EUROGRAPHICS 2015 / O. Sorkine-Hornung and M. Wimmer
(Guest Editors)

Volume 34 (2015), Number 2

A Dimension-reduced Pressure Solver for Liquid Simulations

Ryoichi Ando[1], Nils Thürey[2], and Chris Wojtan[1]

[1]IST Austria, [2]Technische Universität München

Figure 1: *Our method can efficiently compute a coarse pressure solve for high-resolution liquid simulations while taking into account free-surface boundary conditions. Here, three images of a liquid simulation are shown. The pressure solve uses a resolution ($33 \times 25 \times 33$) which is 16^3 times smaller than the resolution of the surface level-set ($513 \times 385 \times 513$). The coarse pressure samples for a line along x are illustrated in yellow on the left.*

Abstract

This work presents a method for efficiently simplifying the pressure projection step in a liquid simulation. We first devise a straightforward dimension reduction technique that dramatically reduces the cost of solving the pressure projection. Next, we introduce a novel change of basis that satisfies free-surface boundary conditions exactly*, regardless of the accuracy of the pressure solve. When combined, these ideas greatly reduce the computational complexity of the pressure solve without compromising free surface boundary conditions at the highest level of detail. Our techniques are easy to parallelize, and they effectively eliminate the computational bottleneck for large liquid simulations.*

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

1. Introduction

In practical simulations of liquids at large resolutions, the pressure projection quickly becomes the computational bottleneck [LZF10]. Previous researchers have increased the speed of this pressure solve (or removed it altogether) with dimension-reduction techniques and clever changes of basis [TLP06, DWLF12]. Unfortunately, these previous techniques do not apply to liquids with a moving free surface. The difficulty occurs because liquids exhibit unpredictable changes in domain shape and topology, which leads to continuously changing Dirichlet boundary conditions. These distinctions tend to violate the assumptions of existing dimension reduction approaches, e.g. a fixed domain topology or the presence of Neumann boundary conditions only. Additionally, model reduced simulations are by construction constrained by a precomputed basis.

In this paper, we present a method for simulating high resolution liquids with a coarse grid pressure solver, while avoiding many of the problems that hinder previous approaches. Our solution is a two step process that first down-

samples the full problem into a space with fewer degrees of freedom, while taking into account all boundary conditions of the original problem. Second, when correcting the velocities with the pressure gradient from the reduced solve, we ensure free surface boundary conditions are still enforced for the original discretization. Our contributions are as follows:

- We devise a straightforward dimension-reduction technique that significantly reduces the cost of solving the pressure projection. The approach is simple and flexible, and it allows adaptive refinement of the pressure basis if desired.
- We introduce a novel change of basis that *exactly* satisfies high-resolution Dirichlet conditions at the free surface, regardless of the resolution and accuracy of the pressure solve.

When combined, these ideas dramatically reduce the computational complexity of the pressure solve without compromising free surface boundary conditions at the highest level of detail. Our techniques are easy to parallelize, and they effectively eliminate the computational bottleneck for large liquid simulations.

2. Related Work

Eulerian solvers based on finite difference discretizations are commonly used in the graphics community, and a good overview can be found in the book by Bridson [Bri08]. Our work focuses on simulations that solve a linear system for calculating the pressure. While we use a regular Eulerian grid with level-set surface tracking [OF03], our approach could be extended to mixed Eulerian-Lagrangian solvers such as *FLIP* [ZB05] or even recently proposed SPH solvers [CIPT14] (given a suitable method to generate a coarse discretization based on the point samples).

Lentine et al. [LZF10] describe a pressure projection that shares our goal of using a coarse grid. However, the method was not designed specifically for liquid simulation, and thus it requires an additional high-resolution pressure solve around the surface to enforce free surface boundary conditions. Several works in recent years have made use of multi-grid methods [MCPN08, MST10, CM11, JKNH13].

Multi-grid can theoretically achieve great efficiency solving Poisson problems, but in practice its convergence strongly hinges on an accurate discretization of the problem on the coarser grids in the hierarchy. Ferstl et al. [FWD13] showed that regular grid multigrid schemes do not guarantee convergence in general without significantly more complicated data-structures. In contrast, our method does not share these problems, because our dimension-reduction strategy is guaranteed to converge and our surface-aware basis satisfies Dirichlet boundary conditions exactly. Additionally, a variety of approaches reduce the degrees of freedom for fluid simulations with model reduction. The approach of Treuille et al. [TLP06], was extended by modular bases, [WST09] deforming domains [SSW*13], and more accurate advection [KD13]. Also, procedural bases have been proposed [DWLF12], or were merged with basis functions from simulations [GKSB13].

The addition of high-resolution detail to low-resolution simulations is common practice when animating smoke [KTJG08, NSCL08] as well as liquids [TWGT10, YWTY12, KTT13]. Alternatively, vortex sheet models can simulate small scale detail in combination with coarse pressure solves [KSK09, BHW13].

Edwards and Bridson [EB14] use higher-order polynomial bases for liquid simulation; while we use a simple and fast tri-linear interpolation for our method, higher order interpolations could also be integrated.

Nielsen and Bridson [NB11] use coarse simulations to guide high-resolution fluid motion. While this work is largely orthogonal to ours, such guiding approaches share similar goals and would complement our algorithm well in a practical setting.

3. A dimension-reduced pressure solver

We perform liquid simulation by discretizing the Euler equations

$$\frac{\partial \boldsymbol{u}}{\partial t} + \boldsymbol{u}\cdot\nabla\boldsymbol{u} = -\frac{1}{\rho}\nabla p + \boldsymbol{f}, \qquad \nabla\cdot\boldsymbol{u} = 0 \tag{1}$$

where $\boldsymbol{u}$ is the liquid velocity, t is time, p is pressure, ρ is density, and $\boldsymbol{f}$ is body acceleration. These equations are subject to the boundary conditions $p = 0$ at the free surface and $\boldsymbol{u}\cdot\boldsymbol{n} = \boldsymbol{v}\cdot\boldsymbol{n}$ at solid obstacles, where $\boldsymbol{n}$ is the surface normal of the obstacle, and $\boldsymbol{v}$ is the obstacle's velocity. During each simulation time step, we find the pressure field that minimizes kinetic energy in the least squares sense [BBB07]

$$\underset{p}{\operatorname{argmin}} \int_{\text{fluid}} \frac{\rho}{2} \left\| \breve{\boldsymbol{u}} - \frac{\Delta t}{\rho}\nabla p \right\|^2 dV \tag{2}$$

which results in a Poisson equation for the pressure field

$$\frac{\Delta t}{\rho}[\nabla^2]p = [\nabla]^T \breve{\boldsymbol{u}}\,. \tag{3}$$

Here $\breve{\boldsymbol{u}}$ is the intermediate velocity after the advection, Δt is the time step size, $[\nabla]$ is the discretized gradient operator, and $[\nabla^2] \equiv [\nabla]^T[\nabla]$ is the discretized Laplacian operator. We use the level set method to track the liquid free surface [OF03], and we enforce the Dirichlet boundary conditions with the ghost fluid method [GFCK02].

To reduce the degrees of freedom in the pressure projection, we introduce the change of variables

$$p = U\tilde{p} \tag{4}$$

where $\tilde{p}$ is a pressure field sampled on a coarse grid, and U is a sparse *up-sampling* matrix that interpolates $\tilde{p}$ onto the high resolution grid. We use linear interpolation in our

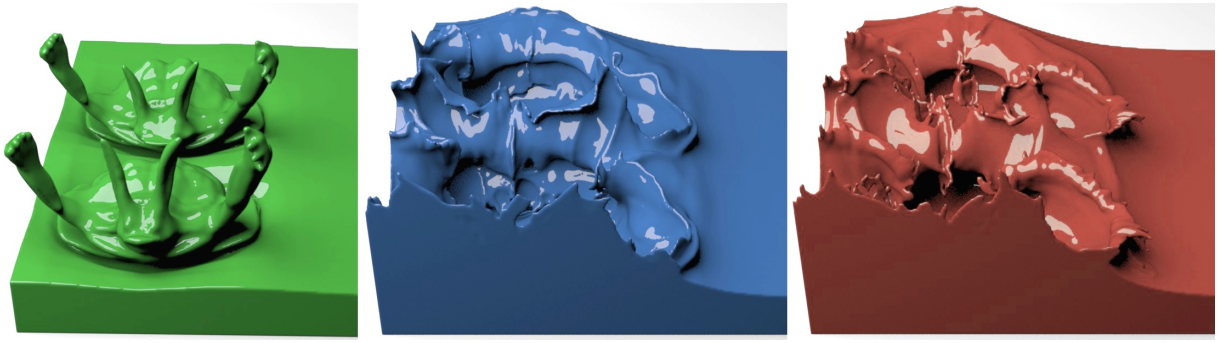

Figure 2: *From left to right: a naively down-sampled coarse pressure solver; our new dimension-reduced pressure solver coupled with higher resolution levelset; a full resolution simulation. A resolution of* $257 \times 205 \times 257$ *was used for velocity and levelset in all cases, while the coarsened versions use a* $17 \times 13 \times 17$ *grid for pressure (reduced by a factor of* 16^3*).*

examples, though higher-order interpolation is also possible at the expense of a denser matrix with poorer conditioning.

Directly applying this change of variables to Eq. (3) results in an overdetermined linear system. We instead substitute Eq. (4) into Eq. (2) and solve for $\tilde{p}$, yielding the least squares system

$$U^T \frac{\Delta t}{\rho} [\nabla^2] U \tilde{p} = U^T [\nabla]^T \check{\boldsymbol{u}}. \tag{5}$$

The left-hand side represents a square, symmetric positive definite matrix that can be solved with a standard preconditioned conjugate gradient solver. Once we have a solution $\tilde{p}$, we compute the gradient $[\nabla] p = [\nabla] U \tilde{p}$ for use in Eq. (1). The boundary conditions embedded into $[\nabla^2]$ ensure that velocities have reasonable behavior near the free surface and solid obstacles.

The construction of the system matrix $U^T \frac{\Delta t}{\rho} [\nabla^2] U$ is trivially parallelizable, so the only potentially serial part of this algorithm is the linear system solve itself. We use an MIC(0) preconditioned conjugate gradient method [Bri08][†], and the solution time is negligible in practice. With such small dimensions, the conjugate gradient algorithm takes on the order of 100 milliseconds even in our most complex examples.

Note that the linear system in Eq. (5) has a significantly lower dimension than Eq. (3). This derivation is inspired by model reduction techniques [TLP06], and shares the goal of reducing dimensionality. However, a fundamental difference is that we employ the reduction only for the pressure solve, which allows us to use off-the-shelf algorithms as simulation components for the other steps of the simulation.

Our approach is also different from a naive down-sampling method, which would generate a low-resolution velocity with $\mathbf{u}_L = U^T \check{\boldsymbol{u}}$, solve Eq. (3) on a coarse grid, and then up-sample a corrected velocity with $U \mathbf{u}_L$. Such an approach fares poorly, as can be seen on the left side of Figure 2. Instead, our reduced system in Eq. (5) is a denser matrix that consists of interpolated versions of all the original equations. It computes a solution that simultaneously accommodates the high-resolution boundary conditions as accurately as possible in the least squares sense.

Eq. (5) also can be regarded as a *Galerkin-based coarsening* scheme that is used for certain classes of multigrid methods [FWD13]. In this context our approach represents a modified 2-level scheme with a zero initial guess, and a boundary-aware prolongation/interpolation operator, described in the next section. We are unaware of any other work that uses Galerkin-based coarsening as an approximation for the actual solution in this way, especially in computer graphics.

4. A surface-aware pressure basis

When solving the original problem of Eq. (3), the degrees of freedom in p near the liquid surface approximately match up with those of the surface tracker; the detailed surface geometry imposes detailed Dirichlet conditions, which are satisfied by the detailed pressure field p. When solving Eq. (5), however, $\tilde{p}$ has far fewer degrees of freedom than the detailed liquid surface. While $\tilde{p}$ still satisfies all free-surface boundary conditions optimally (in the least-squares sense), these mismatched degrees of freedom can slightly violate Dirichlet conditions near complex liquid surface geometry.

We introduce a novel surface-aware pressure basis to fix this constraint violation. By representing the pressure in a basis that satisfies high-resolution Dirichlet boundary conditions by default, we make it impossible for the reduced pressure field to violate free-surface boundary conditions.

In the absence of surface tension, $p = 0$ at the liquid surface. We can force our reduced pressure to meet this condition by directly leveraging our high-resolution surface geometry. Our first approach was to encode the pressure as a

[†] With MIC(0) parameters $\tau = 0.97$ and $\sigma = 1.0$, which we found to give better convergence than the default values.

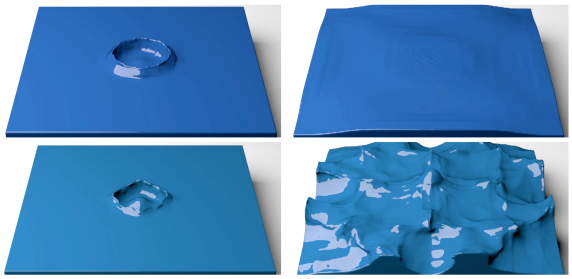

Figure 3: *Waterdrop at* $257 \times 129 \times 257$ *with our pressure solver* $17 \times 9 \times 17$ *(a factor of* 16^3*). The top row uses Eq. (9), while the bottom row uses Eq. (5). Our surface-aware basis allows the simulation to form a rounded crown and settle down, while a surface-oblivious basis produces more grid aligned artifacts and unphysically increases in energy.*

scalar multiple of the signed distance function:

$$p = \Phi U \tilde{p} \tag{6}$$

where ϕ is a diagonal matrix representing the signed distance function value ϕ at the location of each high-resolution pressure sample. This way, the pressure can only equal zero at the liquid surface, but it can have arbitrary values elsewhere. Unfortunately, this approach fails to produce smooth pressure functions, because the signed distance function ϕ is not differentiable at the medial axis. We fix this problem by replacing Φ with a matrix that smoothly transitions between ϕ at the surface and an identity operation within the liquid:

$$p = \hat{\Phi} U \tilde{p} \tag{7}$$

where the diagonal of $\hat{\Phi}$ is given by the modified distance function

$$\hat{\phi} = \sin\left(\frac{\pi}{2}\min\left(1, \frac{-\phi}{2\Delta x}\right)\right) \tag{8}$$

with the grid spacing Δx. Note that $\hat{\phi}$ has the desirable properties that it ranges between 0 and 1 monotonically within the narrow-band of a liquid levelset, it is differentiable within the domain, and it vanishes at the liquid surface. Noting that $\hat{\Phi} = \hat{\Phi}^T$, the Poisson problem is modified accordingly:

$$U^T \hat{\Phi} \frac{\Delta t}{\rho} [\nabla^2] \hat{\Phi} U \tilde{p} = U^T \hat{\Phi} [\nabla]^T \breve{\boldsymbol{u}}. \tag{9}$$

This form is used for all of our tests without surface tension, and is summarized in Figure 5. The importance of our surface-aware basis is highlighted by the comparison in Figure 3.

4.1. Surface tension

For liquids with nonzero surface tension, the Dirichlet condition changes to $p = \sigma H$, where σ is surface tension, and $H = \nabla^2 \phi$ is mean curvature. We modify our pressure basis

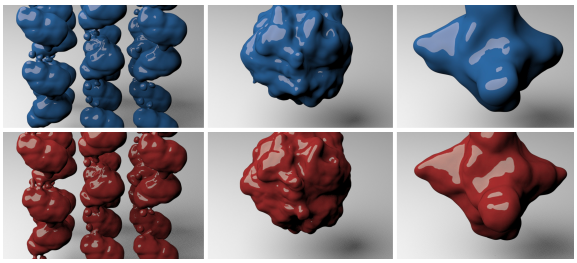

Figure 4: *A surface tension simulation with a base resolution of* 257^3*. The top row shows our pressure solver with a resolution of* 17^3*, while the bottom row shows a full resolution simulation.*

by adding a surface tension term:

$$p = \hat{\Phi} U \tilde{p} + (I - \hat{\Phi}) \sigma H \tag{10}$$

which equals σH at the free surface and transitions to our standard dimension-reduced pressure within the liquid. Substituting this pressure into Eq. (2) gives us a new minimization problem

$$\underset{\tilde{p}}{\operatorname{argmin}} \int_{\text{fluid}} \frac{\rho}{2} \left\| \left(\breve{\boldsymbol{u}} - \frac{\Delta t}{\rho} [\nabla](I - \hat{\Phi}) \sigma H \right) - \frac{\Delta t}{\rho} [\nabla] \hat{\Phi} U \tilde{p} \right\|^2 dV \tag{11}$$

which can be solved in exactly the same way as Eq. (9) by setting $\breve{\boldsymbol{u}} \leftarrow \breve{\boldsymbol{u}} - \frac{\Delta t}{\rho}[\nabla](I - \hat{\Phi})\sigma H$ and solving as normal. One way to interpret this operation is that the surface tension acts on the fluid velocity as a body acceleration integrated over a time step Δt. Eq. (9) then finds the unique pressure $\tilde{p}$ which minimizes the kinetic energy of the original velocity combined with surface tension. Additional discussion and implementation details are available as supplementary material.

These simple modifications guarantee that the pressure satisfies Dirichlet boundary conditions at the highest resolution, even with extremely few degrees of freedom. Figure 4 shows that this basis allows a reduced pressure to exhibit similar motion to a full resolution simulation.

4.2. Removing high-frequency divergence

Our dimension reduction approach finds the $\tilde{p}$ which satisfies the divergence-free constraint optimally in a least squares sense. Unfortunately, the resulting velocity field is not guaranteed to be divergence-free at the fine resolution,

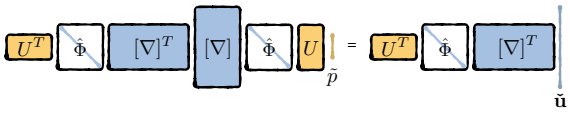

Figure 5: *A graphical summary of how we construct the linear system for our dimension reduced pressure solve. Colors indicate the matrix and vector sizes: blue denotes the dimension of the original system, yellow the reduced degrees of freedom.*

because $\tilde{p}$ does not have enough degrees of freedom to satisfy all divergence constraints exactly. As a result, we found that the high-resolution pressure field exhibits high-frequency divergences.

In our experience, a post-process of only three Jacobi iterations of the high-resolution Poisson system Eq. (3) are sufficient to remove these high-frequency artifacts and produce a pressure field that is qualitatively indistinguishable from a full high-resolution solution.

5. Adaptive pressure reduction

Section 4 showed how we can enforce free surface boundary conditions at the highest resolution using a novel surface-aware basis. Our dimension reduction approach also handles solid boundary conditions, but we have not yet found a pressure basis that enforces solid boundaries exactly at the high resolution, and we view this problem as future work.

In the mean time, if the exact enforcement of high-resolution Neumann boundary conditions is still desired, we can simply adaptively add pressure degrees of freedom to $\tilde{p}$ near boundaries. The corresponding modification of the up-sampling matrix U is straightforward: we simply add identity matrix blocks for the high-resolution $\tilde{p}$ samples. Figure 6 shows how this technique accurately computes pressures near thin obstacles. The same technique can also be used to add detail near the free surface or other interesting flow areas, but we found this unnecessary in our examples.

This adaptive approach adds more degrees of freedom, so the resulting problem increases in size. We found that adding samples near boundaries added so few additional degrees of freedom that the overhead was negligible. We did not observe any visual artifacts due to aggressive grading between coarse and fine pressure samples, unlike most h-adaptive simulation approaches.

If one does not adaptively add these pressure degrees of freedom along solid obstacles, then the Neumann boundary conditions will only be approximately enforced. Specifically, the zero-flux condition will not be guaranteed at every point along a high-resolution solid boundary, leading to potential changes in overall volume, as seen in Figure 9.

6. Algorithm overview

The fluid simulation method is summarized in Algorithm 1. Within a simulation time step, the velocity is first advected as normal. We then construct our reduced system matrix from an up-sampling matrix U, a modified distance matrix $\hat{\Phi}$, and the full resolution system matrix with boundary conditions embedded. We then solve the reduced system and compute the new pressure. As a post-process, we perform a few Jacobi iterations on the high-resolution pressure field. Finally, we update the velocity and advect the surface tracker. To reduce volume loss over time we also employ the method of Kim et al. [KLL*07].

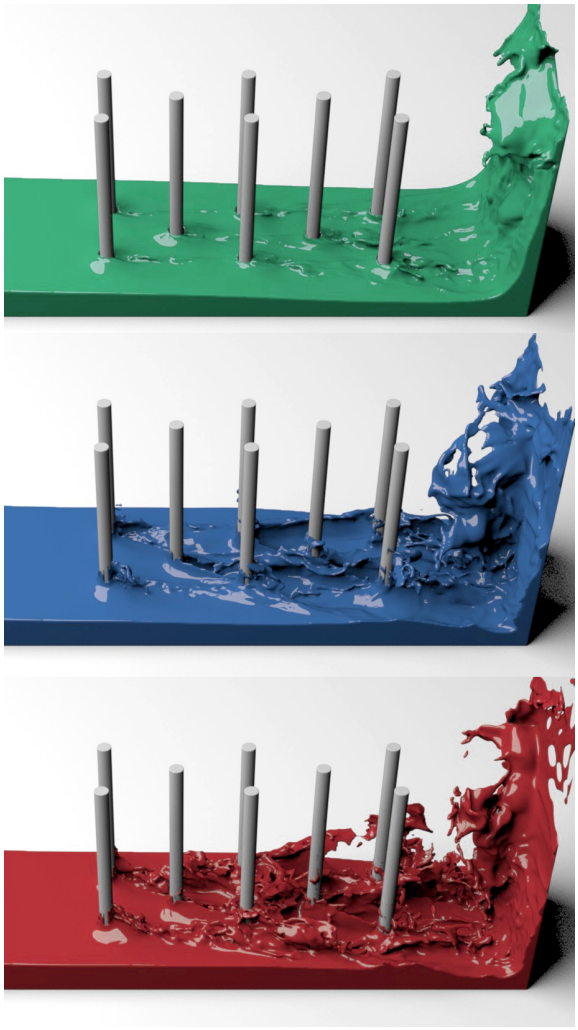

Figure 6: *A* $513 \times 257 \times 129$ *simulation with thin obstacles. Our method without adaptivity (top), our method with adaptivity (middle). A resolution of* $33 \times 17 \times 9$ *pressure grid is used as a coarse grid, and a band of* 6 *fine cells around each cylinder is used for addtional adaptivity. The result is visually similar to a full resolution simulation (bottom).*

7. Results and Discussion

Figure 2 shows a comparison of a simple down-sampling approach (as described in Section 3) with our method and a full simulation. Here our pressure solve uses a resolution that was 16 times lower than the initial resolution in each dimension, resulting in 4096 times fewer degrees of freedom. The naive approach fails to capture any of the high-frequency dynamics expected from a 257^3 resolution.

Figure 1 shows high-resolution ($513 \times 385 \times 513$) liquid splashing through complex obstacles, with a 16^3 times coarser pressure solve ($33 \times 25 \times 33$). The simulation was

Algorithm 1: One simulation time step

input : High resolution grid $\mathcal{G}$, velocity field $\boldsymbol{u}$,
signed distance function ϕ,
vector of mean curvatures $H = \nabla^2\phi$

1 Advect $\boldsymbol{u}$ to get intermediate velocity $\breve{\boldsymbol{u}}$;
2 Construct modified distance matrix $\hat{\Phi}$ according to Eq. (8);
3 Apply surface tension force $\breve{\boldsymbol{u}} \leftarrow \breve{\boldsymbol{u}} - \frac{\Delta t}{\rho}[\nabla](I-\hat{\Phi})\sigma H$;
4 Choose a coarse subset of (adaptive) sample locations $\mathcal{C}$;
5 Construct full-resolution matrices $[\nabla]$ and $\frac{\Delta t}{\rho}[\nabla^2]$, embedding boundary conditions as normal;
6 Construct up-sampling matrix U which interpolates $\mathcal{C}$ to get $\mathcal{G}$;
7 Construct system matrix $U^T\hat{\Phi}M\hat{\Phi}U$ in parallel;
8 Construct right hand side $U^T\hat{\Phi}[\nabla]^T\breve{\boldsymbol{u}}$;
9 Solve the reduced system (9) for $\tilde{p}$ using your method of choice;
10 Compute high-resolution pressure $p = U\tilde{p}$;
11 Perform 3 Jacobi iterations of (3) to improve p;
12 Update velocity $\boldsymbol{u} = \breve{\boldsymbol{u}} - \frac{\Delta t}{\rho}[\nabla]p$;
13 Extrapolate velocity;
14 Advect liquid surface and recompute signed distance ϕ;

run on a workstation with an Intel Core i7-3960X (6 core) CPU with 3.30GHz.

The average time for assembling the dimension-reduced matrix was 12.16 seconds, while solving the pressure system took 110 milliseconds on average. The high-resolution solve, on the other hand, required 4.54 minutes on average to calculate the pressure. Adding the additional solving steps (divergence computation etc.) for our algorithm, the full pressure solve required 25.36 seconds on average, and thus was ca. 11 times faster than a high-resolution pressure solve. For the whole simulation this lead to a speed-up factor of 4.3.

We also compared our method to an algebraic multigrid preconditioned conjugate gradient method in Figure 7. While the method is much faster than a MIC(0) preconditioned conjugate gradient method, it is still about four times slower than our approach. Figure 7 also illustrates the performance of our method at different coarsening resolutions.

Simulations with surface tension forces and obstacle interactions can be seen in Figure 4, Figure 6, and Figure 1, respectively. In each case, our method is able to generate fine wave motions and splashes that are much finer than the sparsely placed pressure samples.

7.1. Discussion

One of the key strengths of our method is its compatibility with existing fluid solvers based on regular grids. Our algorithm does not require any data structures that are more

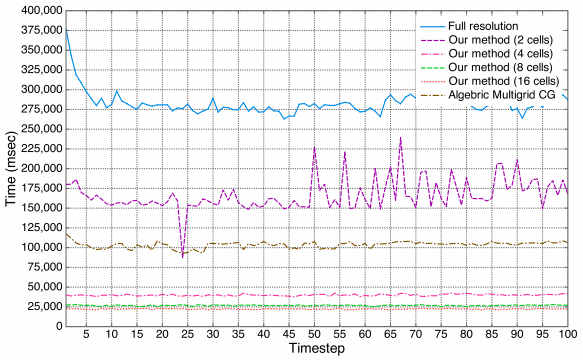

Figure 7: *Projection timings for Figure 1 at full resolution and various reduced resolutions. The timings include steps 2 to 12 from Algorithm 1 for our method, and matrix setup and a MIC(0) conjugate gradient solve for the full resolution.*

complicated than a regular grid. Simply replacing a standard pressure solve with the steps 3 to 12 of Algorithm 1 can yield significant speed-ups for large resolutions. Additionally, our method represents a simple yet powerful way to realize adaptivity. Important regions with high-resolution can be solved in a fully coupled fashion with regions using very coarse pressure samples. In such cases the regular data-structures lead to a substantially simpler implementation that when resorting to algorithms that require trees or more complex meshes.

We also found that the Jacobi smoothing is important to merge the low-resolution pressure and high-resolution surface basis into a desirable solution. Without this "glue" component, high-frequency errors can accumulate over time to give undesirable dynamics. However, when Jacobi smoothing is activated, we have found that three iterations are enough to give excellent results for a wide variety of simulation setups and resolutions. Interestingly, only performing Jacobi iterations without the surface-aware pressure basis of Section 4 has little effect, so the combination of both is crucial to achieve high quality results.

The algorithm presented in section 3 of our paper can be interpreted as a two-level Galerkin coarsening multigrid scheme, so we would like to highlight some benefits that our "special case" algorithm has over general multigrid methods. Multigrid techniques are typically associated with a full hierarchy of levels that must remain topologically consistent for convergence. The notion of a conditional convergence is not directly applicable to our scheme, because the coarse solve is a symmetric positive definite system, and the subsequent transfer of information to the full resolution grid is not an iterative process. High resolution boundary conditions are embedded in the coarse solve by construction, so errors due to topological differences between coarse and fine levels do not lead to noticeable visual artifacts in our examples.

Our surface-aware basis is novel; it further minimizes dif-

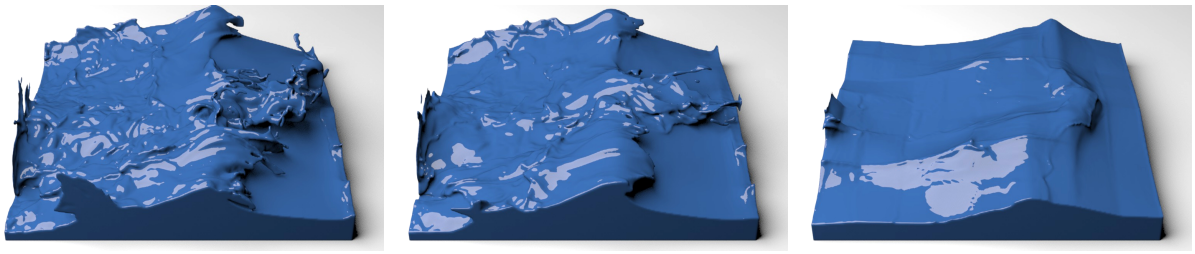

Figure 8: *Changing pressure solver accuracy from left to right:* $129 \times 103 \times 129$ *(factor of* 2^3*),* $33 \times 26 \times 33$ *(factor of* 8^3*) and* $9 \times 7 \times 9$ *(factor of* 32^3*) resolutions. A resolution of* $257 \times 205 \times 257$ *was used for the level set and velocity.*

ferences near the free surface, which is a crucial ingredient for a large coarsening factor between levels. Otherwise we would require far more Jacobi iterations or could only use our coarse solve as a preconditioner for a more complete full-resolution solve. Our supplemental video shows an example of extreme coarsening between levels (up to 32^3), a strategy which proves counterproductive in a traditional multigrid scheme that relies on small differences between consecutive levels. While there are similarities, our proposed approach is not an obvious extension of existing multigrid literature, and it is far simpler to implement than general multigrid methods.

7.2. Limitations

We stress that our method is not designed to solve the Poisson equation exactly at the high resolution, and as such our method can exhibit artifacts that scale with the down-sampling factor. Figure 8 shows how these artifacts develop for more and more extreme coarsening factors. Very large factors tend to lead to an increasing settling speed of the liquid, and fewer small scale dynamics. However, the surface-aware pressure basis allows small scale detail to form despite the sparse pressure samples. Extreme coarsening can also show seams resulting from linear interpolation during up-sampling. This problem can be alleviated in a variety of ways, e.g., by less aggressively coarsening, adding fine degrees of freedom (as described in Section 5), or by investing more time for a narrow-band pressure solve [LZF10].

In the absence of our surface-aware basis functions, the hydrostatic solution in a curved bowl yields a linear pressure function. Our coarse solve embeds the high-resolution boundaries and obtains this linear solution easily. The solution is then transferred to the high resolution grid with a linear interpolation. The analytical solution is perfectly represented, and simulations do not introduce volume loss. Our surface-aware basis actually makes the solution non-linear, which adds subtle errors to intermediate frequencies that are not perfectly represented by the coarse solve or removed by Jacobi iterations. Consequently, this simulation without volume correction features some volume loss over time (Figure 9). However, the errors do not introduce any noticeable noise or waves into the simulation, so a simple volume correction scheme is sufficient to remove all visual artifacts.

Our coarse pressure solve optimizes the divergence-free nature of the velocity field in the least-squares sense, and it does not have enough degrees of freedom to satisfy a perfectly divergence-free velocity field. Thus, this approach can produce weak velocity sources and sinks at the finest scales which average out to zero divergence on the coarser scales. Subsequent Jacobi iterations reduce this phenomena.

Although our approach can make the pressure solve more than an order of magnitude faster (see Figure 7), the total simulation speed also depends on other simulation components, such as advection and extrapolation. Thus, if a solver is primarily busy with work outside of the pressure solve, our method will have less impact due to Amdahl's law. The pressure solve took 85% of one time step in Figure 1 at full resolution, leading to the aforementioned factor of 4.3 compared to using a standard pressure projection.

8. Conclusion and outlook

This paper presented an algorithm can significantly speed up liquid simulations, and it can be readily combined with any existing fluid solver that makes use of a pressure projection. Our reduced pressure solver is easily parallelizeable and straightforward to implement, and it clearly outperforms naive coarsening approaches. Our surface-aware pressure basis is the first method for producing a reduced fluid basis that exactly accounts for Dirichlet boundary conditions at any resolution. When combined, these ideas reduce the computational degrees of freedom by an unprecedented factor of 16^3 in the examples throughout this paper, leading to a dramatic reduction in computation time.

Although this paper focuses entirely on liquid simulation, we believe this algorithm also has potential for high impact in other applications with costly Poisson solves, such as the Poisson editing of high-definition images. The method may also be extendable to non-Cartesian domains, with potential benefits in point cloud surface reconstruction or heat kernel calculations on triangle meshes.

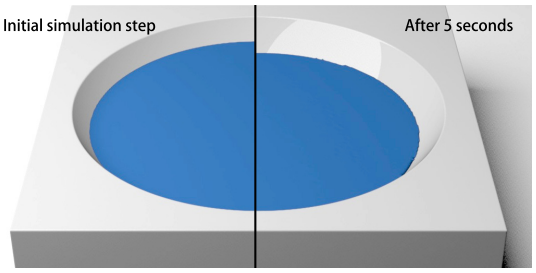

Figure 9: *Without any volume correction, this hydrostatic standing pool in a curved bowl with a coarsening factor of 8^3 results in slight volume loss due to the reduced accuracy at boundaries. However, the errors do not create erroneous surface waves, so a simple volume correction scheme [KLL*07] fixes the problem.*

Acknowledgements

We would like to thank the anonymous reviewers for their help improving this paper. The first author was supported by a JSPS Postdoctoral Fellowship for Research Abroad. Additionally, we also wish to thank Reiji Tsuruno for providing us with computational resources, and Florian Ferstl for the helpful discussions regarding multigrid solvers.

References

[BBB07] BATTY C., BERTAILS F., BRIDSON R.: A fast variational framework for accurate solid-fluid coupling. *ACM Trans. Graph. 26*, 3 (July 2007). 2

[BHW13] BOJSEN-HANSEN M., WOJTAN C.: Liquid surface tracking with error compensation. *ACM Trans. Graph. 32*, 4 (July 2013), 68:1–68:13. 2

[Bri08] BRIDSON R.: *Fluid Simulation for Computer Graphics*. AK Peters/CRC Press, 2008. 2, 3

[CIPT14] CORNELIS J., IHMSEN M., PEER A., TESCHNER M.: Iisph-flip for incompressible fluids. In *Computer Graphics Forum (Proc. Eurographics 2014)* (2014), vol. 33/2, pp. 255–262. 2

[CM11] CHENTANEZ N., MÜLLER M.: Real-time eulerian water simulation using a restricted tall cell grid. *ACM Trans. Graph. 30*, 4 (July 2011), 82:1–82:10. 2

[DWLF12] DE WITT T., LESSIG C., FIUME E.: Fluid simulation using Laplacian eigenfunctions. *ACM Transactions on Graphics (TOG) 31*, 1 (2012), 10. 1, 2

[EB14] EDWARDS E., BRIDSON R.: Detailed water with coarse grids: Combining surface meshes and adaptive discontinuous galerkin. In *ACM Transactions on Graphics (TOG)* (2014), ACM. 2

[FWD13] FERSTL F., WESTERMANN R., DICK C.: Large-scale liquid simulation on adaptive hexahedral grids. *IEEE Transactions on Visualization and Computer Graphics to appear* (2013). 2, 3

[GFCK02] GIBOU F., FEDKIW R. P., CHENG L.-T., KANG M.: A second-order-accurate symmetric discretization of the poisson equation on irregular domains. *Journal of Computational Physics 176*, 1 (2002), 205–227. 2

[GKSB13] GERSZEWSKI D., KAVAN L., SLOAN P.-P., BARGTEIL A. W.: Enhancements to model-reduced fluid simulation. In *Proceedings of the Motion on Games* (2013), ACM, pp. 201–206. 2

[JKNH13] JUNG H.-R., KIM S.-T., NOH J., HONG J.-M.: A heterogeneous cpu–gpu parallel approach to a multigrid poisson solver for incompressible fluid simulation. *Computer Animation and Virtual Worlds 24*, 3-4 (2013), 185–193. 2

[KD13] KIM T., DELANEY J.: Subspace fluid re-simulation. *ACM Transactions on Graphics (TOG) 32*, 4 (2013), 62. 2

[KLL*07] KIM B., LIU Y., LLAMAS I., JIAO X., ROSSIGNAC J.: Simulation of bubbles in foam with the volume control method. In *ACM Transactions on Graphics (TOG)* (2007), vol. 26/3, ACM, p. 98. 5, 8

[KSK09] KIM D., SONG O.-Y., KO H.-S.: Stretching and wiggling liquids. *ACM Transactions on Graphics (TOG) 28*, 5 (2009), 120. 2

[KTJG08] KIM T., THÜREY N., JAMES D., GROSS M.: Wavelet turbulence for fluid simulation. *ACM Trans. Graph. 27*, 3 (Aug. 2008), 50:1–50:6. 2

[KTT13] KIM T., TESSENDORF J., THÜREY N.: Closest point turbulence for liquid surfaces. *ACM Trans. Graph. 32*, 2 (Apr. 2013), 15:1–15:13. 2

[LZF10] LENTINE M., ZHENG W., FEDKIW R.: A novel algorithm for incompressible flow using only a coarse grid projection. *ACM Transactions on Graphics (TOG) 29*, 4 (2010), 114. 1, 2, 7

[MCPN08] MOLEMAKER J., COHEN J. M., PATEL S., NOH J.: Low viscosity flow simulations for animation. In *ACM SIGGRAPH / EG Symposium on Computer Animation* (July 2008), pp. 9–18. 2

[MST10] MCADAMS A., SIFAKIS E., TERAN J.: A parallel multigrid poisson solver for fluids simulation on large grids. In *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2010), Eurographics Association, pp. 65–74. 2

[NB11] NIELSEN M. B., BRIDSON R.: Guide shapes for high resolution naturalistic liquid simulation. *ACM Trans. Graph. 30*, 4 (July 2011), 83:1–83:8. 2

[NSCL08] NARAIN R., SEWALL J., CARLSON M., LIN M. C.: Fast animation of turbulence using energy transport and procedural synthesis. In *ACM Transactions on Graphics (TOG)* (2008), vol. 27, ACM, p. 166. 2

[OF03] OSHER S., FEDKIW R.: *Level set methods and dynamic implicit surfaces*. Springer, 2003. 2

[SSW*13] STANTON M., SHENG Y., WICKE M., PERAZZI F., YUEN A., NARASIMHAN S., TREUILLE A.: Non-polynomial galerkin projection on deforming meshes. *ACM Transactions on Graphics (TOG) 32*, 4 (2013), 86. 2

[TLP06] TREUILLE A., LEWIS A., POPOVIĆ Z.: Model reduction for real-time fluids. *ACM Trans. Graph. 25*, 3 (July 2006), 826–834. 1, 2, 3

[TWGT10] THÜREY N., WOJTAN C., GROSS M., TURK G.: A multiscale approach to mesh-based surface tension flows. *ACM Transactions on Graphics (TOG) 29*, 4 (2010), 48. 2

[WST09] WICKE M., STANTON M., TREUILLE A.: Modular bases for fluid dynamics. In *ACM Transactions on Graphics (TOG)* (2009), vol. 28/3, ACM, p. 39. 2

[YWTY12] YU J., WOJTAN C., TURK G., YAP C.: Explicit mesh surfaces for particle based fluids. *EUROGRAPHICS 2012 30* (2012), 41–48. 2

[ZB05] ZHU Y., BRIDSON R.: Animating sand as a fluid. *ACM Trans. Graph. 24*, 3 (July 2005), 965–972. 2