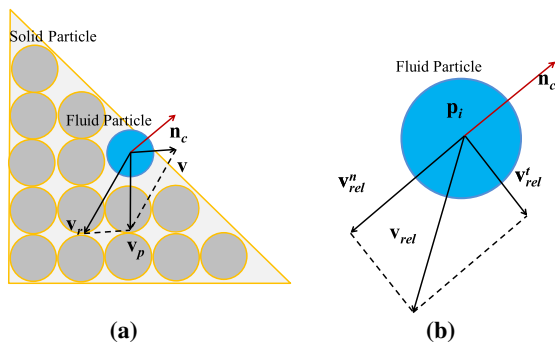


Fig. 5 Penetration prevention for particles, and $\mathbf{v}_{rel}^n$, $\mathbf{v}_{rel}^t$ are the relative velocities along the normal and tangential directions

impose a fluid–solid boundary condition on the velocity of fluid particle $\mathbf{v}_p$ as:

$$\mathbf{v}_p^{\text{new}} = \mathbf{v}_p - \mathbf{v}_{rel}^n = \mathbf{v}_p - (\mathbf{v}_{rel} \cdot \mathbf{n}_c)\mathbf{n}_c, \quad (19)$$

where the relative velocity along the normal direction $\mathbf{v}_{rel}^n$ is subtracted to make the particle's velocity equals to the solid's velocity in the normal direction.

In fact, the above velocity correction on fluid particle is equivalent to enforcing an impulse on the particle. To conserve momentum during the collision handling, we compute the collision force and impose this force on the solid via

$$\mathbf{f} = \frac{m_p(\mathbf{v}_{rel} \cdot \mathbf{n}_c)\mathbf{n}_c}{\Delta t}. \quad (20)$$

5.3 Melting simulation

Simulation of heat-based melting is also an interesting topic in computer graphics [15, 17], since our unified model is computed in a grid-based solver, it is quite convenient to combine a simplified heat transfer solver, which can drive the shape matching constraint to realize melting simulation dynamically.

To simulate melting, we initiate all the particles with a temperature attribute, the temperature update only depends upon the heat transfer. In each time step, temperature is mapped from particles to each grid cell with a weighting function:

$$\frac{T^{n+1} - T^n}{\Delta t} = b \left(\frac{\partial^2 T^{n+1}}{\partial x^2} + \frac{\partial^2 T^{n+1}}{\partial y^2} + \frac{\partial^2 T^{n+1}}{\partial z^2} \right), \quad (21)$$

where T^n is the given temperature field obtained in the last time step, T^{n+1} is the current one we need to update, and b is the corresponding thermal diffusivity parameter. After updating the temperature on grid T_g^{n+1} , we map the temperature changes of the grid to particles, and then update the particle temperature by $T_p^{n+1} = T_p^n + \Delta T_p$. When the temperature of a solid particle reaches to melting point, convert it to a fluid

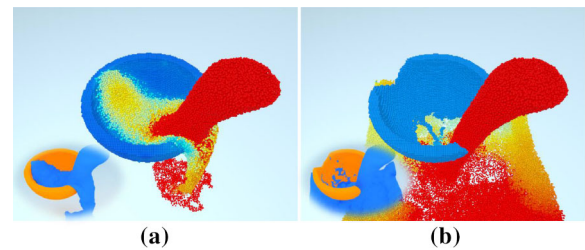


Fig. 6 Illustration of heat transfer among solids and liquids. The particles are colored according to temperatures (blue means low and red means high). And we give the render results at the bottom left corner in a close-up view

particle and alter it with the fluid's attributes, then release it from being confined as solid. Thus, this particle will become a free fluid particle, while the other solid particles still hold the integrity of solid constraints.

In each time step, since our shape matching method is performed after updating all the FLIP particles, the solid constraints are dynamically changed as the number and positions of the FLIP particles change, this realtime coupling manner avoids the problem of the constraint using the nonexistent position of “melting particles”. That is to say, our shape matching constraints can be dynamically coupled with the FLIP algorithm as geometry changes. Thus, the significant advantage of our constraints is that, shape matching constraints can dynamically change their controlling scopes as some of the solid particles become fluid particles during melting simulation without other artificial handling [23]. It ensures that local solid melts correctly, and the rest of solid parts will keep proper geometry and stable kinematics characteristics (i.e., moving and rotating). Figure 6 shows the heat transfer process of a melting bowl, with a part of the solid object melting due to the heat source on the left, the dynamic shape matching constraints ensure the correct geometry of the rest parts of the solid, and its movement and interaction are not affected by the particles outside the shape matching control scope.

6 Constraint for two-phase immiscible fluids

Since we use FLIP particles to compute all materials' kinetic equations, it is easy to implement miscible multiphase fluid simulation. And for immiscible two-phase fluid simulation, it is possible to be realized by using a sparse shape matching constraint to control the immiscible phase. In theory, if a soft body has very low stiffness, the soft body will present a fluid-like performance. But in practice, we found that the direct use of our proposed constraint for immiscible fluid will make the simulation unstable and lead to unrealistic results. So we further improve our proposed shape matching constraint to realize the stable and realistic simulation of immiscible two-

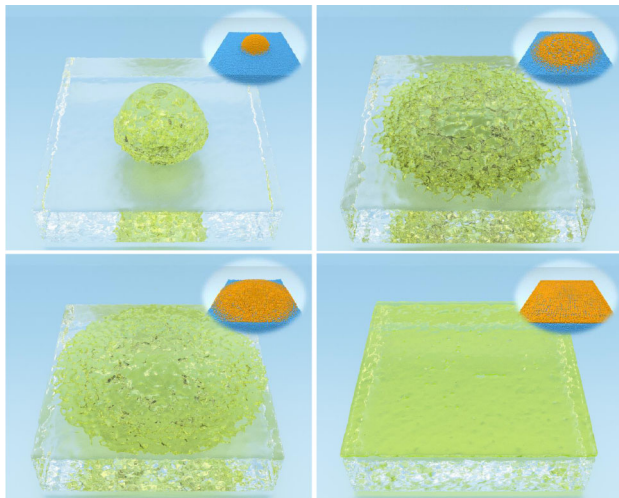


Fig. 7 Two-phase immiscible fluid simulation result, wherein a drop of oil falls into the water and interacts with each other. The particles' display is attached on the upper right corner

phase fluid. The shape matching constraint for immiscible fluid should satisfy the following two conditions: First, the constraint should maintain the immiscible phase fluid in a union set that can be separated from other phases; Second, the constraint should be relaxed and robust enough to ensure accurate fluid dynamics.

To imitate liquid's dynamics, we first minimize the parameter $\alpha = 0.05$ to reduce the stiffness, which makes the material soften as much as possible. When interacting with fluid particles, since rotation matrix $\mathbf{R}$ may make some of the particles rotate in an artificial manner, we increase β to 0.8, so that the linear transformation will play a dominant role. And we compute the shape center (the center of all the particles' positions) via $\mathbf{x}_{cs} = \sum x_i / N \cdot \mathbf{x}_{cs}$ is the geometric center of shape matching particles, and N is the number of the particles. Then we compute distance from each particle's relative position to the mass center $|\mathbf{x} - \mathbf{x}_{cm}|$, and compare it with $|c \cdot \mathbf{x}_{cs}|$ ($c \in [1, 3]$). If $|\mathbf{x} - \mathbf{x}_{cm}| > |c \cdot \mathbf{x}_{cs}|$, it means that, this particle moves too far away from the geometric center, which may affect other normal particles' movement in an unstable way. Thus, we change this particle to be a fluid particle and place it back to the center of mass $\mathbf{x} = \mathbf{x}_{cm} + \text{random}(\Delta x)$, so that it can move freely and eliminate the negative influences on other particles. The modification of the shape matching constraint is documented in Algorithm 2 (blue texts highlight our improved algorithm compared to Algorithm 1).

Although directly changing to fluid particle may suffer from the non-conservation problem, it can avoid error accumulation and maintain the stability for most of the particles. In practice, as shown in Fig. 7, this improved shape matching constraint achieves good performance when simulating two-phase immiscible fluids.

Table 1 Performance of experiments

Scenes	Total particles	Grid size	Avg. time/ timestep (ms)
Figure 1	390k	64^3	80.09
Figure 6	120k (fluid)	64^3	89.09
Figure 7	470k	64^3	152.44
Figure 8	80k (fluid)	64^3	82.06
Figure 9	8k	$64^2 \times 96$	31.05
Figure 11 (top)	2110k	$96^2 \times 192$	955.18
Figure 12	680k	$64^2 \times 128$	216.71
Figure 13a	460k	64^3	102.51
Figure 13b	460k	64^3	114.46
Figure 13c	460k	64^3	147.94

7 Experiments and evaluations

We implement our method on a PC with an NVIDIA GeForce GTX 1080 GPU, Intel Core I7 CPU using C++ and CUDA. And we demonstrate the capabilities of our hybrid framework via several simulation scenarios. Table 1 documents the performance of our experiments, indicating the high efficiency of our CUDA-based implementation.

7.1 CUDA-based numerical computation

We implement the entire modeling framework based on CUDA for efficiency. For each particle, we invoke a CUDA thread to calculate which grid cell it belongs to, and then use a CUDA thread for each grid cell to interpolate its velocity from particles.

7.2 Graphics results and discussion

We display several kinds of experiments in this section to demonstrate the advantages of our unified hybrid approach.

7.2.1 Solids simulation

Figure 1 shows the simulations of solid objects with range of elasticities and stiffnesses. Each body has a shape matching constraint, and all of its attributes are solved by MultiFLIP solver. This scenario illustrates that, with our hybrid framework, we can simulate most types of solid deformations that traditional PBD could accommodate.

7.2.2 Deformation

Figure 8 demonstrates the realistic rendering results of plastic body deformation. As the water flows over the surface of

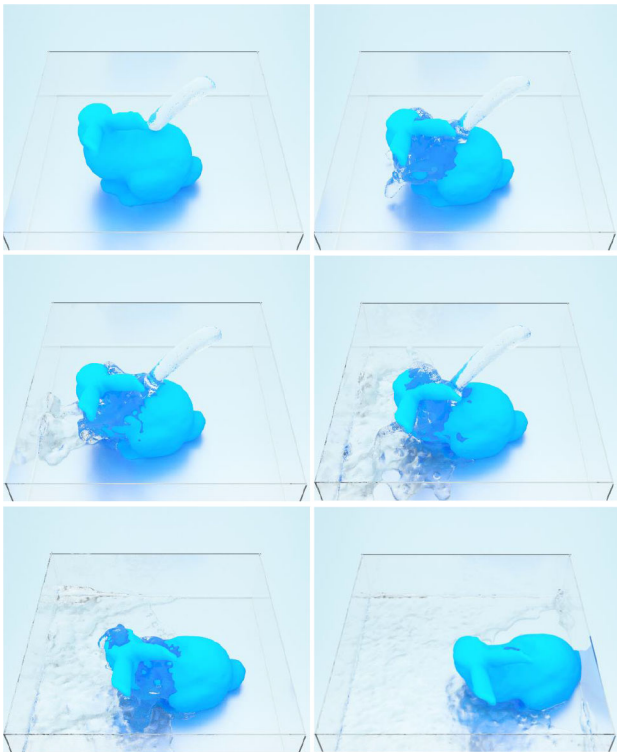


Fig. 8 Deformation. From top to bottom: Fluid pours on a deformable plastic bunny

the plastic bunny, solid particles are enforced to change the global topology, which is handled by the shape matching constraint. Meanwhile, with our surface tracking method,

we can ensure a relatively smooth surface of the deformable object.

7.2.3 Fluid–solid interaction

Figure 9 shows a scenario of fluid–solid interaction. We clamp one end of a pipe of plastic film and pour fluid into it. Fluid flows through the pipe and drops on the floor, and the pipe is deformed. With the penetration preventing measurement, though the pipe is quite thin (two layers of particles), there is no fluid particle penetrating through the pipe boundary, which demonstrates that our model has high accuracy.

Figures 10 and 11 show two scenarios of fluid–solid interactions. To compare our shape matching-FLIP approach with most related works that also use unified fluid particles to simulate solid–liquid coupling, we have combined Carlson’s and Macklin’s solid constraints into our GPU-based FLIP framework. As shown in Fig. 10, compared to *Rigid Fluid* of Carlson et al. [10] (it used a simple rigidity constraint to handle the motion of rigid objects), our method costs a little more time (as Table 2 shows), since time costs are very close, the extra time consumption can be negligible during simulation. However, our approach can be extended to simulate various phenomena, including soft bodies–fluid coupling and two-phase flows, which cannot be achieved by *Rigid Fluid* [10]. And compared to shape matching constraints of Macklin et al. [27], our constraints can handle wider-range materials’ coupling with fluid.

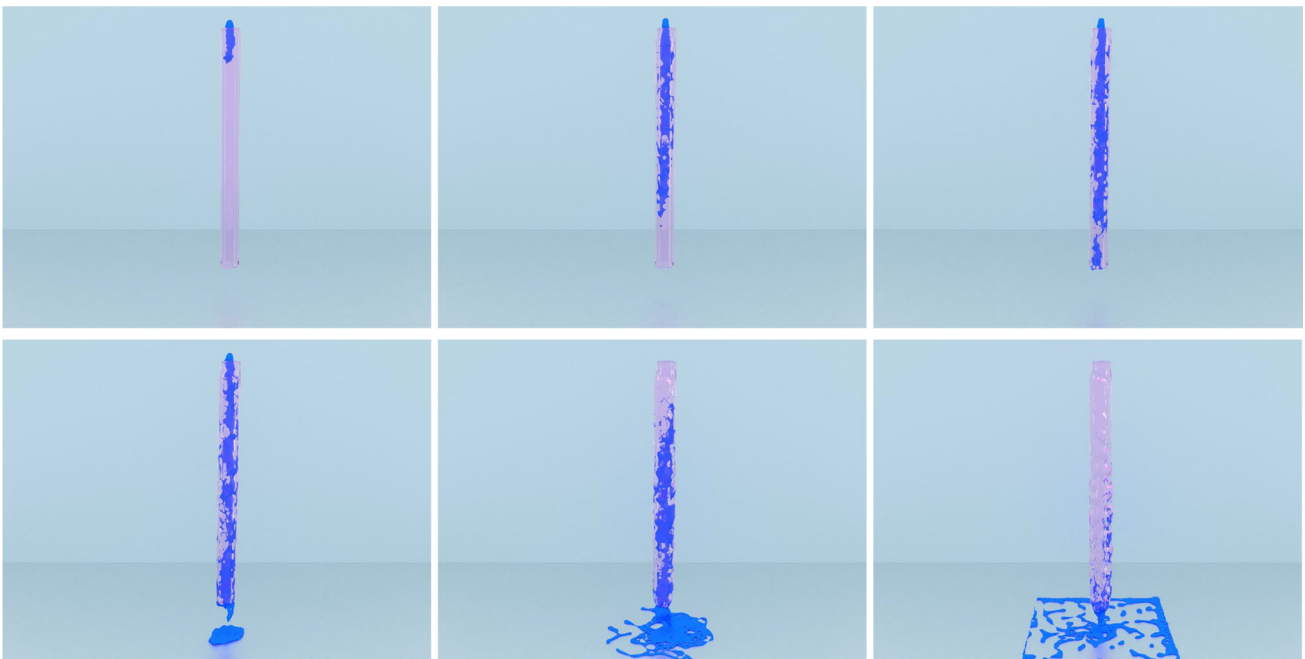


Fig. 9 Fluid–solid interaction. Fluid pours through a pipe of plastic film and flows through it and interacts with the pipe. The pipe of plastic film is quite thin, but no fluid particle can penetrate through the pipe

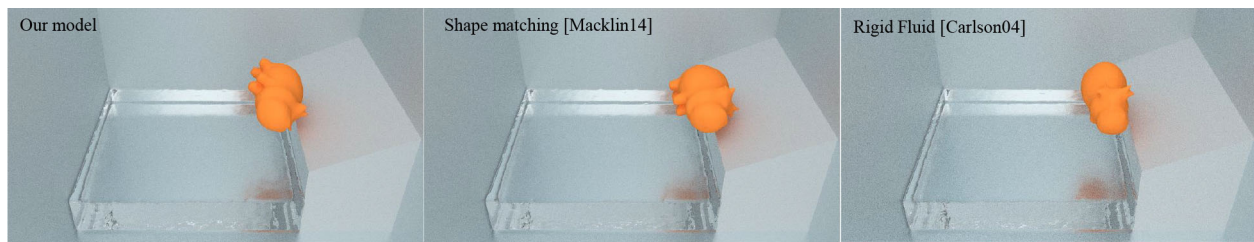


Fig. 10 Comparison of rigid body constraints. The first figure shows our shape matching model, the middle one uses the method of Macklin et al. [27], and the right figure displays rigid fluid model of Carlson et al. [10]. All the solid constraints are combined into GPU-based FLIP framework

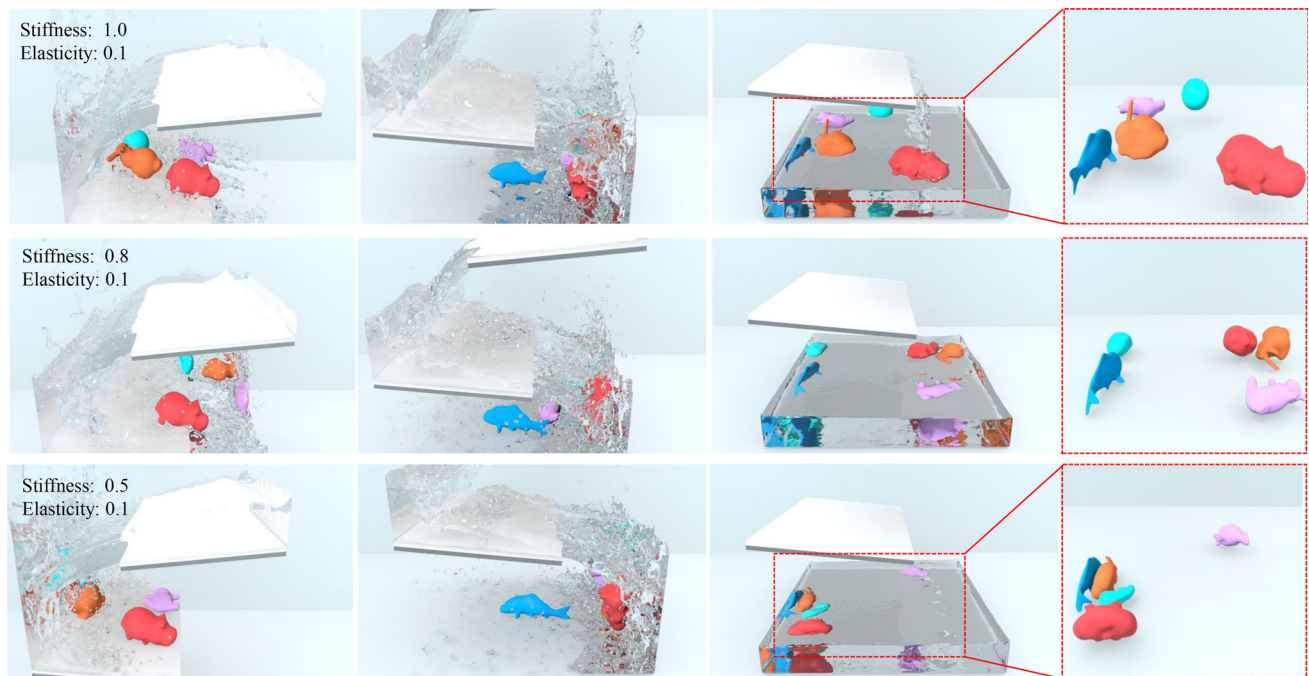


Fig. 11 Fluid–solid interactions with different stiffnesses. From top-left to bottom-right: Dam-breaking water flows through two static boards and interacts with several deformable plastic objects

Table 2 Performance comparison of different solid constraints

Scenes	Total particles	Grid size	Avg. time/ timestep (ms)
Figure 10 (left)	460k	64 ³	102.51
Figure 10 (middle)	460k	64 ³	104.47
Figure 10 (right)	460k	64 ³	87.67

Figure 11 shows dam-breaking water flows through two static boards and interacts with several deformable plastic objects. To imitate the buoyancy force of solid, we add an external force (with inverse gravity direction) to each solid particle. The buoyancy needs to be smaller than gravity, so that the solid particle will float on the water when interacting with fluid particles.

7.2.4 Melting

Figure 6 demonstrates fluid–solid coupling and melting scenario. We provide the particles' display with rendering result at bottom left corner, wherein the ice melts due to the heat absorption from the pouring hot water. When a solid particle's temperature rises to melting point, it is removed from shape matching constraint and becomes a fluid particle.

7.2.5 Two-phase immiscible fluids

Figures 7 and 12 show the simulation results of two-phase immiscible fluids. Figure 7 shows a drop of oil falls into water. And Fig. 12 shows oil-like material bunny couples with dam-breaking water. The immiscible fluid interacts with water and finally floats on the surface. To imitate the buoyancy force of solid, we add an external force (with inverse gravity

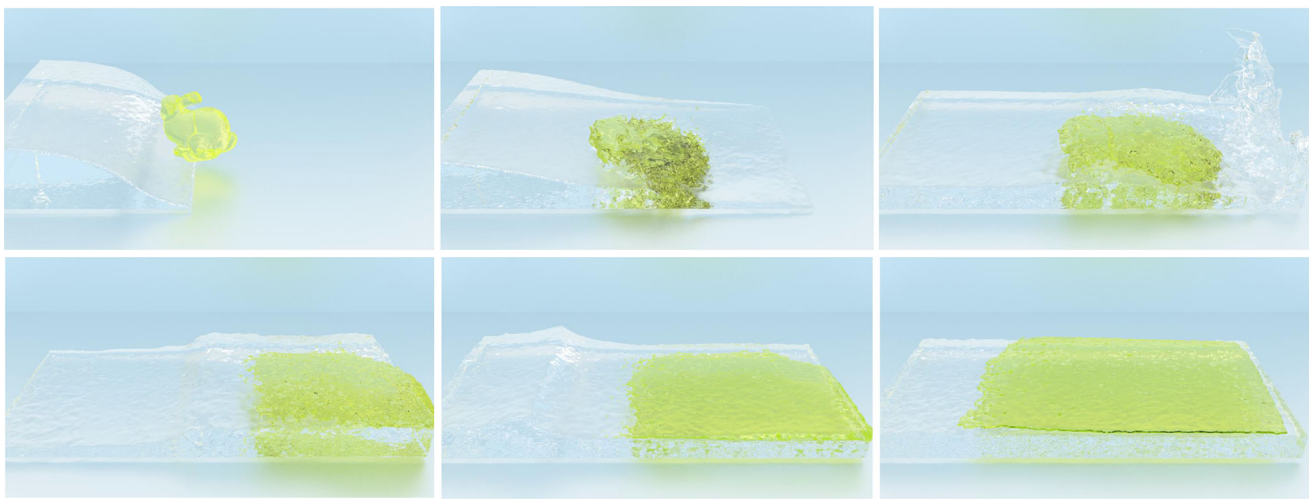


Fig. 12 Two-phase immiscible fluid simulation. Bunny has oil-like material and interacts with water. Though these two kinds of fluids interact and mix as the water flows, they finally separate, and the oil-like fluid floats over the water surface

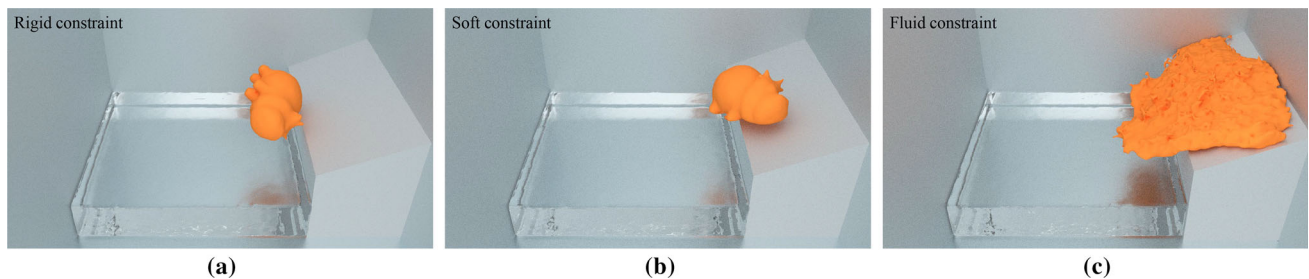


Fig. 13 Variable constraints for different materials. **a** Rigid–fluid coupling. **b** Soft body interacts with boundary and water (stiffness: 0.5 and elasticity: 0.1). **c** Immiscible two-phase flow

direction) to oil particles, and set their initial density equal to 0.8 times of water. The buoyancy needs to be smaller than gravity, so that the solid particle will float on the water when interacting with fluid particles.

7.2.6 Variable materials coupling with water

To illustrate our shape matching constraints can provide a flexible and fast way for variable material–fluid coupling simulations. Figure 13 displays the expandability of our proposed shape matching constraints. In a same scenario, we can simulate different phenomenon through different parameters of shape matching constraints. (a) Shows the rigid–fluid coupling with stiffness of solid is 1.0. (b) Shows soft body drops on boundary and interacts with water, here we set the elasticity to 0.1 and stiffness is 0.5. (c) Shows immiscible two-phase flow coupling with our improved shape matching method (comparison with classical shape matching of PBD can be found in our attached video).

7.3 Limitation

Even though the tight coupling of FLIP and shape matching models, together with the custom-designed algorithms, enables more flexibility to empower simulation results, one limitation of our method is the non-conservation problem pertinent to our improved shape matching constraint. Although the numerical dissipation can be ignored in the vast majority of cases, it expects to deteriorate as the complexity of details increases. For some complicated phenomena such as splash and turbulence, we will have to consider tradeoff between preserving the details and ensuring the stability of the algorithm. And another limitation is that we cannot handle elastic objects interacting with fluid well because the water force is continual and non-uniform, which is hard for our current shape matching method to compute feedback force of elastic deformation.

8 Conclusion and future work

In this paper, we have detailed a novel framework for fluid-relevant phenomena simulation by integrating the FLIP method and shape matching constraint. Our shape matching-FLIP-coupled approach is distinct from exiting techniques, which provides a flexible and fast manner for soft material–fluid coupling simulation with wide range stiffness spanning from rigid body to viscous liquid. The novel technical elements also include the boundary handling algorithm, measurement for penetration prevention, and improved constraints for two-phase fluid interaction. We have illustrated various types of experiments and demonstrated the advantages of our unified framework.

At present, although our integrated framework has already successfully simulated numerous fascinating scenes, our shape matching coupled framework is not perfect, we shall continue to expand our constraints to enable more complex graphics applications with high visual fidelity in the near future, such as fluid interacts with large elastic objects and various kind of multiphase fluid.

Acknowledgements This research is supported in part by National Natural Science Foundation of China (Nos. 61672077 and 61532002), National Key R&D Program of China (No. 2017YFF0106407), Applied Basic Research Program of Qingdao (No. 161013xx), National Science Foundation of USA (Nos. IIS-0949467, IIS-1047715, and IIS-1049448) and the Academic Excellence Foundation of BUAA for Ph.D. Students.

References

- Akinci, N., Ihmsen, M., Akinci, G., Solenthaler, B., Teschner, M.: Versatile rigid-fluid coupling for incompressible SPH. *ACM Trans. Graph.* **31**(4), 62:1–62:8 (2012)
- Akinci, N., Cornelis, J., Akinci, G., Teschner, M.: Coupling elastic solids with smoothed particle hydrodynamics fluids. *Comput. Anim. Virtual Worlds* **24**(3–4), 195–203 (2013)
- Allard, J., Courtecuisse, H., Faure, F.: Implicit FEM and fluid coupling on GPU for interactive multiphysics simulation. *ACM SIGGRAPH 2011 Talks*, pp. 52:1–52:1 (2011)
- Ando, R., Thurey, N., Tsuruno, R.: Preserving fluid sheets with adaptively sampled anisotropic particles. *IEEE Trans. Vis. Comput. Graph.* **18**(8), 1202–1214 (2012)
- Batty, C., Bertails, F., Bridson, R.: A fast variational framework for accurate solid–fluid coupling. *ACM Trans. Graph.* **26**(3), 100 (2007)
- Bender, J., Müller, M., Macklin, M.: Position-based simulation methods in computer graphics. In: *Tutorial Proceedings of Eurographics* (2015)
- Bender, J., Dan, K., Charrier, P., Weber, D.: Position-based simulation of continuous materials. *Comput. Graph.* **44**, 1–10 (2014)
- Bender, J., Müller, M., Otaduy, M.A., Teschner, M., Macklin, M.: A survey on positionbased simulation methods in computer graphics. *Comput. Graph. Forum* **33**(6), 228251 (2014)
- Boyd, L., Bridson, R.: MultiFLIP for energetic two-phase fluid simulation. *ACM Trans. Graph.* **31**(2), 16:1–16:12 (2012)
- Carlson, M., Mucha, P.J., Turk, G.: Rigid fluid: animating the interplay between rigid bodies and fluid. *ACM Trans. Graph.* **23**(3), 377–384 (2004)
- Clausen, P., Wicke, M., Shewchuk, J.R., O’Brien, J.F.: Simulating liquids and solid–liquid interactions with lagrangian meshes. *ACM Trans. Graph.* **32**(2), 17 (2013)
- Cornelis, J., Bender, J., Gissler, C., Ihmsen, M., Teschner, M.: An optimized source term formulation for incompressible SPH. *Vis. Comput* (2018). <https://doi.org/10.1007/s00371-018-1488-8>
- Cornelis, J., Ihmsen, M., Peer, A., Teschner, M.: IISPH–FLIP for incompressible fluids. *Comput. Graph. Forum* **33**(2), 255–262 (2014)
- Ferstl, F., Ando, R., Wojtan, C., Westermann, R., Thurey, N.: Narrow band flip for liquid simulations. *Int. J. Numer. Methods Fluids* **35**(2), 225–232 (2016)
- Gagnon, J., Paquette, E.: Procedural and interactive icicle modeling. *Vis. Comput.* **27**(6–8), 451–461 (2011)
- Gao, Y., Li, S., Qin, H., Hao, A.: A novel fluid-solid coupling framework integrating flip and shape matching methods. In: *Proceedings of the Computer Graphics International Conference*, p. 11. ACM (2017)
- Gao, Y., Li, S., Yang, L., Qin, H., Hao, A.: An efficient heat-based model for solid–liquid–gas phase transition and dynamic interaction. *Graph. Models* **94**, 14–24 (2017)
- Gerszewski, D., Bargteil, A.W.: Physics-based animation of large-scale splashing liquids. *ACM Trans. Graph.* **32**(6), 185:1–185:6 (2013)
- Ihmsen, M., Orthmann, J., Solenthaler, B., Kolb, A., Teschner, M.: SPH fluids in computer graphics. *Eurographics 2014-State of the Art Reports*, pp. 21–42 (2014)
- Keiser, R., Adams, B., Gasser, D., Bazzi, P., Dutre, P., Gross, M.: A unified Lagrangian approach to solid–fluid animation, vol. **2005**, pp. 125–148 (2005)
- Lenaerts, T., Dutre, P.: Mixing fluids and granular materials. *Comput. Graph. Forum* **28**(2), 213–218 (2009)
- Lenaerts, T., Adams, B., Dutre, P.: Porous flow in particle-based fluid simulations. *ACM Trans. Graph.* **27**(3), 15–19 (2008)
- Lii, S.Y., Wong, S.K.: Ice melting simulation with water flow handling. *Vis. Comput.* **30**(5), 531–538 (2014)
- Ller, M., Chentanez, N., Kim, T.Y., Macklin, M.: Strain based dynamics. In: *Proceedings of ACM SIGGRAPH/Eurographics Symposium Computer Animation*, pp. 149–157 (2014)
- Macklin, M., Müller, M., Chentanez, N.: Xpbd: Position-based simulation of compliant constrained dynamics (2016)
- Macklin, M.: Position based fluids. *ACM Trans. Graph.* **32**(32), 104:1–104:12 (2013)
- Macklin, M., Ller, M., Chentanez, N., Kim, T.Y.: Unified particle physics for real-time applications. *ACM Trans. Graph.* **33**(4), 1–12 (2014)
- Martinek, M., Grosso, R., Greiner, G.: Interactive partial 3d shape matching with geometric distance optimization. *Vis. Comput.* **31**(2), 223–233 (2015)
- Misztal, M.K., Erleben, K., Bargteil, A., Fursund, J., Christensen, B., Bærentzen, J.A., Bridson, R.: Multiphase flow of immiscible fluids on unstructured moving meshes, pp. 97–106. In: *Proceedings of ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2012)
- Müller, M., Heidelberger, B., Teschner, M., Gross, M.: Meshless deformations based on shape matching. *ACM Trans. Graph.* **24**(3), 471–478 (2005)
- Müller, M., Heidelberger, B., Hennix, M., Ratcliff, J.: Position based dynamics. *J. Vis. Commun. Image Represent.* **18**(2), 109–118 (2007)
- Peer, A., Teschner, M.: Prescribed velocity gradients for highly viscous SPH fluids with vorticity diffusion. *IEEE Trans. Vis. Comput. Graph.* **23**, 2656–2662 (2016)

33. Selino, A., Jones, M.: Large and small eddies matter: animating trees in wind using coarse fluid simulation and synthetic turbulence. *Comput. Graph. Forum* **32**(1), 75–84 (2013)
34. Shao, X., Zhou, Z., MagnenatThalmann, N., Wu, W.: Stable and fast fluid–solid coupling for incompressible SPH. *Comput. Graph. Forum* **34**(1), 191–204 (2015)
35. Tournier, M., Nesme, M., Gilles, B.: Stable constrained dynamics. *ACM Trans. Graph.* **34**(4), 1–10 (2015)
36. Wong, T.H., Leach, G., Zambetta, F.: An adaptive octree grid for GPU-based collision detection of deformable objects. *Vis. Comput.* **30**(6–8), 729–738 (2014)
37. Yan, X., Jiang, Y.T., Li, C.F., Martin, R.R., Hu, S.M.: Multi-phase SPH simulation for interactive fluids and solids. *ACM Trans. Graph.* **35**(4), 79 (2016)
38. Yang, L., Li, S., Hao, A., Qin, H.: Realtime two-way coupling of meshless fluids and nonlinear fem. *Comput. Graph. Forum* **31**(7), 2037–2046 (2012)
39. Yang, L., Li, S., Hao, A., Qin, H.: Hybrid particle-grid modeling for multi-scale droplet/spray simulation. *Comput. Graph. Forum* **33**(7), 199–208 (2014)
40. Zhu, Y., Bridson, R.: Animating sand as a fluid. *ACM Trans. Graph.* **24**(3), 965–972 (2005)



Yang Gao is a Ph.D. student in School of Computer Science and Engineering at Beihang University. He is currently doing his academic research at the State Key Laboratory of Virtual Reality Technology and Systems. His research interests include medical image processing, computer graphics, physics-based modeling and simulation, particularly in fluid.



Shuai Li received the Ph.D. degree in computer science from Beihang University. He is currently an associate professor at the State Key Laboratory of Virtual Reality Technology and Systems, Beihang University. His research interests include computer graphics, physics-based modeling and simulation, virtual surgery simulation, computer vision, and medical image processing.



Hong Qin received the B.S. and M.S. degrees in computer science from Peking University. He received the PhD degree in computer science from the University of Toronto. He is a professor of computer science in the Department of Computer Science, Stony Brook University. His research interests include geometric and solid modeling, graphics, physics-based modeling and simulation, computer-aided geometric design, visualization, and scientific computing.



Yinghao Xu is a master student at the State Key Laboratory of Virtual Reality Technology and Systems, Beihang University. His research interests include computer graphics, physics-based modeling and simulation.



Aimin Hao is a professor in Computer Science School and the Associate Director of State Key Laboratory of Virtual Reality Technology and Systems at Beihang University. He received his B.S., M.S., and Ph. D. in Computer Science at Beihang University. His research interests are in virtual reality, computer simulation, computer graphics, geometric modeling, image processing, and computer vision.