Resolving Fluid Boundary Layers with Particle Strength Exchange and Weak Adaptivity

Xinxin Zhang*
UBC Computer Science

Minchen Li*
UBC Computer Science

Robert Bridson*
UBC Computer Science, Autodesk Canada

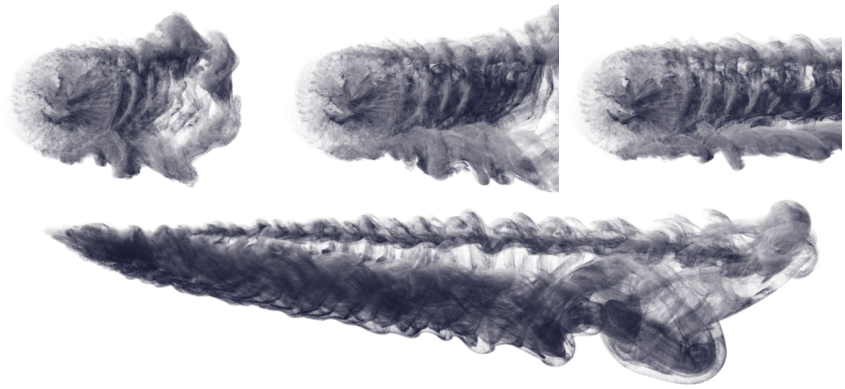

Figure 1: *Simulation of a rotating fan in wind, Re $\approx 10^5$, visualized with smoke marker particles. Large scale motion and detailed boundary layer dynamics are cheaply and easily coupled using our method, producing high quality fluid animations at low cost. Top: top view of fan wake at frames 80, 120 and 200. Bottom: side view at frame 300.*

Abstract

Most fluid scenarios in graphics have a high Reynolds number, where viscosity is dominated by inertial effects, thus most solvers drop viscosity altogether: numerical damping from coarse grids is generally stronger than physical viscosity while resembling it in character. However, viscosity remains crucial near solid boundaries, in the *boundary layer*, to a large extent determining the look of the flow as a function of Reynolds number. Typical graphics simulations do not resolve boundary layer dynamics, so their look is determined mostly by numerical errors with the given grid size and time step, rather than physical parameters. We introduce two complementary techniques to capture boundary layer dynamics, bringing more physical control and predictability. We extend the FLIP particle-grid method with viscous particle strength exchange[Rivoalen and Huberson 2001] to better transfer momentum at solid boundaries, dubbed VFLIP. We also introduce Weakly Higher Resolution Regional Projection (WHIRP), a cheap and simple way to increase grid resolution where important by overlaying high resolution grids on the global coarse grid.

Keywords: fluid simulation, boundary layer

Concepts: •**Computing methodologies** → **Physical simulation;** •**Mathematics of computing** → *Partial differential equations;*

*e-mails: {zhxx, minchenl, rbridson}@cs.ubc.ca

SIGGRAPH '16 Technical Paper,, July 24 - 28, 2016, Anaheim, CA,
ISBN: 978-1-4503-4279-7/16/07 $15.00
DOI: http://dx.doi.org/10.1145/2897824.2925910

ACM Reference Format
Zhang, X., Li, M., Bridson, R. 2016. Resolving Fluid Boundary Layers with Particle Strength Exchange and Weak Adaptivity. ACM Trans. Graph. 35, 4, Article 76 (July 2016), 8 pages. DOI = 10.1145/2897824.2925910 http://doi.acm.org/10.1145/2897824.2925910.

1 Introduction

In computer graphics, the incompressible Navier-Stokes equations are often used to produce realistic fluid animation. Storing fluid quantities on a Cartesian grid and performing pressure projection to handle incompressibility and boundary conditions, Eulerian approaches have proved their merit in scalably handling complex boundary shapes and maintaining stability with large time steps [Bridson 2008].

Despite the ease of use and implementation of Eulerian approaches, their ability to simulate high Reynolds flow remains a problem. In this type of flow, fluid motion is often strongly influenced by boundary layer dynamics. A boundary layer of vanishing thickness usually cannot be resolved at practical grid resolutions, making the no-stick or no-slip boundary conditions both diverge from physical predictability. Besides the possibility of producing inconsistent results under grid refinement, the flow motion is poorly or not at all controlled by changing the Reynolds number, as shown in Fig. 2: simulations at low resolution, which don't resolve the boundary layer, cannot hope to resemble the results produced by high resolution simulations.

Fluid-structure interactions in slightly viscous flow has been successfully modeled by vortex methods [Chorin 1973; Stock and Gharakhani 2010]. Boundaries are viewed as generators of vorticity in vortex methods. However, solving the boundary integral equation for general geometry in 3D is non-trivial, and concerns also remain about how to reliably achieve stability with 3D vortex stretching [Gamito et al. 1995; Zhang and Bridson 2014]: we have found it hard in practice to generally adopt vortex methods for computer graphics.

On the other hand, while the variational framework [Batty et al. 2007] with FLIP advection [Zhu and Bridson 2005] is capable of simulating free-slip boundaries nicely, with insufficient grid resolution the momentum exchange near (more physical) no-slip boundaries is only poorly resolved.

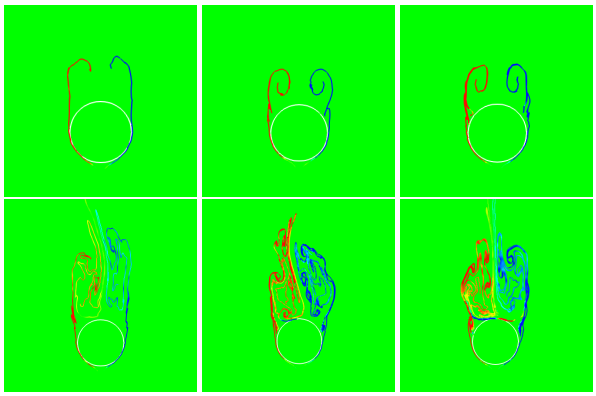

Figure 2: *Vortex shedding from a cylinder in 2D at Re=15000, zoom-in view near the boundary. Left column: simulation results at low resolution (approximately 50×50 grid cells shown). Middle column: simulation obtained with $4\times$ resolution. Right column: simulation obtained with our method, with the same coarse grid but a $4\times$ refined grid overlaid just around the solid. Our method produces results visually consistent with the high resolution reference because only the boundary layer needs that resolving power.*

To capture near-boundary flow accurately without incurring the same expense for the domain as a whole, adaptive methods have been proposed for graphics by many researchers, e.g. using adaptive grids [Losasso et al. 2004; Setaluri et al. 2014; Ando et al. 2013] or domain decomposition [Golas et al. 2012]. While the former category has advantages in more smoothly varying the degree of refinement, there is extra overhead in mesh generation and memory indexing. On the other hand, domain decomposition solvers require extra iterations per time-step to couple the solutions between different domains.

We take inspiration from how, in FLIP, the particles carry flow velocities and move with the flow, while an Eulerian grid is used to adjust the particle velocities to a divergence-free state. In this paper, we extend this philosophy to arrive at an efficient, easy to implement, highly adaptive fluid solver:

- The classic FLIP scheme is augmented with a particle strength exchange (PSE) method [Mas-Gallic 1995] to solve the convection-diffusion part of the Navier-Stokes equations.
- By seeding extra ghost boundary particles at the boundary, our solver captures the boundary layer dynamics more accurately, producing results visually consistent with higher resolution simulations (cf. Fig. 2).
- Our time integration weakly couples the regionally refined solutions efficiently, alleviating the need for sophisticated global solvers.
- The regional refinements can be placed arbitrarily, even overlapping each other, without any geometric operations to merge them together, simplifying mesh generation for spatially adaptive solvers.

2 Related work

Boundary layers are often the source of turbulence in high Reynolds number flows. Pfaff et al. [2009] attempted to capture this effect by seeding vortex particles from a precomputed artificial boundary layer to enhance a coarse simulation. However, their approach is limited to static boundaries and may be less physically plausible than desired as the vortex particles are seeded by chance. In contrast, our method aims to solve the viscous dynamics at the right scale, alleviating the limitation to static boundaries and improving direct physical control of the result.

Table 1: *Symbol abbreviations used throughout this paper.*

ν	Kinematic viscosity of the fluid
$\boldsymbol{u}_{p,i}$	Velocity stored on particle i
$\boldsymbol{x}_i$	Position of particle i
$\boldsymbol{u}_g(\boldsymbol{x})$	Velocity sampled on grid at position $\boldsymbol{x}$
Ω	Fluid domain
$\Omega_{\text{sub},i}$	i^{th} Sub-domain with particle-mesh refinement
$W_h(\boldsymbol{x})$	Kernel used to spread a particle quantity to the grid
$H_\tau(\boldsymbol{x})$	Heat kernel for particle momentum exchange
h	Local grid cell spacing

Figure 3: *Our method captures the velocity field near the boundary efficiently and with high apparent fidelity. Left: zoom-in near the boundary flow. Right: the entire simulation, Re=15000.*

When modelling fluid motion with vortex elements, vorticity can be seeded near the boundary by diffusing the boundary vortex sheet [Park and Kim 2005; Stock and Gharakhani 2010]. In these approaches, the unknown vortex sheet strength is determined by solving boundary integral equations. Besides the expense of the GMRES solve for the vortex street strength, there are questions of solvability of the equations themselves for arbitrary topology under general motion, which is significant for computer graphics applications.

Particles, along with Eulerian velocity projections, have been widely used in computer graphics to capture fluid features which may fall between grid samples and be smoothed away by purely Eulerian solvers, e.g. FLIP [Zhu and Bridson 2005], or derivative particles [Song et al. 2007] and the closely related APIC [Jiang et al. 2015]. We further extend this concept: in our method, particles are used to represent the change of flow momentum due to diffusion and external forces, while multiple-resolution Eulerian grids bounding different subregions are used to project particle momentum towards their divergence-free state (along with a global coarse grid).

Our scheme is related to the Iterated Orthogonal Projection (IOP) framework proposed by Molemaker et al. [2008] but differs in two aspects: instead of re-projecting the field solution globally in consecutive iterations, our projection is only applied to a set of locally refined sub-domains; during each projection our pressure discretization respects the solid boundary conditions as per Batty et al. [2007], while in IOP, boundary conditions are only imposed in a separate step of the iteration.

Chimera grids proposed by English et al. [2013] are a promising tool for adaptive, large-scale fluid computations, but require nontrivial domain discretization to merge grids together, and a relatively expensive global solver. While the global solve may be critical for water simulations, in this paper we focus our attention on purely gaseous phenomena and demonstrate a faster and simpler weak coupling of grids is effective.

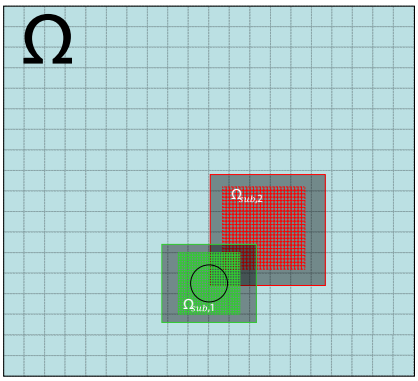

Figure 4: *A typical domain construction used in our method. Ω indicates the global Eulerian domain where fluid motion is loosely captured. Eulerian subdomains with finer resolution can be placed anywhere to enhance the simulation quality locally, such as the green domain($\Omega_{\mathrm{sub},1}$) for near-boundary turbulence and the red domain ($\Omega_{\mathrm{sub},2}$) where the camera was placed. Particles are seeded to fill the space; within gray areas, particles are seeded at higher density to track finer details.*

Domain decomposition schemes have been used as preconditioners for iterative linear solvers within pressure projection (e.g. [Edwards and Bridson 2014]), or as a useful tool to decouple fluid features with different solution methods [Golas et al. 2012]. Our method also makes use of domain decomposition concepts: we solve the convection-diffusion and force calculation part on particles, and use multiple Eulerian grids as an auxiliary tool to perform velocity projection to obtain an adaptive refined and weakly coupled solution in the fluid domain.

Stock et al. [2010] proposed an efficient one-way coupling to simulate rotor wake. The fluid state is first updated with a vortex particle time step, which provides the velocity boundary condition for the Eulerian sub-domain to advance itself. Vortex particles in the Eulerian sub-domain can then interpolate vorticity changes for the next time-step. In contrast, our space-filling Lagrangian particles carry fluid momentum instead of vorticity, to avoid the potential complexities associated with vortex stretching and viscous boundary conditions in general three-dimensional flows.

Multigrid solvers are becoming increasingly popular in graphics to accelerate the pressure solve (e.g. [McAdams et al. 2010; Ferstl et al. 2014], [Ando et al. 2015]) We likewise use multigrid for the various grids in our solver.

While Lentine et al. [2010] used multi-level grids to obtain detailed fluid animations efficiently, their algorithm is limited to purely Eulerian schemes, whereas our solver is designed for hybrid particle-mesh methods like FLIP, which offer less numerical diffusion.

A common technique in industry is to inject additional detail with vorticity confinement [Steinhoff and Underhill 1994; Fedkiw et al. 2001]) or post-process turbulence synthesis (e.g. [Kim et al. 2008; Schechter and Bridson 2008]). We did not use such methods, focusing on a more direct approach of better solving the Navier-Stokes equations at higher Reynolds numbers, where the look can be controlled primarily with actual physical parameters. However, procedural techniques like these could be added as an additional layer to our method to produce even greater details.

3 VFLIP and Weakly Higher Resolution Regional Projection (WHIRP).

The incompressible Navier-Stokes equations we approximately solve are:

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

We adopt the usual FLIP framework to handle the material derivative $D\boldsymbol{u}/Dt$ on the left hand side, storing velocity on particles which are moved in a grid-based velocity field. The pressure gradient and incompressibility condition (together with boundary conditions) are handled by a separate pressure projection step, augmented in this paper with multiple grids (see figure 4). We use particle strength exchange (PSE) for the viscous term, and integrate body forces $\boldsymbol{f}$ with an Euler step split from the rest of the time integration as usual.

An overview of our algorithm is given in Fig. 5, while each subroutine called in a time step is listed in Alg. 1. Technical details of each subroutine are given in the corresponding subsections, §3.1, §3.2,§3.3 and §3.4.

Algorithm 1 TimeStep(Δt)

```
// Convection-diffusion §3.1
For each particle i
    u_{p,i} = PSE(Δtν);
    x_i = ForwardTrace(u_g, Δt, x_i)
// Hierarchical projection §3.2
In fluid domain Ω
    particles = Collect particles in Ω
    DivergenceFreeUpdate(particles)
For each sub-domain Ω_{sub,i} in dx descending order
    particles = Collect particles in Ω_{sub,i}
    DivergenceFreeUpdate(particles)
SeedAndDeleteParticles // §3.4
```

Algorithm 2 DivergenceFreeUpdate(particles)

```
u*_g = Splat particle velocities to grid cells
u_g = Project(u*_g)
δu = u_g − u*_g
For each particle i
    u_{p,i} = u_{p,i} + δu(x_i)
```

3.1 Solving the convection-diffusion equation with ghost particles

Given particles and their velocity at time step n, the convection-diffusion part of the Navier-Stokes equation,

$$\frac{D\boldsymbol{u}}{Dt} = \nu\Delta\boldsymbol{u}, \tag{2}$$

can be solved efficiently with a so-called Particle Strength Exchange method. To deal with boundary objects, at each time step we seed ghost particles randomly in a $2h$ band inside the boundary (where h is the spacing of the grid covering the solid, and with the same density as regular FLIP particle seeding), and set their velocity to that of the solid object. Given particle velocity $\boldsymbol{u}_{p,i}$, the updated particle strength is then calculated using the stable formula

$$\boldsymbol{u}_{p,i}^{n+1} = \boldsymbol{u}_{p,i}^{n} + \frac{\sum_{j\in\eta}(\boldsymbol{u}_{p,j}-\boldsymbol{u}_{p,i})H_\tau(\boldsymbol{x}_i-\boldsymbol{x}_j)}{\sum_{j\in\eta}H_\tau(\boldsymbol{x}_i-\boldsymbol{x}_j)} \tag{3}$$

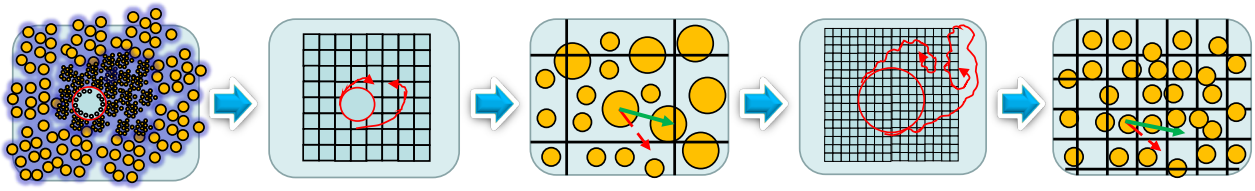

Figure 5: *Overview of our algorithm. At the beginning of each time step, the velocity of each particle is known, the convection-diffusion part of Navier-Stokes equation is solved by a PSE method (§3.1). We then splat the particle velocity to the coarse grid, and make the resulting field divergence-free §3.2. The velocity change is interpolated to all particles for divergence correction. Then, for any subdomain $\Omega_{sub,i}$ in the field, the corrected particle momentum is splatted to the grid again and made locally divergence-free. Any particle within a subdomain $\Omega_{sub,i}$ is considered a small-scale particle and collects its momentum correction from the corresponding domain.*

where η is the neighborhood in which we look for particles (a box of size $2h$), and $H_\tau(\boldsymbol{x})$ is the heat kernel with $\tau = \nu\delta t$, which reads

$$H_{\nu\delta t}(\boldsymbol{x}) = \frac{1}{(4\pi\nu\delta t)^{d/2}} \exp\left(\frac{-\|\boldsymbol{x}\|^2}{4\nu\delta t}\right) \tag{4}$$

The $(4\pi\nu\delta t)^{-d/2}$ factor can be elided due to the normalization in Equation 3.

Once the particle velocites have been diffused, particles are passively advected through the divergence-free flow field. We used Ralston's 3rd order Runge-Kutta time integrator [1962], as the most efficient optimal RK scheme with a stability region overlapping the pure imaginary axis (for rotations):

$$\begin{aligned}
\boldsymbol{u}_i' &= \text{SampleVelocity}(\boldsymbol{x}_i) \\
\boldsymbol{u}_i'' &= \text{SampleVelocity}(\boldsymbol{x}_i + 0.5\Delta t\boldsymbol{u}_i') \\
\boldsymbol{u}_i''' &= \text{SampleVelocity}(\boldsymbol{x}_i + 0.75\Delta t\boldsymbol{u}_i'') \\
\boldsymbol{x}_i^{n+1} &= \boldsymbol{x}_i^n + \frac{2}{9}\Delta t\boldsymbol{u}' + \frac{3}{9}\Delta t\boldsymbol{u}'' + \frac{4}{9}\Delta t\boldsymbol{u}'''
\end{aligned} \tag{5}$$

When sampling velocity from the velocity buffer, we use trilinear interpolation. In the case where multiple overlapping grids are detected, we take the grid with the finest resolution. While this sounds overly simple, and potentially introduces discontinuities at the edges of grids, we have not observed any noticeable artifacts as typically the velocity jumps would be small, and it is only the integral in time of the velocity (the particle trajectories) which we observe. It would not be difficult to use a partition-of-unity blend to get a smoother velocity at grid edges if for some reason this was necessary. After advection, we further update the particle velocities with forcing terms such as buoyancy before pressure projection.

3.2 Regional projection for particle velocity correction

Once the post advection particle velocities are known, we use a regional projection method to correct the velocity field to an approximately divergence-free state. Following Batty et al. [2007], finding the inviscid divergence-free projection of an input velocity field $\boldsymbol{u}^*$ is equivalent to a minimization problem:

$$\min_p \int_\Omega \frac{1}{2}\rho\|\boldsymbol{u}^* - \frac{\Delta t}{\rho}\nabla p\|^2 - \int_\Omega \frac{1}{2}\rho\|\boldsymbol{u}^*\|^2 + \int_S p\hat{n}\cdot\Delta t\boldsymbol{u}_{pre}, \tag{6}$$

where $\boldsymbol{u}_{pre}$ is typically a prescribed velocity at solid or fluid domain boundaries S.

As outlined in Alg. 2, in our regional projection, we first splat all particle velocities to the Eulerian domain Ω with a wide kernel $W_h(\boldsymbol{x})$, as if we were computing the large eddy filtered version of the velocity field:

$$\boldsymbol{u}_g(\boldsymbol{x}) = \frac{\sum_j \boldsymbol{u}_j W_h(\boldsymbol{x} - \boldsymbol{x}_j)}{\sum_j W_h(\boldsymbol{x} - \boldsymbol{x}_j)}. \tag{7}$$

Here $\boldsymbol{x}$ is a fixed grid cell position, and $\boldsymbol{x}_j$ and $\boldsymbol{u}_j$ are the the position and velocity of particle j respectively.

In our implementation, the usual bi-/trilinear hat function kernel was used:

$$\begin{aligned}
W_h(\boldsymbol{x}) &= \prod_{\theta=1}^{d} \phi\left(\frac{\boldsymbol{x}_\theta}{h}\right), \text{with} \\
\phi(r) &= \begin{cases} 1-|r|, & r \in [-1,1), \\ 0, & \text{else}, \end{cases}
\end{aligned} \tag{8}$$

where $d = 2, 3$ is the spatial dimension, h is the kernel width which simply equals the grid spacing, and $\boldsymbol{x}_\theta$ is the θ^{th} component of vector $\boldsymbol{x}$.

After we splat the particle velocity to the grid, the velocity field on the grid is made divergence-free by pressure projection. Our projection uses a standard staggered grid and face-weighted pressure discretization similar to Batty et al. [2007]. The velocity change is then interpolated to the particles to correct their velocities.

At this stage the velocity defined on the particles is (approximately) divergence-free at the coarse scale. To increase the flow fidelity in regions of interest (near boundaries or cameras, for example), further projections with higher resolution in each subdomain are performed with essentially the same procedure. We splat the coarse-corrected particle velocity to the corresponding refined domain using the kernel $W_h(\boldsymbol{x})$ with the smaller refined h, make the grid-based velocity field divergence-free, and interpolate the velocity change back to the particles in that region. In each subdomain projection, at the domain boundaries we restrict the normal component of velocity not to change (as established by splatting from the particles). We repeat this process until all the regional refined domains are processed.

During each projection, the Poisson equation for pressure is solved with a multi-grid preconditioned conjugate gradient method, whose implementation details are given in §3.3.

3.3 Our pressure solver

Similar to McAdams et al. [2010] and Setaluri et al. [2014], we use a multigrid preconditioned Krylov solver for our pressure Poisson equation. Our multigrid solver is constructed semi-algebraically: we construct $\mathcal{R}_L$ (the restriction matrix mapping the residual of level L to a coarser level) and $\mathcal{P}_L$ (the prolongation matrix used

to add coarse level corrections back to the level L solution) using geometric information given by the fluid domain, but use matrix multiplications to construct the coefficient matrix A_{L+1} of each coarsening level $L+1$ (Alg. 3).

Algorithm 3 MultiGridCoarsening(A_L,$\mathcal{P}_L$,$\mathcal{R}_L$, A_{L+1})

```
for each cell in coarse level
  i ← index of this cell
  for each sub cell in fine level
    j ← index of this sub cell
    if A_L(j, j) ≠ 0
      R_L(i, j) = 1/2^d //d = 2,3 dimension of the problem
      P_L(j, i) = 1.0
A_{L+1} ← 0.5 R_L A_L P_L
```

As shown in Alg. 3, our restriction and prolongation matrices are transposes of each other, up to a scale factor, with restriction equivalent to simply averaging residuals across the fine level grid cells covered by a coarse grid cell, and prolongation equivalent to taking the nearest neighbor coarse value; this is a "aggregation"-style multigrid. For pure multigrid, this does not produce optimal convergence; we improve the scheme by applying a scaling of $\frac{1}{2}$ to the coarse level coefficient matrix, which, away from solid boundaries, gives exactly the same coefficient matrix as if we were to rediscretize the PDE on the coarse grid. For cases with solid boundaries, this coarsening strategy acts as a volume weighted rediscretization at the coarse level. Our coarsening strategy doesn't increase the size of the discrete Poisson stencil: a coarse cell has at most six neighbors with non-zero coefficients. This in turn allows for the usual red-black ordering to parallelize Gauss-Seidel sweeps, and in general makes smoothing far more efficient. For other implementation details of multigrid-preconditioned Krylov solvers, please refer to McAdams et al. [2010].

In all our experiments, the multigrid-preconditioned conjugate gradient solver appeared to converge with a constant rate of approximately 0.1 per iteration, though more complex schemes may be necessary for extremely complex solid geometry.

3.4 Seeding and deleting particles

At the end of each time step, particles are seeded and deleted as necessary to maintain reasonable sampling of the fluid in each grid cell, according to the highest resolution grid nearby. Newly seeded particles take their velocity by interpolation from the finest available grid. We delete particles randomly from cells where the count exceeds a threshold (16 in our examples), and seed new particles where the count is below another threshold (8 in our examples).

Referring to figure 4, the grey areas surrounding (but not inside) refined subdomains are treated with the finer grid spacing: we maintain a higher particle density in the grey areas as there is a chance that the fluid in the grey region will be advected into the refined subdomain during the course of the next time step. For our examples we used a bandwidth of $3h$, where h is the refined grid spacing, for this intermediate zone but it could of course be determined more dynamically and frugally based on the local velocity and the time step size.

As an alternative to avoid higher particle counts, we have experimented with only seeding extra particles in the refined regions, not the grey neighborhoods, but widening the support of the particle-to-grid kernel near the edge of the refined grid to avoid gaps in the data. This exchanges more work for less storage, which may be preferable in some cases, though the results are similar.

Figure 6: *Impulsively started flow past a sphere at Re=15000 simulated using our solver*

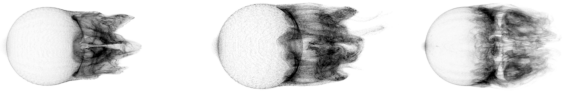

Figure 7: *Vortex shedding in impulsively started flow around a $2m$ diameter sphere at Re=20000. Left: naïve grid-based viscosity model with $h = 0.0625$. Middle: naïve solver with $h = 0.03125$ (twice the resolution). Right: our model using a coarse grid of $h = 0.125$ and a refined grid around the sphere with $h = 0.03125$. The computation time for the left and right simulations are roughly equal.*

4 Results and discussion

We parallelized the proposed methods on a desktop machine with an Intel(R) Core(TM) i7-3930K CPU and 32 GB RAM. We compared our new solver to a typical inviscid FLIP solver as used in graphics, as well as a "naïve" viscous Navier-Stokes solver where viscosity is added with an explicit-in-time finite difference (since we are interested in very high Reynolds numbers, this explicit step was always stable). Simulations were visualized by judiciously injecting smoke marker particles which are passively advected with the grid velocity field. The accompanying video shows comparisons between the methods for different resolutions and different Reynolds numbers in a variety of examples.

In Fig. 6, we show a flow simulation frame obtained with our solver at Reynolds number of 15000: the flow separates from the boundary at small angle, it remains laminar with structured vortices for about one diameter, and then becomes turbulent. This is similar to figure 55 on page 34 of An Album of Fluid Motion [1982].

As shown in Fig. 7, with the naïve uniform-resolution grid-based viscosity model, a much higher resolution is necessary to capture important fluid motions such as vorticity detaching from the boundary layer. With our sub-grid particle-based viscosity model and regional refinement, the characteristic vortex shedding can be achieved at much lower cost.

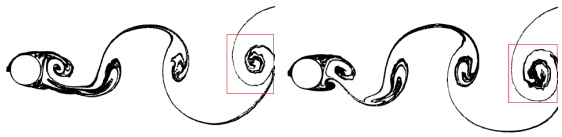

Figure 8: *For a flow past cylinder simulation at $Re = 800$, inflow speed $\mathrm{U} = 5m/s$, and cylinder diameter $D = 1.6m$, the experimental model predicts a Strouhal number of around 0.2 [Lienhard 1966], hence a shedding frequency of around 0.625. The above pictures are from frame 754 and frame 920 of our simulation with $\Delta t = 0.01$, where vortices shed off the same side just reach the end of the domain; this gives a shedding frequency just over 0.6.*

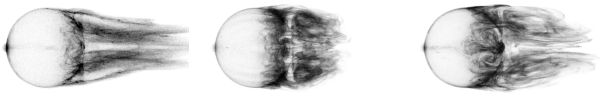

Figure 9: *Our method on the same scenario as Fig. 7 with different resolutions. Left: coarse simulation (global $h = 0.25$, inner $h = 0.0625$). Middle: $2\times$ refinement for both (global $h = 0.125$, inner $h = 0.03125$). Right: only the inner grid is refined (global $h = 0.25$, inner $h = 0.03125$). Important fluid features can be cheaply captured by only increasing the inner grid resolution.*

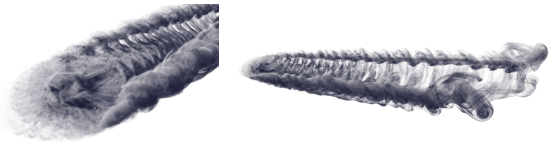

Figure 10: *Rotating fan (each blade is $1.3m$ long, rotating at one cycle per second) in a constant wind of $2m/s$, with $\nu = 10^{-4}$, giving Re $\approx 10^5$. The refined grid has $h = 0.005m$. Left: zoom-in view at the fan blades. Right: the entire simulation.*

In Fig. 8 we performed a 2D flow past cylinder simulation. The cylinder is only about 12 cells wide in the coarse grid; a $4\times$ local refinement is applied to resolve the boundary layer. The shedding frequency produced by our solver matches the experimental model.

In Fig. 9 we take the same scenario (impulsively started flow around a sphere at Re=20000) but look at changing global and inner grid resolution in our method. Vortex shedding is largely dominated by how well a solid boundary is resolved. Our method can resolve some sub-grid scale viscosity effects, but it fails to predict the near-boundary behavior when the refined grid is too coarse to resolve it. However, increasing the resolution of the local grid just around the boundary improves the simulation quality greatly without requiring refinement in the rest of the domain.

When the resolution of the coarse domain isn't enough to represent the solid object, our regional refinement may still be enough to capture the solid boundary well (Fig. 1, Fig. 10). Vortex shedding from solid objects is essential to fluid animation in these scenarios, which can be faithfully captured by our method.

Our method makes it much cheaper to adequately resolve the viscous boundary layer, locally and on-the-fly, alleviating the limitation to parametrize boundary layer with far field flow motions as in Pfaff et al. [2009]. It is thus suitable for more general simulation scenarios, e.g. buoyancy-driven plumes as shown in Fig. 11.

5 Conclusions, limitations, and future work

We extended incompressible FLIP in two ways, with a particle-based viscosity model that can resolve momentum exchange at higher resolution than the grid, and with an extremely simple regional projection method (together with particle seeding/deletion rules) to adaptively refine in important areas, without needing more complex grid or mesh structures to globally couple different resolutions. Very little extra code beyond a standard FLIP simulator is required to implement this paper. Moreover we showed that matching the qualitative look of some high Reynolds numbers scenarios hinges just on resolving the viscous boundary layer, even while the rest of the domain can use a coarse grid.

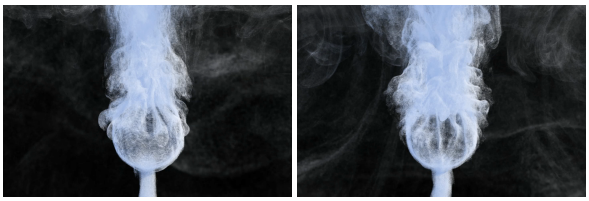

Figure 11: *Smoke rising around a sphere of diameter 0.3m, with $\nu = 10^{-4}$ and the inner grid $h = 0.005m$, Re $\approx 10^3$. Left: simulation with a uniform FLIP solver. Right: simulation using our solver with a coarser global grid and a $4\times$ refined grid around the sphere. With approximately the same computation time, our solver captures structures in the boundary disturbances over the bottom of the sphere more sharply.*

The typical inviscid solver used in graphics doesn't take viscosity into account at all, and for example can only lead to vortex shedding through numerical errors related to grid size and time step. Both the naïve method and our method will fail to give characteristic results when the boundary layer isn't resolved, again leading to simulations whose look is more controlled by numerical errors related to grid size and time step than physical parameters. However, our method allows for resolving boundary layers much more efficiently, at which point physical parameters are at last a useful control and further refinement will reliably give similar but more detailed results, as artists generally hope for.

Our solver is still limited formally to first order accuracy in time, and the PSE approach is likely only first order in space: while we feel it's valuable for coarse grids and large time steps, it is not an efficient approach for convergence to a fully accurate solution. The PSE part is also inappropriate for highly viscous flows, where an implicit grid-based solver is almost certainly the better option.

The global time step is also a limitation of this approach to spatial-only adaptivity. The time step selected in graphics simulations, when multiple steps per frame are taken, is frequently determined by restricting the CFL number, i.e. limiting the number of grid cells a fluid particle may travel through in one time step. In refined subdomains with smaller grid cells, this is a much more stringent restriction than is needed for the coarse global grid, wasting some computational effort.

We have not considered fluid phenomena beyond simple smoke scenarios. Fire and explosions, where nonzero divergence may be prescribed by the pressure projection, would require some adjustment to our regional projection scheme. Free surface liquids, such as commonly used for water, would require even more thought if regional grids overlap the free surface.

There are several other avenues for further research. The convection-diffusion part we currently handle with PSE might be improved with other ideas from grid-free methods; we could also look at using an effective eddy viscosity to better model unresolved turbulence. Our regional projection is not limited to regular grids, but could also use unstructured tetrahedral meshes to conform more accurately to solid boundaries (while avoiding the need for a global, conforming, tetrahedral mesh, which could be a useful savings for simulations involving moving rigid bodies). Last but not least, it is possible to also track the vorticity on each particle to correct the missing angular momentum during particle advection, similar to Zhang et al. [2015]; replacing the FLIP framework with APIC [Jiang et al. 2015] might also allow for better tracking of vorticity.

Table 2: *Timings of various simulations associated with this paper, measured on a desktop machine with Intel(R) Core(TM) i7-3930K CPU and 32 GB RAM.*

Example	Method	Δt	Outer Grid Size	Inner Grid Size	Reynolds Number	Simulation Time (seconds per frame)
Flow past sphere	Naïve solver	0.03	160 x 80 x 80	N/A	1,000	12.52
		0.03	320 x 160 x 160	N/A	20,000	73.46
	Our method	0.03	80 x 40 x 40	48 x 64 x 64	1,000	11.21
		0.03	160 x 80 x 80	96 x 128 x 128	20,000	70.76
		0.03	80 x 40 x 40	96 x 128 x 128	15,000	56.07
Rotating fan in wind	Our method	0.03	240 x 120 x 120	144 x 96 x 144	$\approx 10^5$	120.40
Rising smoke plume		0.05	50 x 100 x 50	100 x 200 x 100	$\approx 10^3$	79.10
Flow past sphere	FLIP	0.03	160 x 80 x 80	N/A	N/A	11.31
	FLIP + WHIRP	0.03	80 x 40 x 40	40 x 40 x 40	N/A	4.06
	VFLIP	0.03	160 x 80 x 80	N/A	N/A	15.41
	VFLIP + WHIRP	0.03	80 x 40 x 40	40 x 40 x 40	N/A	4.20

Table 3: *Timings of flow past sphere simulations with only differences in outer and inner grid sizes, measured on a desktop machine with Intel(R) Core(TM) i7-3930K CPU and 32 GB RAM.*

Simulation Time (seconds per frame)		Outer Grid Size 100x50x50	150x75x75	200x100x100
Inner Grid Size	66x66x66	12.19	19.71	34.36
	100x100x100	28.99	35.46	50.52
	200x200x200	166.14	168.64	185.57

6 Acknowledgments

This research was supported by a grant from the Natural Sciences and Engineering Research Council of Canada.

References

ANDO, R., THÜREY, N., AND WOJTAN, C. 2013. Highly adaptive liquid simulations on tetrahedral meshes. *ACM Trans. Graph. (Proc. SIGGRAPH 2013)* (July).

ANDO, R., THÜREY, N., AND WOJTAN, C. 2015. A dimension-reduced pressure solver for liquid simulations. *EUROGRAPHICS 2015*.

BATTY, C., BERTAILS, F., AND BRIDSON, R. 2007. A fast variational framework for accurate solid-fluid coupling. *ACM Trans. Graph. (Proc. SIGGRAPH) 26*, 3, 100.

BRIDSON, R. 2008. *Fluid Simulation for Computer Graphics*. A K Peters / CRC Press.

CHORIN, A. J. 1973. Numerical study of slightly viscous flow. *Journal of Fluid Mechanics Digital Archive 57*, 04, 785–796.

EDWARDS, E., AND BRIDSON, R. 2014. Detailed water with coarse grids: combining surface meshes and adaptive discontinuous Galerkin. *ACM Trans. Graph. (Proc. SIGGRAPH) 33*, 4, 136:1–9.

ENGLISH, R. E., QIU, L., YU, Y., AND FEDKIW, R. 2013. Chimera grids for water simulation. In *Proceedings of the 12th ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, ACM, New York, NY, USA, SCA '13, 85–94.

FEDKIW, R., STAM, J., AND JENSEN, H. W. 2001. Visual simulation of smoke. In *Proc. ACM SIGGRAPH*, 15–22.

FERSTL, F., WESTERMANN, R., AND DICK, C. 2014. Large-scale liquid simulation on adaptive hexahedral grids. *Visualization and Computer Graphics, IEEE Transactions on PP*, 99, 1–1.

GAMITO, M. N., LOPES, P. F., AND GOMES, M. R. 1995. Two-dimensional simulation of gaseous phenomena using vortex particles. In *Computer Animation and Simulation '95*, Eurographics. Springer Vienna, 3–15.

GOLAS, A., NARAIN, R., SEWALL, J., KRAJCEVSKI, P., DUBEY, P., AND LIN, M. 2012. Large-scale fluid simulation using velocity-vorticity domain decomposition. *ACM Transactions on Graphics (TOG) 31*, 6, 148.

JIANG, C., SCHROEDER, C., SELLE, A., TERAN, J., AND STOMAKHIN, A. 2015. The affine particle-in-cell method. *ACM Trans. Graph. 34*, 4 (July), 51:1–51:10.

KIM, T., THUREY, N., JAMES, D., AND GROSS, M. H. 2008. Wavelet turbulence for fluid simulation. *ACM Trans. Graph. (Proc. SIGGRAPH) 27*, 3, 50.

LENTINE, M., ZHENG, W., AND FEDKIW, R. 2010. A novel algorithm for incompressible flow using only a coarse grid projection. *ACM Trans. Graph. 29*, 4 (July), 114:1–114:9.

LIENHARD, J. 1966. *Synopsis of Lift, Drag, and Vortex Frequency Data for Rigid Circular Cylinders*. Bulletin (Washington State University. College of Engineering. Research Division). Technical Extension Service, Washington State University.

LOSASSO, F., GIBOU, F., AND FEDKIW, R. 2004. Simulating water and smoke with an octree data structure. *ACM Trans. Graph. (Proc. SIGGRAPH) 23*, 3, 457–462.

MAS-GALLIC, S. 1995. Particle approximation of a linear convection-diffusion problem with Neumann boundary conditions. *SIAM J. Numer. Anal. 32*, 4 (Aug.), 1098–1125.

MCADAMS, A., SIFAKIS, E., AND TERAN, J. 2010. A parallel multigrid poisson solver for fluids simulation on large grids. In *Proc. ACM SIGGRAPH / Eurographics Symp. Comp. Anim.*, 65–74.

MOLEMAKER, J., COHEN, J. M., PATEL, S., AND NOH, J. 2008. Low viscosity flow simulations for animation. In *SCA '08: Pro-*

ceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation, Eurographics Association, 9–18.

PARK, S. I., AND KIM, M.-J. 2005. Vortex fluid for gaseous phenomena. In *Proc. ACM SIGGRAPH / Eurographics Symp. Comp. Animation*, 261–270.

PFAFF, T., THUEREY, N., SELLE, A., AND GROSS, M. 2009. Synthetic turbulence using artificial boundary layers. *ACM Trans. Graph. 28*, 5, 121.

RALSTON, A. 1962. Runge-Kutta methods with minimum error bounds. *Mathematics of Computation 16*, 80, 431–437.

RIVOALEN, E., AND HUBERSON, S. 2001. The particle strength exchange method applied to axisymmetric viscous flows. *J. Comput. Phys. 168*, 2 (Apr.), 519–526.

SCHECHTER, H., AND BRIDSON, R. 2008. Evolving sub-grid turbulence for smoke animation. In *Proceedings of the 2008 ACM/Eurographics Symposium on Computer Animation*.

SETALURI, R., AANJANEYA, M., BAUER, S., AND SIFAKIS, E. 2014. SPGrid: A sparse paged grid structure applied to adaptive smoke simulation. *ACM Trans. Graph. (Proc. SIGGRAPH Asia) 33*, 6, 205:1–205:12.

SONG, O.-Y., KIM, D., AND KO, H.-S. 2007. Derivative particles for simulating detailed movements of fluids. *IEEE Trans. Vis. Comp. Graph. 13*, 4, 711–719.

STEINHOFF, J., AND UNDERHILL, D. 1994. Modification of the Euler equations for "vorticity confinement": Application to the computation of interacting vortex rings. *Physics of Fluids 6*, 2738–2744.

STOCK, M. J., AND GHARAKHANI, A. 2010. A GPU-accelerated boundary element method and vortex particle method. In *AIAA 40th Fluid Dynamics Conference and Exhibit (July 2010)*, 1.

STOCK, M. J., GHARAKHANI, A., AND STONE, C. P. 2010. Modeling rotor wakes with a hybrid OVERFLOW-vortex method on a GPU cluster. In *28th AIAA Applied Aerodynamics Conference*, vol. 20.

VAN DYKE, M. 1982. *An album of fluid motion*. Parabolic Press.

ZHANG, X., AND BRIDSON, R. 2014. A PPPM fast summation method for fluids and beyond. *ACM Trans. Graph. (Proc. SIGGRAPH Asia) 33*, 6, 206:1–11.

ZHANG, X., BRIDSON, R., AND GREIF, C. 2015. Restoring the missing vorticity in advection-projection fluid solvers. *ACM Trans. Graph. 34*, 4 (July), 52:1–52:8.

ZHU, Y., AND BRIDSON, R. 2005. Animating sand as a fluid. *ACM Trans. Graph. (Proc. SIGGRAPH) 24*, 3, 965–972.