DOI: 10.1111/cgf.12324
EUROGRAPHICS 2014 / B. Lévy and J. Kautz
(Guest Editors)

Volume 33 (2014), Number 2

IISPH-FLIP for incompressible fluids

Jens Cornelis   Markus Ihmsen   Andreas Peer   Matthias Teschner

University of Freiburg, Germany

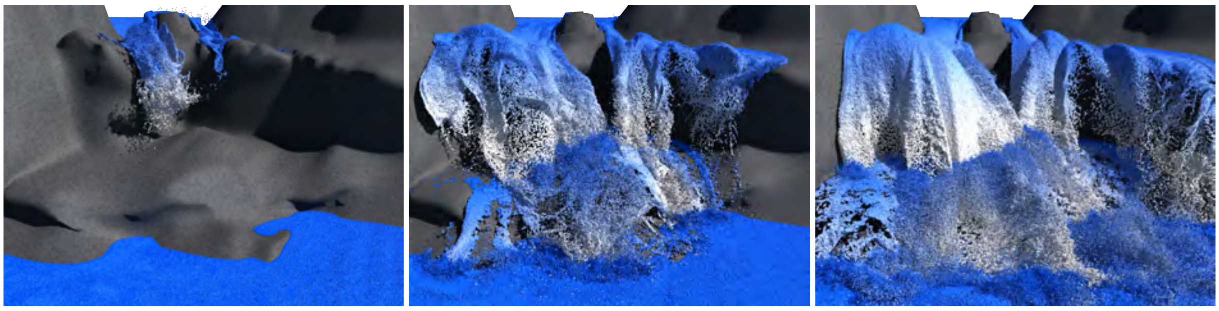


Figure 1: *Scene with 160 million FLIP particles. Particles are color coded with respect to velocity.*

Abstract

We propose to use Implicit Incompressible Smoothed Particle Hydrodynamics (IISPH) for pressure projection and boundary handling in Fluid-Implicit-Particle (FLIP) solvers for the simulation of incompressible fluids. This novel combination addresses two issues of existing SPH and FLIP solvers, namely mass preservation in FLIP and efficiency and memory consumption in SPH. First, the SPH component enables the simulation of incompressible fluids with perfect mass preservation. Second, the FLIP component efficiently enriches the SPH component with detail that is comparable to a standard SPH simulation with the same number of particles, while improving the performance by a factor of 7 and significantly reducing the memory consumption. We demonstrate that the proposed IISPH-FLIP solver can simulate incompressible fluids with a quantifiable, imperceptible density deviation below 0.1%*. We show large-scale scenarios with up to 160 million particles that have been processed on a single desktop PC using only* 15*GB of memory. One- and two-way coupled solids are illustrated.*

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

1. Introduction

The Fluid-Implicit-Particle (FLIP) concept is widely used for visual effects and represents the state-of-the-art for high-quality, versatile, robust, and efficient fluid simulations [ATW13]. FLIP has been proposed by Brackbill and Ruppel [BR86, BKR88] and an adapted version has been introduced to the graphics community by Zhu and Bridson [ZB05, Bri08]. In FLIP, the fluid is represented with particles, while the projection step is performed on an auxiliary Eulerian grid. Versatile effects have been integrated into FLIP, e.g., two-way coupling with solids [BBB07] and multi-phase flows [BB12]. In terms of memory consumption and computation time, FLIP has conceptual advantages compared to other particle methods, e.g., Smoothed Particle Hydrodynamics (SPH). First, according to [ATT12], FLIP handles larger time steps than SPH. Second, the original FLIP concept does not need to find, store, and process the

© 2014 The Author(s)
Computer Graphics Forum © 2014 The Eurographics Association and John Wiley & Sons Ltd. Published by John Wiley & Sons Ltd.

comparatively large number of neighboring particles. Third, FLIP particles can be flexibly processed. They can easily be generated or deleted to preserve fluid sheets, split or merged to account for adaptive grids, or repositioned to account for uneven spatial particle distributions [ATT12, ATW13]. All these operations are designed to have as little effect on the fluid dynamics as possible, while they would pose severe stability issues in SPH simulations.

Existing FLIP solvers employ particles to account for advection. Pressure projection, body forces and boundary handling [BBB07], however, are generally handled on fixed [ZB05, ATT12] or adaptive Eulerian grids [BR86, ATW13]. As an alternative to Eulerian grids, we propose to perform the pressure computation on a Lagrangian grid, i.e., SPH particles using IISPH [ICS*13]. Such an IISPH-FLIP combination is characterized by an accurate mass preservation, a typical issue in FLIP solvers that work with Eulerian grids. While it is rather involved to avoid and even to quantify volume changes due to mass loss in Eulerian pressure projection, e.g. [CM12], we show that volume changes due to density deviations can be exactly quantified and efficiently minimized in the proposed solver. Also, potential mass gain due to a reseeding of FLIP particles as discussed in [GB13] is not an issue, as mass conservation is exclusively handled by the IISPH pressure projection. In all presented IISPH-FLIP experiments, the volume change is at an imperceptible level below 0.1%.

The proposed technique is a novel variant of the FLIP concept. Velocities of FLIP particles are interpolated onto the auxiliary Lagrangian grid, i.e., SPH particles, where a divergence-free velocity field is computed by solving a pressure Poisson equation with IISPH. Velocity differences from consecutive steps are interpolated back and added to the velocity of FLIP particles which are finally advected.

As the proposed solver integrates IISPH into a FLIP solver, its implementation follows the FLIP concept. However, the positions of pressure samples, i.e., SPH particle positions are not fixed, but advected with their velocity. The interpolation between SPH and FLIP particles is performed as in FLIP. Pressure projection is computed with IISPH [ICS*13].

Existing FLIP approaches produce highly detailed simulations with a large number of FLIP particles on coarse Eulerian grids. The same effect holds for IISPH-FLIP, where the interpolation between coarsely sampled SPH particles and a larger number of FLIP particles enriches the fluid simulation.

This leads to a second perspective onto the proposed solver. It can also be seen as an SPH or IISPH extension, where the SPH sampling is enriched with additional FLIP particles and the FLIP interpolation strategy. Experimental results show that the overall flow and detail in IISPH-FLIP are comparable to IISPH with the same number of particles, but with a significantly improved performance and reduced memory requirements compared to IISPH. Scenes with up to 160 million FLIP particles have been computed on a single PC with 15GB of memory (Fig. 1).

2. Related work

The combination of different components to a new solver is rather popular in graphics applications and it is justified by benefits of one solver that addresses drawbacks of the other solver in certain scenarios and vice versa for other scenarios. A popular approach is the combination of SPH particles with grid solvers, e.g., [LTKF08,LHK09,GLHB09,ZYF10]. These approaches represent parts of the fluid domain with SPH where appropriate, while other parts of the domain are solved on a grid. Both domains are usually two-way coupled. In contrast to these approaches, we propose to represent the entire fluid domain with SPH particles and to also enrich the entire domain with FLIP particles.

Other approaches combine multiple resolutions. E.g., [LZF10] computes pressure on a coarse grid which is refined on a fine grid if required. In the context of SPH, [RWT11] also computes pressure on a coarse grid. The resulting pressure forces are combined with pressure forces based on WCSPH [BT07] or PCISPH [SP09] for high-resolution SPH particles. In contrast to these approaches, we propose to compute a low-resolution pressure field on SPH particles with IISPH which is combined with high-resolution FLIP particles.

Our approach works with two spatial resolutions. Nevertheless, it is less related to spatially adaptive approaches, e.g., [APKG07, SG11, ZLC*13], where different parts of the domain are represented with different levels of detail. In contrast, our approach samples the entire fluid with low-resolution SPH particles and high-resolution FLIP particles.

Our approach is closely related to FLIP, e.g., [BR86, ZB05, ATW13, GB13], where we propose a novel variant. We show that IISPH can be used for pressure projection, efficiently addressing the mass and volume preservation in standard FLIP. Our approach is also closely related to SPH, e.g., [Mon92, DC96, MCG03, Mon05, BT07, SP09, HLWW12, ICS*13, IOS*14]. Here, we show that our approach efficiently enriches standard IISPH with detail.

It is well accepted that the performance of SPH solvers based on state equations is low when small density deviations need to be enforced as discussed in, e.g., [SP09]. However, predictive-corrective formulations as proposed in [SP09, HLWW12] alleviate this issue and [ICS*13] further showed that IISPH outperforms predictive-corrective formulations [SP09, BLS12]. A discussion of further IISPH properties that motivate its use in the pressure projection of FLIP is given in Sec. 3.3.

3. Basics

We propose to use the IISPH concept for the pressure computation within a FLIP framework, which can also be seen as an SPH extension with FLIP particles. For completeness, both components are briefly introduced.

3.1. FLIP

Standard FLIP [BR86] works with FLIP particles and an auxiliary coarse Eulerian grid. The velocity at a grid node is initialized with the weighted average of adjacent FLIP particle velocities plus the velocity change due to body forces. Then, pressure projection computes velocity changes that result in a divergence-free velocity field on the grid. The respective pressure Poisson equation is commonly solved with, e.g., Conjugate Gradient using a Modified Incomplete Cholesky preconditioner [Bri08, GB13] (MICCG), while boundary handling can be incorporated with [BBB07]. The velocity change due to body forces and pressure projection is interpolated back from the grid to the FLIP particles and added to their previous velocity. This resulting particle velocity is commonly blended with the interpolated total velocity of the grid as proposed in [Bri08]. Then, FLIP particles are advected. The particle motion is clipped at solid boundaries.

3.2. IISPH

IISPH [ICS*13] is an incompressible SPH variant which implements the standard projection method [Cho68] in SPH. Velocity changes due to body forces and viscosity are added to SPH particle velocities. This intermediate velocity is corrected by solving a pressure Poisson equation and applying the respective velocity change that results from the pressure gradient. Finally, SPH particles are advected with the corrected velocity. IISPH uses a density invariance condition in the source term of the Poisson equation to avoid volume drift. As volume drift is not an issue in IISPH, a low number of solver iterations is sufficient. Boundary handling is incorporated in IISPH based on [AIA*12, IAGT10, ACAT13].

3.3. IISPH properties

IISPH uses a density invariance condition in the source term of the Poisson equation which enables the correction of previously introduced density or volume deviations. This is in contrast to the divergence-free condition, where deviations at previous timesteps are not considered. For the divergence-free condition, the resulting volume drift can only be counteracted with a comparable large number of solver iterations. In contrast, volume drift is not an issue in IISPH and a low number of solver iterations is sufficient. [ICS*13] discusses a scenario with up to 40 million pressure samples (SPH particles) and a timestep of 0.025s, where the IISPH solver requires only 4.1 iterations on average to solve the Poisson equation with an imperceptible density deviation. IISPH can be implemented in a matrix-free way with a memory footprint of seven scalar values per particle. These properties motivate IISPH as an interesting candidate for an efficient volume-preserving pressure projection in FLIP. Performance comparisons to the pressure projection in standard FLIP [Bri08] are discussed in Sec. 5.

4. IISPH-FLIP

FLIP has been introduced as a concept to handle the advection term in Eulerian fluid simulations. Thus, the projection step in standard FLIP is computed on an Eulerian grid. It is, however, also possible to employ a set of Lagrangian sample positions, e.g., SPH particles, for the pressure computation in FLIP as briefly mentioned in [BKR88]. This combination is not obviously useful as advection can be trivially handled in Lagrangian approaches without FLIP particles. Still, the combination of Lagrangian SPH and FLIP particles has the above-mentioned benefits regarding mass preservation in FLIP and efficient computation of detail in SPH. In the following, we describe the concept of the proposed IISPH-FLIP solver followed by implementation details.

4.1. Concept

Our approach works with high-resolution FLIP particles at positions $\mathbf{x}_F$ with velocities $\mathbf{v}_F$ and low-resolution SPH particles at positions $\mathbf{x}_S$ with velocities $\mathbf{v}_S$. Following [ZB05], eight FLIP particles are used per pressure sample, i.e., SPH particle. If not relevant, the time index is omitted. Further, a kernel function $W_{ij} = W(\|\mathbf{x}_i - \mathbf{x}_j\|, 2h)$ with compact support is used that depends on the distance of FLIP or SPH particles i and j. We employ a cubic spline function with a support of $2h$ [Mon92]. The spacing of SPH particles is h.

For each time step, the intermediate velocity $\mathbf{v}^*_{i_S}$ of an SPH particle i is reinitialized by interpolating the velocities $\mathbf{v}_{j_F}$ of neighboring FLIP particles j using the kernel function $W_{i_S j_F}$:

$$\mathbf{v}^*_{i_S} = \frac{1}{\sum_{j_F} W_{i_S j_F}} \sum_{j_F} \mathbf{v}_{j_F} W_{i_S j_F}. \qquad (1)$$

In contrast to the standard SPH interpolation of field quantities, the sum has to be normalized. This is due to the fact that the actual volume or density of FLIP particles is not known. As SPH particles carry their velocity, this initialization is not necessarily required. The procedure, however, reduces the numerical diffusion of the IISPH simulation. Additionally, the resulting cohesion effects are similar to standard FLIP and we therefore prefer this additional interpolation.

Now, IISPH processes the SPH particles. Following the projection or splitting concept, IISPH computes the pressure at SPH particles by solving a Poisson equation and velocities $\mathbf{v}_{i_S}(t+\Delta t)$. Boundary handling is incorporated into IISPH. The tolerated density deviation is always set to 0.1% in our

Algorithm 1 IISPH-FLIP update

```
for all SPH particle $i$ do
    search FLIP neighbors
    search and store SPH neighbors
    compute $\mathbf{v}^*_{i_S}$ (1)
compute $\mathbf{v}_{i_S}(t+\Delta t)$ with IISPH
for all FLIP particle $i$ do
    search SPH neighbors
    compute $\mathbf{v}^{PIC}_{i_F}$ (2)
    compute $\mathbf{v}^{FLIP}_{i_F}$ (3)
    compute $\mathbf{v}_{i_F}(t+\Delta t)$ (4)
    compute $\mathbf{x}_{i_F}(t+\Delta t)$ (5)
for all SPH particle $i$ do
    compute $\mathbf{x}_{i_S}(t+\Delta t)$ (6)
```

scenarios. In contrast to standard IISPH, artificial viscosity is not required.

In the next step, velocities of SPH particles are interpolated back onto FLIP particles. Here, we follow the standard technique, e.g., [ZB05, Bri08, ATT12, GB13], where FLIP and PIC interpolations are blended. While FLIP interpolates velocity differences, PIC interpolates total velocities onto the FLIP particles. Considering all SPH particles within the kernel support, the interpolated PIC velocity at a FLIP particle is computed as

$$\mathbf{v}^{PIC}_{i_F} = \frac{1}{\sum_{j_S} W_{i_F j_S}} \sum_{j_S} \mathbf{v}_{j_S}(t+\Delta t) W_{i_F j_S}. \qquad (2)$$

Accordingly, the interpolated FLIP velocity is computed as

$$\mathbf{v}^{FLIP}_{i_F} = \mathbf{v}_{i_F}(t) + \frac{1}{\sum_{j_S} W_{i_F j_S}} \sum_{j_S} (\mathbf{v}_{j_S}(t+\Delta t) - \mathbf{v}^*_{j_S}) W_{i_F j_S}. \quad (3)$$

Both velocities are blended with a regularization parameter $\alpha = min(\frac{6\Delta t\nu}{h^2}, 1)$ that is related to the time step Δt, the SPH particle spacing h and the fluid viscosity ν which is set to 0.01 in our scenarios:

$$\mathbf{v}_{i_F}(t+\Delta t) = \alpha \mathbf{v}^{PIC}_{i_F} + (1-\alpha)\mathbf{v}^{FLIP}_{i_F}. \qquad (4)$$

Finally, positions of FLIP and SPH particles are updated with an Euler step with the updated velocities.:

$$\mathbf{x}_{i_F}(t+\Delta t) = \mathbf{x}_{i_F}(t) + \Delta t \mathbf{v}_{i_F}(t+\Delta t) \qquad (5)$$

$$\mathbf{x}_{i_S}(t+\Delta t) = \mathbf{x}_{i_S}(t) + \Delta t \mathbf{v}_{i_S}(t+\Delta t). \qquad (6)$$

The algorithm is outlined in Alg. 1.

4.2. Implementation

Neighborhood search: We use a combination of z-index sort and compact hashing [IABT11] to accelerate the neighborhood search. All FLIP and SPH particles are stored in the same uniform grid with an edge length of $2h$ as the same cubic spline kernel with the same support of $2h$ is used in IISPH and for the velocity interpolations between FLIP and SPH particles. Prior to the pressure computation, FLIP and SPH neighbors are queried for each SPH particle. The ~ 240 FLIP neighbors are not stored as they are only used once in Eq. 1. The ~ 40 SPH neighbors are stored as they are required in the iterative IISPH solver. After the pressure projection, a second query finds the ~ 40 SPH neighbors for each FLIP particle. Again, these neighbors are not stored as they are only used for velocity interpolation, Eqs. 2 and 3.

Boundary handling: We represent solid boundaries with particles and use the concept of [AIA*12] to handle the one- and two-way coupling of SPH particles with boundary particles. The boundary handling for FLIP particles is realized by a spatially adaptive α in Eq. 4. If the neighboring SPH particle of a FLIP particle is in the influence radius of a boundary particle, we set $\alpha = 1$. This technique alleviates the clustering of FLIP particles at boundaries. It is in contrast to standard FLIP, where the path of FLIP particles is commonly clipped at boundaries.

Particle properties: SPH particles carry the standard properties that are required for IISPH. FLIP particles, however, require less properties. In particular, references to FLIP or SPH neighbors are not stored. Together with the matrix-free IISPH implementation, this results in a memory-efficient implementation, e.g. 15GB for 160 million particles (Fig. 1).

Isolated particles: If an SPH particle does not have FLIP neighbors, Eq. 1 cannot be computed. Instead, $\mathbf{v}^*_{i_S}$ is set to the velocity of the isolated SPH particle. If a FLIP particle does not have neighbors, gravity is applied to its velocity.

5. Results

We have implemented IISPH, standard FLIP and IISHP-FLIP to compare simulation results in Sec. 5.1 and performance in Sec. 5.2. IISPH and IISPH-FLIP employ the cubic spline kernel [Mon05]. In IISPH, we additionally model the viscous force as in [Mon05] and surface tension as in [BT07]. Note, that we do not need to apply additional viscosity in IISPH-FLIP. Surface tension is optional and not applied in the presented IISPH-FLIP simulations. Our implementation of the standard FLIP solver follows [ZB05] with eight FLIP particles per pressure sample and employs the variational approach for boundary handling as in [BBB07]. The pressure system is solved using MICCG [Bri08]. The images were rendered with Houdini's Mantra renderer [Sid13]. The accompanying videos are encoded with 50 frames per second.

In order to compare visual properties and the performance of the implemented solvers, we set up a breaking dam scenario as specified in [KFV*05].

5.1. Visual comparison

We first compare standard FLIP with IISPH in order to demonstrate the relation and peculiarities of the two solvers.

We believe that this helps to assess the benefits of the proposed IISPH-FLIP method. A matching frame of the breaking dam scenario simulated with all considered methods is shown in Figure 2.

IISPH vs FLIP

The pressure computation in IISPH is performed on the SPH particles, while FLIP computes the pressure on the auxiliary grid. Thus, the distance between pressure samples in IISPH corresponds to the SPH particle spacing h, while it corresponds to the distance of the grid nodes Δx_{grid} in FLIP. The detail in FLIP simulations is distinctly increased by the additional degrees of freedom of the employed FLIP particles. We highlight this effect by comparing IISPH and FLIP (i) for the same resolution of the pressure field and (ii) for the same number of FLIP particles.

The *same resolution of the pressure field* is obtained by setting the SPH particle spacing h to the grid spacing $\Delta x_{grid} = 1.85$cm in FLIP. Thereby, the number of particles differs: 96k SPH particles in IISPH and 772k FLIP particles in FLIP. Although pressure is computed at the same resolution, the low-resolution IISPH simulation has noticeably higher numerical diffusion which results in damping. This can be observed from the lower vorticity and the slower advection of the wave.

The number of SPH particles is increased to the *same number of FLIP particles* as in the FLIP simulation by halving the SPH particle spacing to $h = 0.925$cm. For this resolution, the level of numerical viscosity of the IISPH simulation is comparable to the FLIP simulation, as can be assessed by comparing the velocity field for different frames. It also seems that the waves are moving at the same speed.

Figure 2 also illustrates that FLIP suffers from compression. In contrast, IISPH preserves the fluids volume in both resolutions. The tolerated volume compression was set to 0.1% for IISPH.

IISPH-FLIP vs FLIP

The simulation result of IISPH-FLIP looks similar to FLIP for the same number of FLIP particles and the same resolution of the pressure field. The level of numerical diffusion in IISPH-FLIP is comparable to standard FLIP. While standard FLIP suffers from mass loss and hence volume compression, our IISPH-FLIP variant preserves the fluid volume with an unperceivable error of 0.1%.

IISPH-FLIP vs IISPH

Although only $\frac{1}{8}$ of the number of pressure samples are used in IISPH-FLIP in comparison to the high-resolution IISPH simulation, a similar visual detail and vorticity can be observed. In contrast, low-resolution IISPH suffers from higher numerical diffusion, less detail and the velocity field is smoothed. Thus, IISPH-FLIP leads to a preferred simulation result in comparison to the low-resolution IISPH simulation, while outperforming the high-resolution IISPH simulation with the same number of particles by a factor of 7.

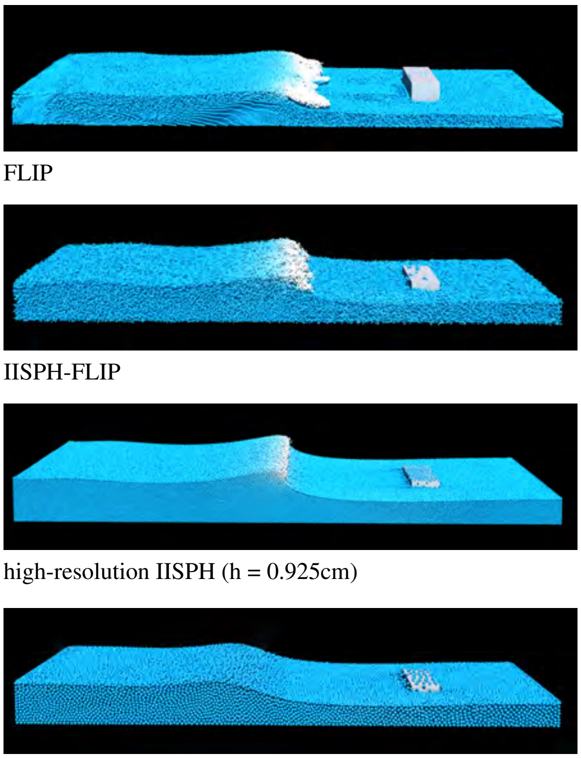

FLIP
IISPH-FLIP
high-resolution IISPH (h = 0.925cm)
low-resolution IISPH (h = 1.85cm)

Figure 2: *The obstacle allows for a visual comparison of mass loss. Distinct differences in the velocity field are perceivable for the considered methods. Particles are color coded with respect to velocity.*

Interpolation radii

IISPH-FLIP works with three different support radii: r for the IISPH kernel function, $r_{F\rightarrow S}$ for the velocity interpolation from FLIP particles to SPH particles and $r_{S\rightarrow F}$ for the velocity interpolation from SPH particles back to FLIP particles. In terms of the IISPH kernel support r, we follow [Mon92] and use $r = 2h$ in combination with the cubic spline function and particle spacing h.

Compared to the kernel support r, the support radii $r_{F\rightarrow S}$ and $r_{S\rightarrow F}$ for the velocity interpolations are less crucial for the stability, but influence the performance and the visual outcome. We experimented with various combinations of $r_{F\rightarrow S}$ and $r_{S\rightarrow F}$ using the breaking dam scene shown in Figure 3.

Solver	particles	time / frame (s)
IISPH (h = 0.925cm)	772k	21.4
IISPH (h = 1.85cm)	96k	1.31
FLIP ($\Delta x_{grid} = 1.85$cm)	772k	11.4
IISPH-FLIP (h = 1.85cm)	772k	3.1

Table 1: *Performance comparison of IISPH in two different resolutions, FLIP and IISPH-FLIP. The particle numbers refer to SPH particles for IISPH and to FLIP particles for FLIP and IISPH-FLIP.*

For $r_{F\to S} = 2h$, the velocity interpolation from FLIP particles to SPH particles takes 0.82s and reduces to 0.60s for $r_{F\to S} = h$. The same holds for the interpolation of the resulting velocities from IISPH particles back to FLIP particles, where the computation for $r_{S\to F} = 2h$ takes 0.97s and only 0.71s for $r_{S\to F} = h$. The overall computation time per frame ranges from 2.62s for $r_{F\to S} = r_{S\to F} = h$ up to 3.1s for $r_{F\to S} = r_{S\to F} = 2h$.

As illustrated in Figure 3, the visual appearance is less promising for smaller interpolation radii, since FLIP particles tend to stick to their closest accompanying SPH particle, leading to popcorn-like patterns of FLIP particles. Hence, we prefer to use $r_{F\to S} = r_{S\to F} = 2h$ for the velocity interpolations in order to obtain visually satisfying results.

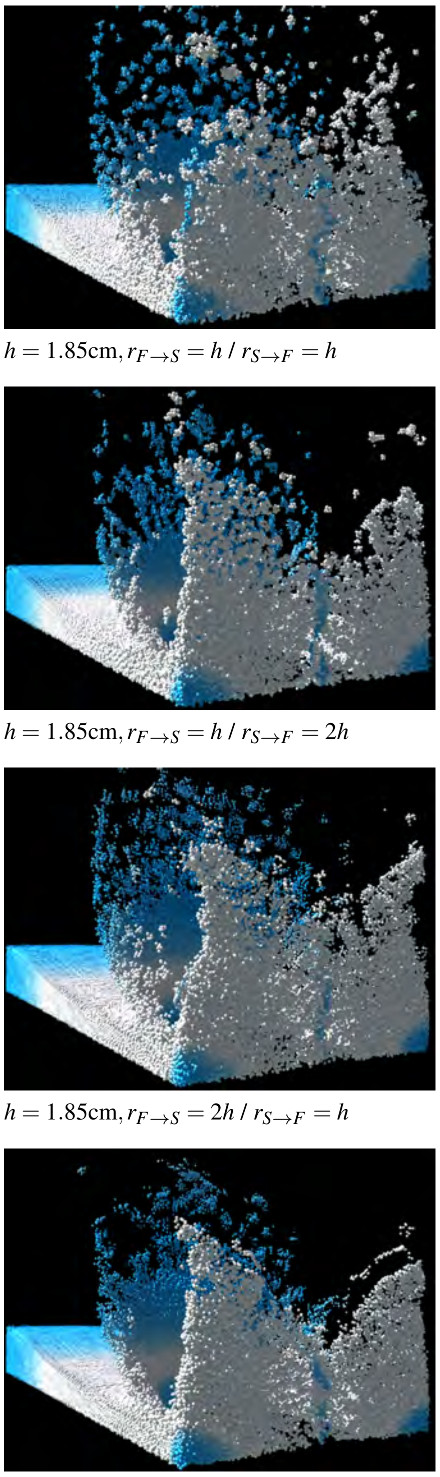
5.2. Performance comparison

We simulated the breaking dam scenario for all solvers on a 6-core Xeon X5680 3.33 GHz with 24 GBs of RAM using all threads. The performance measurements are summarized in Table 1.

All simulations are computed with an adaptive time step holding a CFL number of 1. Thus, the practical time step is restricted by the distance of the pressure-sampling points: h for IISPH and IISPH-FLIP, Δx_{grid} for FLIP. Thus, the same time step is used for IISPH-FLIP, FLIP and low-resolution IISPH with $\Delta x_{grid} = h = 1.85$cm, while it is halved for the high-resolution IISPH simulation with $h = 0.925$cm.

IISPH-FLIP adds computational overhead to an IISPH simulation with the same number of pressure samples (SPH particles) as quantities have to be interpolated from SPH particles to FLIP particles and back. For the given scene, it took 2.5 times longer to compute a frame with IISPH-FLIP than in the low-resolution IISPH simulation. IISPH-FLIP took 3.1s per frame. This timing composes of 0.24s neighborhood search, 0.82s for the interpolation of velocities from FLIP particles to SPH particles, 1.07s for the IISPH pressure projection and 0.97s for the interpolation of the resulting velocities back to FLIP particles. In contrast, the low-resolution IISPH took 1.31s per frame.

However, as discussed in Sec. 5.1, the additional proce-


$h = 1.85\text{cm}, r_{F\to S} = h$ / $r_{S\to F} = h$

$h = 1.85\text{cm}, r_{F\to S} = h$ / $r_{S\to F} = 2h$

$h = 1.85\text{cm}, r_{F\to S} = 2h$ / $r_{S\to F} = h$

$h = 1.85\text{cm}, r_{F\to S} = 2h$ / $r_{S\to F} = 2h$

Figure 3: *Different combinations of support radii for the velocity interpolations between FLIP and SPH particles.*

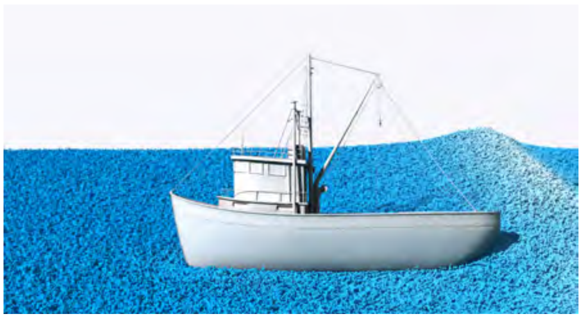


Figure 4: *A two-way coupled boat, plausibly floating on 40M FLIP particles. The particles are not penetrating the solid object.*

dures strikingly increase the visual detail and make the simulation even comparable to the high-resolution IISPH result. Compared to the IISPH simulation with the same number of particles, IISPH-FLIP significantly increases the performance as the complexity of the pressure field is reduced by factor 8 and the practical time step can be doubled. For the IISPH simulation with 772k SPH particles, it took 21.4s per frame which is almost 7 times slower than the 3.1s required by IISPH-FLIP.

IISPH-FLIP outperforms our standard FLIP implementation. This is due to the fact that the performance of FLIP solvers is influenced by numerous aspects which have not been optimized in our implementation. However, it should be noted that IISPH performs the pressure computation in only 9 iterations on average, while our standard FLIP implementation using MICCG took 110 iterations on average.

5.3. Large-scale scenario

One major limitation of IISPH is its memory consumption. Our approach overcomes this limit since FLIP particles do not need to store any properties related to the pressure projection. FLIP particle neighbors are not stored.

The scene in Figure 1 consists of 160 million FLIP particles. In our implementation of IISPH, the memory consumption of this scene would be 48GB, which exceeds the limits of current standard working machines. In contrast, our IISPH-FLIP implementation consumed only 15GB for this scene and took 10 minutes per frame on average to compute on a 16-core 3.46 GHz Intel i7. The simulation of the scene with the same number of SPH particles would not have been possible within an acceptable time.

5.4. Two-way coupling

As described in Section 4.2, there is no need to clamp the FLIP particles trace at solid boundaries. The boundary collision is implicitly handled and it is not necessary to query the neighborhood of FLIP particles for boundary particles and the computation of the hydrodynamic forces can be performed effectively without any additional neighborhood query. An example for a coupled solid is demonstrated in the scene shown in Figure 4 with 40M FLIP particles. Although the solid interaction of FLIP particles is not explicitly considered, no FLIP particles penetrate the solid boundary and the boat floats plausibly on the fluid particles.

6. Conclusion and future work

We have shown how to incorporate IISPH for the pressure projection and boundary handling into a standard FLIP simulator. Usually, mass conservation is an issue in FLIP simulations. Due to the mass preservation of SPH, our proposed IISPH-FLIP counteracts this issue and simulates incompressible fluids. On the other hand, SPH usually suffers from numerical diffusion, especially for low-resolution simulations. Adding FLIP particles helps to preserve vorticity and adds details to IISPH. While IISPH-FLIP shows a behavior and degree-of-detail that is comparable to an IISPH simulation with half the pressure-sampling distance, it outperforms such a simulation by a factor of 7.

We have shown, that the boundary handling for FLIP particles can be handled implicitly by spatially adapting the viscosity of the fluid. We do not need to clamp the particle movement at boundaries and can handle two-way coupling of solid objects by only relying on SPH particles.

Since FLIP particles do not need to store properties that are required for the pressure projection and the neighborhood can be computed online instead of the memory consuming storage of neighbors and distances, our IISPH-FLIP simulation has a low memory footprint. We have shown a scene with 160 million FLIP particles computed on a single desktop PC, consuming only 15GB of memory.

Since we employ both, FLIP and SPH, additional techniques, e.g., vorticity confinement for SPH or particle reseeding for FLIP particles could be investigated.

References

[ACAT13] Akinci N., Cornelis J., Akinci G., Teschner M.: Coupling elastic solids with smoothed particle hydrodynamics fluids. *Computer Animation and Virtual Worlds 24* (2013), 195–203. 3

[AIA*12] Akinci N., Ihmsen M., Akinci G., Solenthaler B., Teschner M.: Versatile rigid-fluid coupling for incompressible SPH. *ACM Transactions on Graphics (Proceedings of the ACM SIGGRAPH) 31*, 4 (2012), 62:1–62:8. 3, 4

[APKG07] Adams B., Pauly M., Keiser R., Guibas L.: Adaptively sampled particle fluids. In *ACM Transactions on Graphics (Proceedings SIGGRAPH)* (2007), vol. 26, pp. 48:1–48:7. 2

[ATT12] Ando R., Thuerey N., Tsuruno R.: Preserving fluid sheets with adaptively sampled anisotropic particles. *IEEE Transactions on Visualization and Computer Graphics 18*, 8 (2012), 1202–1214. 1, 2, 3

[ATW13] Ando R., Thuerey N., Wojtan C.: Highly adaptive liquid simulations on tetrahedral meshes. *ACM Transactions on Graphics (Proceedings of the ACM SIGGRAPH) (in press)* (2013). 1, 2

[BB12] Boyd L., Bridson R.: MultiFLIP for energetic two-phase fluid simulation. *ACM Transactions on Graphics (Proceedings of the ACM SIGGRAPH) 31*, 2 (2012), 16:1–16:12. 1

[BBB07] Batty C., Bertails F., Bridson R.: A fast variational framework for accurate solid-fluid coupling. *ACM Trans. Graph. 26*, 3 (2007). 1, 2, 3, 4

[BKR88] Brackbill J., Kothe D., Ruppel H.: FLIP: A low-dissipation, particle-in-cell method for fluid flow. *Computer Physics Communications 48* (1988), 25–38. 1, 3

[BLS12] Bodin K., Lacoursière C., Servin M.: Constraint fluids. *IEEE Transactions on Visualization and Computer Graphics 18*, 3 (2012), 516–526. 2

[BR86] Brackbill J., Ruppel H.: FLIP: A method for adaptively zoned, particle-in-cell calculations for fluid flows in two dimensions. *Journal of Computational Physics 65* (1986), 314–343. 1, 2, 3

[Bri08] Bridson R.: *Fluid Simulation for Computer Graphics*. A K Peters/CRC Press, 2008. 1, 3, 4

[BT07] Becker M., Teschner M.: Weakly compressible SPH for free surface flows. *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2007), 209–217. 2, 4

[Cho68] Chorin A. J.: Numerical solution of the Navier-Stokes equations. *Mathematics of Computation 22*, 104 (1968), 745–762. 3

[CM12] Chentanez N., Müller M.: Mass-conserving Eulerian liquid simulation. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2012), SCA '12, pp. 245–254. 2

[DC96] Desbrun M., Cani M.-P.: Smoothed particles: A new paradigm for animating highly deformable bodies. In *Eurographics Workshop on Computer Animation and Simulation (EGCAS)* (1996), Boulic R., Hegron G., (Eds.), Springer-Verlag, pp. 61–76. Published under the name Marie-Paule Gascuel. 2

[GB13] Gerszewski D., Bargteil A. W.: Physics-based animation of large-scale splashing liquids. *ACM Trans. on Graphics (Proceedings of the ACM SIGGRAPH Asia) 32*, 6 (2013), 1–6. 2, 3

[GLHB09] Gao Y., Li C.-F., Hu S.-M., Barsky B. A.: Simulating gaseous fluids with low and high speeds. *Computer Graphics Forum 28*, 28 (2009), 1845–1852. 2

[HLWW12] He X., Liu N., Wang H., Wang G.: Local Poisson SPH for viscous incompressible fluids. *Computer Graphics Forum 31* (2012), 1948–1958. 2

[IABT11] Ihmsen M., Akinci N., Becker M., Teschner M.: A parallel SPH implementation on multi-core CPUs. *Computer Graphics Forum 30*, 1 (2011), 99–112. 4

[IAGT10] Ihmsen M., Akinci N., Gissler M., Teschner M.: Boundary handling and adaptive time-stepping for PCISPH. In *Proceedings VRIPHYS* (2010), pp. 79–88. 3

[ICS*13] Ihmsen M., Cornelis J., Solenthaler B., Horvath C., Teschner M.: Implicit incompressible SPH. *IEEE Transactions on Visualization and Computer Graphics 19* (2013). 2, 3

[IOS*14] Ihmsen M., Orthmann J., Solenthaler B., Kolb A., Teschner M.: SPH fluids in computer graphics. *State-of-the-Art Report Eurographics* (2014). 2

[KFV*05] Kleefsman K. M. T., Fekken G., Veldman A. E. P., Iwanowski B., Buchner B.: A volume-of-fluid based simulation method for wave impact problems. *Journal of Computational Physics 206*, 1 (2005), 363–393. 4

[LHK09] Lee H.-Y., Hong J.-M., Kim C.-H.: Interchangeable SPH and level set method in multiphase fluids. *The Visual Computer 25*, 5-7 (2009), 713–718. 2

[LTKF08] Losasso F., Talton J., Kwatra N., Fedkiw R.: Two-way coupled SPH and particle level set fluid simulation. *IEEE Transactions on Visualization and Computer Graphics 14*, 4 (2008), 797–804. 2

[LZF10] Lentine M., Zheng W., Fedkiw R.: A novel algorithm for incompressible flow using only a coarse grid projection. *ACM Trans. Graph. 29*, 4 (2010), 114:1–114:9. 2

[MCG03] Müller M., Charypar D., Gross M.: Particle-based fluid simulation for interactive applications. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2003), pp. 154–159. 2

[Mon92] Monaghan J.: Smoothed particle hydrodynamics. *Ann. Rev. Astron. Astrophys. 30* (1992), 543–574. 2, 3, 5

[Mon05] Monaghan J.: Smoothed particle hydrodynamics. *Reports on Progress in Physics 68*, 8 (2005), 1703–1759. 2, 4

[RWT11] Raveendran K., Wojtan C., Turk G.: Hybrid smoothed particle hydrodynamics. In *Proceedings of the 2011 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2011), SCA '11, ACM, pp. 33–42. 2

[SG11] Solenthaler B., Gross M.: Two-scale particle simulation. *ACM Transactions on Graphics (Proceedings SIGGRAPH) 30*, 4 (2011), 81:1–81:8. 2

[Sid13] Side Effects Software: Houdini [software]. *http://www.sidefx.com/* (2013). 4

[SP09] Solenthaler B., Pajarola R.: Predictive-corrective incompressible SPH. *ACM Trans. on Graphics (Proceedings of the ACM SIGGRAPH) 28* (2009), 40:1–40:6. 2

[ZB05] Zhu Y., Bridson R.: Animating sand as a fluid. *ACM Transactions on Graphics (Proceedings of the ACM SIGGRAPH) 24*, 3 (2005), 965–972. 1, 2, 3, 4

[ZLC*13] Zhu B., Lu W., Cong M., Kim B., Fedkiw R.: A new grid structure for domain extension. *ACM Trans. Graph. 32*, 4 (2013), 63:1–63:12. 2

[ZYF10] Zhu B., Yang X., Fan Y.: Creating and preserving vortical details in SPH fluid. *Computer Graphics Forum 29*, 7 (2010), 2207–2214. 2