Eurographics/ ACM SIGGRAPH Symposium on Computer Animation (2008)
M. Gross and D. James (Editors)

Evolving Sub-Grid Turbulence for Smoke Animation

H. Schechter[†1] and R. Bridson[‡1]

[1]The University of British Columbia, Canada

Abstract

We introduce a simple turbulence model for smoke animation, qualitatively capturing the transport, diffusion, and spectral cascade of turbulent energy unresolved on a typical simulation grid. We track the mean kinetic energy per octave of turbulence in each grid cell, and a novel "net rotation" variable for modeling the self-advection of turbulent eddies. These additions to a standard fluid solver drive a procedural post-process, layering plausible dynamically evolving turbulent details on top of the large-scale simulated motion. Finally, to make the most of the simulation grid before jumping to procedural sub-grid models, we propose a new multistep predictor to alleviate the nonphysical dissipation of angular momentum in standard graphics fluid solvers.

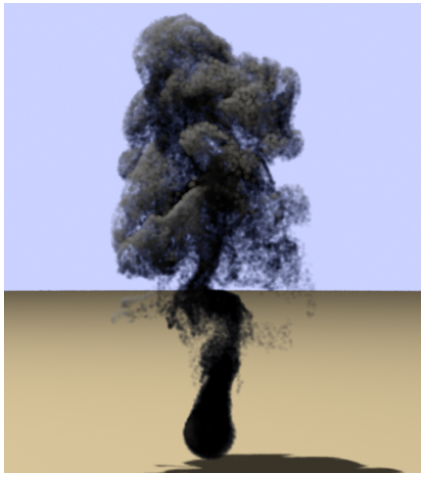
Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism

1. Contributions

Animating turbulent fluid velocity fields, from the delicate swirls of milk stirred into coffee to the violent roiling of a volcanic eruption, poses serious challenges. While the look of the large-scale components of motion are well captured by direct simulation of the fluid equations, increasing the grid resolution to capture the smallest turbulent scales hits a severe scalability problem. The usual solution of augmenting a coarse simulation with procedurally synthesized turbulent detail generally is limited by the visual implausibility of this detail: while some statistics or invariants may be matched, visually important aspects of the time evolution of turbulence are still missing.

This paper attempts to bridge the gap by introducing a turbulence model that simulates the transfer of energy in subgrid scales, then providing a procedural method for instantiating a high resolution turbulent velocity field from this data, which can be evaluated on the fly for a marker particle pass when rendering. For each octave of sub-grid detail, in each grid cell, we evolve both a kinetic energy density E and a net rotation θ for generating the turbulence.

In addition, since full fluid simulation is generally preferred to procedural models when feasible, we address one of the chief remaining sources of nonphysical energy dissipation in graphics fluid solvers, adding a predictor step to the usual time splitting of the incompressible Euler equations. This helps make the most of a limited grid size.



2. Background

The field of turbulence modeling is vast; for an overview we refer readers to the recent text by Pope [Pop05] or the earlier

† e-mail: hagitsch@cs.ubc.ca
‡ e-mail: rbridson@cs.ubc.ca

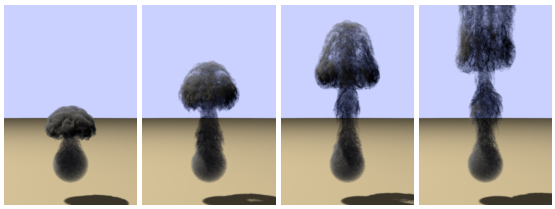


Figure 1: *Frames from an animation with the new turbulence model.*

classic by Tennekes and Lumley [TL72]. For an introduction to basic fluid simulation techniques within graphics, see Bridson and Müller-Fischer's course notes [BMF07].

There is a huge disparity in length scales for turbulent flow: e.g. in the atmosphere large-scale features may be measured in kilometres, with the smallest features in millimetres. A fine enough grid to capture everything (the idea behind the Direct Numerical Simulation approach in science) may be enormous: n^3 grid cells with $n > 1000$. Unfortunately, turbulence fills the entire volume with fine-scale features, eliminating the benefit of adaptive grid methods, and to resolve the small scales time steps must be proportional to the grid spacing, resulting in at least $O(n^4)$ work. This severely limits the scalability of straight simulation for capturing turbulence.

However, turbulence research has developed higher level models which promise efficient simulation, based on the observation that smaller scales in turbulence quickly become isotropic and more or less statistically independent. This allows useful notions of average effect and evolution without need of resolving all details, with the chief visible actions being:

- Enhanced mixing, which when averaged over appropriate length and time scales can be interpreted as a diffusion process, with a so-called "eddy viscosity". This is why stirring milk in coffee is much more effective than letting it sit still.
- Transfer of energy from large-scale structures to smaller-scale eddies and ultimately to the smallest scales where molecular viscosity takes over to dissipate kinetic energy into heat. This is the cause of such a large range of length scales.

The simplest model that captures these, the Kolmogorov $5/3$ power law, has long been used in graphics for synthesizing velocity fields with plausible statistics. Assuming uniform, steady, self-preserving turbulence, where the energy continuously injected at the largest scales matches the energy dissipated by viscosity at the smallest scales and an equilibrium in the intermedate scales, this provides a simple formula for the amount of energy and rate of fluctuation in each frequency band.

Many researchers have augmented fluid simulations with sub-grid turbulence models, i.e. directly simulating the large scales while estimate the effect of unresolved smaller scales. These methods separate the velocity field into a sum of the "mean flow" (the large-scale components) and a turbulent fluctuation. For the Reynolds Averaged Navier-Stokes (RANS) approach, the mean-flow averaging is taken over appropriate time scales, and for the Large Eddy Simulation (LES) approach the averaging is done with a spatial kernel such as a Gaussian. Averaging the Navier-Stokes equations gives rise to very similar equations for the mean flow, but with the addition of a "Reynolds stress" giving the effect of turbulent mixing on the mean flow—at its simplest modeled as a nonlinear viscosity. For this paper, where we try to capture the qualitative look but don't provide quantitatively accurate results, we assume numerical dissipation in the usual first-order accurate fluid solvers dominates this term and thus ignore it.

Some turbulence methods go further by tracking information about the turbulent fluctuations to produce better estimates of their effect on the mean flow. We focus in particular on the popular $k-\varepsilon$ model, originating in work by Harlow and Nakayama [HN67]. Here two more fields are added to the simulation, k being the kinetic energy density of the turbulent fluctations (energy per unit volume) and ε representing the rate of energy dissipation in the viscous scales. Unlike our new model, this does not explicitly track the cascade of energy from large scales to small scales, but like our model simulates the spatial transport, diffusion, and ultimate dissipation of turbulent energy.

Turning to computer graphics, Shinya and Fournier [SF92] and Stam and Fiume [SF93] used the Kolmogorov $5/3$ law and the inverse Fourier transform to generate incompressible velocity fields with plausible statistics, possibly layered on top of a large-scale flow. This was later also used by Rasmussen et al. [RNGF03] to break up smoothness in the interpolation of 3D velocity fields from simulated 2D slices. Kniss and Hart [KH04] demonstrated that by taking the curl of vector valued noise functions, plausible incompressible velocity fields can be created without need for Fourier transforms; Bridson et al. [BHN07] extended the curl-noise approach to respect solid boundaries and conveniently construct larger-scale patterns, and illustrated the attraction of spatial modulating turbulence. Most recently Kim et al. [KTJG08] combined curl-noise with a wavelet form of the $5/3$ law, extending the energy density measured in the highest octave of a simulation into a turbulent detail layer and further advecting the noise texture with the flow to capture spatial transport of turbulent flow features, with marker particles for rendering.

Neyret's work on advected textures [Ney03] is closest in spirit to this paper. He introduces a single representative vorticity vector per simulation grid cell and per octave of turbulence, advected with the flow. The vectors are gradually blended from coarse octaves to fine octaves, and are

used to rotate the gradients in flow noise [PN01]. The flow noise is then used to distort a texture map (e.g. a smoke-like hypertexture), with smooth regeneration of texture detail via blending of the original when total deformation has increased past a limit. We instead track (and conserve in all but the finest octave) kinetic energy, include the spatial diffusion of kinetic energy, and properly decouple the rotation of nearby gradient vectors in flow noise (so not all of the gradient vectors in one simulation cell are rotated with the same vorticity); we further combine it with curl-noise to produce velocity fields for marker particle advection in rendering.

Our predictor method for reducing nonphysical rotational dissipation in the large-scale fluid simulation also has antecedents in CFD and graphics. The chief culprits are in the treatment of the advection term in the momentum equation. Fedkiw et al. [FSJ01] observed that some of the strong numerical dissipation of the trilinear interpolation semi-Lagrangian method introduced by Stam [Sta99] can be reduced by using a sharper Catmull-Rom-based interpolant instead; many other improvements to pure advection followed (e.g. Kim et al.'s BFECC approach [KLLR05]) culminating in Zhu and Bridson's FLIP method [ZB05] which essentially eliminates all numerical dissipation from advection via use of particles. However, this is not the only source of dissipation: the first order time splitting used in graphics, where velocities are separately advected and then projected to be incompressible, also introduces large errors. In particular, angular momentum is not conserved even with a perfect advection step, as can readily be seen if a rigidly rotating fluid is advected by 90° in one time step: the angular velocity would be entirely transferred into a divergent field, which pressure projection would subsequently zero out. In the scientific literature, predictor-corrector, multistep or Runge-Kutta methods are used in conjunction with high resolution discretizations of the advection term and small time steps to avoid this problem. However, these aren't directly applicable to the large time steps taken with semi-Lagrangian advection or FLIP in graphics.

Vorticity confinement [FSJ01] helps recover some of the lost angular momentum, by adding artificial forces to enhance spin around local maxima of the existing vorticity. Selle et al. [SRF05] extended this with spin particles, to boost the smallest-scale vorticity present on the grid. However neither technique is applicable to the simplest case, reducing the dissipation in rigid rotation, and thus we propose a more general solution in this paper.

We also note that alternative approaches to fluid simulation, in particular vortex particle methods (e.g. [YUM86, Gam95, AN05, PK05]), do not suffer from this dissipation. However vorticity methods have their own share of problems, particular in 3D, such as in handling free surface and solid boundary conditions, thus most work has centred around velocity/pressure methods.

3. Characterizing Sub-Grid Turbulence

Our goal is to simulate the average behavior of the sub-grid turbulence, leaving the details of instantiating a plausible velocity field to a post-simulation phase. In particular, we want to describe the turbulence without recourse to higher resolution grids. Note that we assume that the combination of FLIP with our new multistep time integration will capture all grid-level turbulence directly in simulation, thus we only focus on the missing sub-grid part.

We extend the $k-\varepsilon$ approach of tracking mean kinetic energy density: instead of a single total kinetic energy density per grid cell, we break it up into octaves corresponding to spatial frequency bands (similar to the usual notion of "turbulence" textures in graphics). For example, with three octaves we store three kinetic energy densities in each grid cell (i,j,k): $E^{(1)}_{ijk}$, $E^{(2)}_{ijk}$, and $E^{(3)}_{ijk}$, using E instead of the traditional k to avoid confusion with grid cell indices. The first, $E^{(1)}$, corresponds to the components with wavelengths approximately between Δx and $\frac{1}{2}\Delta x$, the $E^{(2)}$ value for wavelengths between $\frac{1}{2}\Delta x$ and $\frac{1}{4}\Delta x$, etc. This allows us to track the transfer of energy in spectrum as well as space—opening the door to handling the transition from laminar to turbulent flow, or the unsteady evolution and decay of unsustained turbulence. This still has attractive scalability: to increase the apparent resolution of a simulation by a factor of 2^k our memory use only increases by $O(n^3k)$. We cannot distinguish variations in turbulent energy at sub-grid resolution, but since the turbulent energy undergoes a diffusion process, sub-grid variations shouldn't be visually significant.

We also address the issue of temporal coherence in the turbulence field. While the energy density is enough to procedurally synthesize a plausible velocity field for a single frame, we want to reflect the correlation from one frame to the next. Inspired by flow noise [PN01] we add an additional scalar variable $\theta^{(b)}_{ijk}$ in each grid cell for each octave, an estimate of the average amount of rotation in that region of space up to the current time caused by the turbulent components, i.e. the time integral of the magnitude of vorticity at a fixed location in space. We use this to directly control flow noise when instantiating velocities.

4. Evolving Sub-Grid Turbulence

We break up the problem into plausibly evolving the kinetic energies $E^{(b)}_{ijk}$ and separately updating the θ angles.

4.1. Energy Evolution

Note that the large-scale flow on the grid transports with it small-scale turbulent components. Therefore, just as in the $k-\varepsilon$ model we begin with advecting the kinetic energy densities with the usual fluid variables in the simulation.

In addition, the enhanced mixing of turbulent flow is usually modeled as a nonlinear spatial diffusion term, spreading

turbulent energy across space. We make a crude simplification to constant-coefficient linear diffusion instead; from a scientific viewpoint this is probably unjustifiable, yet we argue it visually captures the qualitative effect of eddy viscosity while avoiding numerical complications caused by more accurate nonlinear models.

Finally, the nonlinear advection term in the Navier-Stokes momentum equation causes transfer of kinetic energy between different frequency bands. While this happens in both directions, it is widely modeled that the dominant effect in turbulence is the "spectral cascade" of energy from low frequencies to higher frequencies, with the majority of the energy transfer being between nearby wavelengths. We therefore include a simple transfer term from the kinetic energy in one octave to the next octave along, similar to Neyret's vorticity transfer. Again, the true energy transfer is a nonlinear process and our constant-coefficient linear simplification is scientifically invalid, but is useful as the simplest possible model that captures the qualitative visual effect of the spectral cascade.

We discretize this for a single time step on a grid as follows, with given coefficients $\alpha \geq 0$ and $\beta \geq 0$ controlling spatial diffusion and spectral cascade respectively:

- Advect the per-octave kinetic energies with the large-scale flow to get intermediate energies $\bar{E}^{(b)}_{ijk}$.
- Apply spatial diffusion and scale-to-scale transfer to get the energies at the next time step:

$$E^{(b)}_{ijk} = \bar{E}^{(b)}_{ijk} + \alpha\Delta t \begin{pmatrix} \bar{E}^{(b)}_{i+1,jk} + \bar{E}^{(b)}_{i-1,jk} \\ +\bar{E}^{(b)}_{ij+1,k} + \bar{E}^{(b)}_{ij-1,k} \\ +\bar{E}^{(b)}_{ijk+1} + \bar{E}^{(b)}_{ijk-1} \\ -6\bar{E}^{(b)}_{ijk} \end{pmatrix} + \beta\Delta t(\bar{E}^{(b-1)}_{ijk} - \bar{E}^{(b)}_{ijk}) \quad (1)$$

There is a time step restriction of $\Delta t < (6\alpha + \beta)^{-1}$, which in our examples was never particularly restrictive. Strictly speaking the α and β coefficients should take into account length scales and the energies themselves, but we choose to keep them as simple tunable constants. In our evolution equation the first octave $\bar{E}^{(1)}$ refers to a nonexistent $\bar{E}^{(0)}$: this ideally should be energy pulled from the large-scale simulation itself, but we instead simply seeded it with a spatial texture, e.g. introducing turbulent energy just in the source of a smoke plume. At the last octave, we optionally add another term $\gamma\Delta t\bar{E}^{(b)}$ with $0 \leq \gamma \leq \beta$ to partially cancel out the loss of energy to untracked octaves. See figure 2 for an example of the effect of both diffusion and spectral cascade.

We underscore that this evolution doesn't influence the large-scale motion at all, under the assumption that errors in the large-scale fluid simulation will dominate the effect of sub-grid turbulence. This decoupling has the attractive side-effect that turbulence evolution can be done as a post-process: once a suitable large-scale simulation has been found, different initial conditions and parameters for turbulence evolution can be tried out very efficiently.

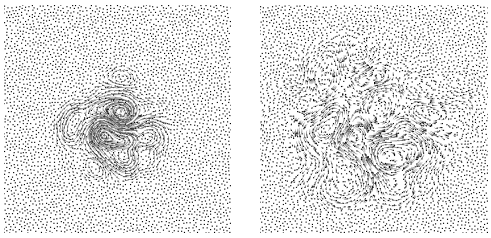


Figure 2: *Our turbulence model shows an initial disturbance in the lowest octave (left) diffusing in space and cascading energy to higher octaves later in time (right).*

We ran examples of the new turbulence model coupled with a FLIP simulation of smoke [ZB05], with the standard addition of heat and buoyancy forces to the fluid solver. We advected the turbulent kinetic energy variables with the FLIP particles, transferring them to the grid, applying diffusion and spectral cascade on the grid, and transferring the result back to the particles in the usual FLIP way.

4.2. Net Rotation Evolution

The net rotation $\theta^{(b)}_{ijk}$ is an estimate of the time integral of the magnitude of vorticity stemming from the b'th octave of turbulence in grid cell (i, j, k). This is *not* a quantity to be advected: it's an Eulerian history variable. We also emphasize this is just a scalar, unlike the vector-valued vorticity, since we are using this to estimate average rotation over a region of space (containing many differently-oriented vorticities), analagous to our use of scalar kinetic energy density rather than mean velocity.

At every time step of the turbulence evolution simulation, we increment $\theta^{(b)}_{ijk}$ by an estimate of the average magnitude of the vorticities in the b'th octave, which we assume are proportional to the mean speed (available from the kinetic energy density) divided by the wave length:

$$\|\omega\|^{(b)}_{ijk} \sim \frac{\sqrt{2E^{(b)}_{ijk}/\rho}}{2^{-b}\Delta x} \quad (2)$$

This is dimensionally correct and consistent with the observation that turbulence typically remains isotropic. Introducing a constant of proportionality δ we use the following update:

$$\theta^{(b)}_{ijk} \leftarrow \theta^{(b)}_{ijk} + \delta\Delta t \frac{\sqrt{2E^{(b)}_{ijk}/\rho}}{2^{-b}\Delta x} \quad (3)$$

For the full effect we should also introduce a $\theta^{(0)}$ variable that is updated by the magnitude of vorticity in the large-scale simulation, but we have not yet experimented with this.

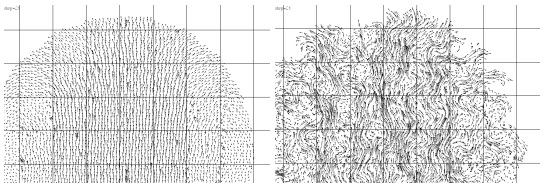


Figure 3: *On the left is the large-scale velocity field interpolated from the displayed grid (unrealistically smooth for illustration purposes). On the right we add sub-grid turbulent scales to get the total velocity field.*

5. Instantiating Velocity Fields

Once we have completed a large-scale fluid simulation and then turbulence evolution, we generate a plausible total velocity field by interpolating from the large-scale velocity grid and adding to it the curl of a vector potential derived from the turbulence quantities, as in the curl-noise approach of Kniss and Hart [KH04] and Bridson et al. [BHN07]: see figure 3.

Perlin and Neyret's flow noise [PN01] was originally designed to provide fluid-like textures which animate in time with a pseudo-advection quality (unlike regular time-animated noise which smoothly changes without coherent flow structure): the gradient vectors underling Perlin noise are rotated in time as if by fluid vorticity. Neyret further used this for deforming other textures in his advected texture work [Ney03], and Bridson et al. [BHN07] suggested it is well suited for use in curl-noise to generate velocity fields featuring vortices (corresponding to peaks in the noise functions) which naturally swirl around and interact with other vortices rather than simply smoothly appearing and disappearing. We therefore adopt flow noise to build the vector potential $\vec{\psi}$ for the turbulent velocity contribution; however, since vorticities in turbulence should be uniformly distributed due to isotropy, we modify Neyret's scheme which only used a single vorticity per grid cell per octave.

We begin with an auxiliary array of 128 pairs of uniformly randomly selected orthogonal unit three-dimensional vectors, $\hat{p}$ and $\hat{q}$, which will be used to parameterize a plane of rotation at each point in the lattice Instead of Perlin's hash function to map lattice points to this auxiliary array, which induces severe axis-aligned artifacts in frequency space (see figure 8(a) and (b) from Cook and DeRose's work [CD05]) we use the following:

$$\text{hash}(i,j,k) = H(i \text{ xor } H(j \text{ xor } H(k))) \bmod 128 \qquad (4)$$

where $H(s)$ provides a good quality pseudo-random permutation of the 32-bit integers:

- $H(s)$:
- $s \leftarrow (s \text{ xor } 2747636419) \cdot 2654435769 \bmod 2^{32}$
- $s \leftarrow (s \text{ xor } \lfloor s/2^{16} \rfloor) \cdot 2654435769 \bmod 2^{32}$
- $s \leftarrow (s \text{ xor } \lfloor s/2^{16} \rfloor) \cdot 2654435769 \bmod 2^{32}$
- return s

This has in fact proven useful as a reasonably efficient stateless pseudo-random number generator in other work.

To evaluate a given octave of this noise at a point, we look up the rotated gradients at the corners of the lattice cell containing the point. This lattice, for octave b, has a spacing of $2^{-b+1}\Delta x$ compared to the Δx spacing of the simulation grid. At each lattice point we use the hash to look up a pair of orthogonal unit vectors, $\hat{p}$ and $\hat{q}$, trilinearly interpolate the net rotation $\hat{\theta}$ in this octave from the simulation, and finally form a rotated gradient vector as:

$$\vec{g} = \cos\hat{\theta}\hat{p} + \sin\hat{\theta}\hat{q} \qquad (5)$$

One of the most expensive parts of this calculation is the trigonometic function evaluation, but after all the modeling errors in the system it's not crucial that these be accurate; we substituted the following cubic spline

$$\sin\theta \approx 12\sqrt{3}t\left(t-\frac{1}{2}\right)(t-1) \qquad (6)$$

where t is the fractional part of $\theta/(2\pi)$, and similarly approximated $\cos(\theta) = \sin(\theta + \frac{1}{2}\pi)$. We found this was sufficiently accurate and much faster. (Note that this spline has zeros at the correct roots of sin and attains a max and min of ± 1 as expected.) Finally, we evaluate three independent noise functions to get a vector noise value $\vec{N}^{(b)}(\vec{x},t)$.

We also interpolate the kinetic energy density in an octave at any point in space from the values on the simulation grid, using quadratic B-splines, to estimate an appropriate amplitude for modulating the noise:

$$A^{(b)}(\vec{x},t) = C2^{-b}\Delta x\sqrt{2E^{(b)}(\vec{x},t)/\rho} \qquad (7)$$

The user-defined constant C should be approximately 1, so that the actual kinetic energy density of the velocity field below will match up to the stored $E^{(b)}$. Adding up the octaves gives the vector potential ψ:

$$\vec{\psi}(\vec{x},t) = \sum_b A^{(b)}(\vec{x},t)\vec{N}^{(b)}(\vec{x},t) \qquad (8)$$

Further modifications to respect solid boundaries can be made per Bridson et al. [BHN07]. The turbulent part of the total velocity field is the curl of $\vec{\psi}$.

For rendering we typically run many smoke marker particles through the total velocity field (seeded in the appropriate places for the given simulation), evaluating velocity wherever and whenever needed. With several octaves of noise, all the interpolation and gradient evaluations can be rather more expensive than simple interpolation of velocity from a grid, even with optimizations such as our trigonometic approximation. However, we do highlight that it is embarrasingly parallel—each marker particle can be advected without reference to any of the others—and the amount of data involved is fairly small relative to the apparent resolution of the turbulent velocity field. Further optimization, however, may be possible by using more memory: evaluating at least some of

the octaves on higher resolution grids once and then interpolating from there, or evaluating on smaller tiles that are cached for reuse by multiple particles.

6. Reducing Angular Dissipation in Simulation

We have so far looked at layering in sub-grid turbulence; it is of course also imperative that as much simulation detail as possible is retained in the large-scale simulation. We propose a simple multistep predictor to reduce the afore mentioned loss of angular momentum caused by the first order time splitting of advection.

To motivate our method, we point out that the incompressibility constraint can be arrived at as the infinite limit of the second viscosity coefficient λ:

$$\frac{d\vec{u}}{dt} = \frac{\lambda}{\rho}\nabla(\nabla\cdot\vec{u}) \tag{9}$$

(Note this is not the physically correct limit in which compressible flow becomes asymptotically incompressible, but rather a useful thought experiment which avoids the need of considering variables other than velocity and equations other than momentum.) As $\lambda\to\infty$, any divergent motions are damped out to nothing, giving back incompressible flow; other motions, such as shearing, are unaffected.

For a stiff problem such as equation 9, an implicit integrator with stiff decay is recommended. For example, if Backwards Euler is used for the stiff viscosity term, and we then take the limit of the solution as $\lambda\to\infty$, we find we are back to the usual pressure projection splitting method: $\vec{u}_{n+1}$ is the divergence-free part of the advected $\vec{u}_n$.

To get higher accuracy we consider better implicit methods which still posess stiff decay, such as BDF2. This is a multistep method which, after again assuming the advection term is handled separately, would discretize equation 9 as:

$$\vec{u}_{n+1} = \frac{4}{3}\vec{u}_n - \frac{1}{3}\vec{u}_{n-1} + \frac{2}{3}\Delta t\frac{\lambda}{\rho}\nabla(\nabla\cdot\vec{u}_{n+1}) \tag{10}$$

If we solve this, then take the limit as $\lambda\to\infty$, we get pressure projection at time $n+1$ of $\frac{4}{3}\vec{u}_n - \frac{1}{3}\vec{u}_{n-1}$, which can be seen as a simple predictor for the new velocity. (We emphasize it is the final time $n+1$ velocity that is made divergence-free: there is no divergence error in the velocity field in which we advect particles.)

This analysis isn't fully worked out, but our preliminary experiments with FLIP, where we transfer the predicted new particle velocities, based on their current and past values, to the grid for pressure projection instead of the current velocity, are very promising: the method remains apparently unconditionally stable, with significantly improved conservation of angular momentum yet without introducing noise. Visually flows remain a lot more lively and areas of rotation are damped much more slowly, without blowing up.

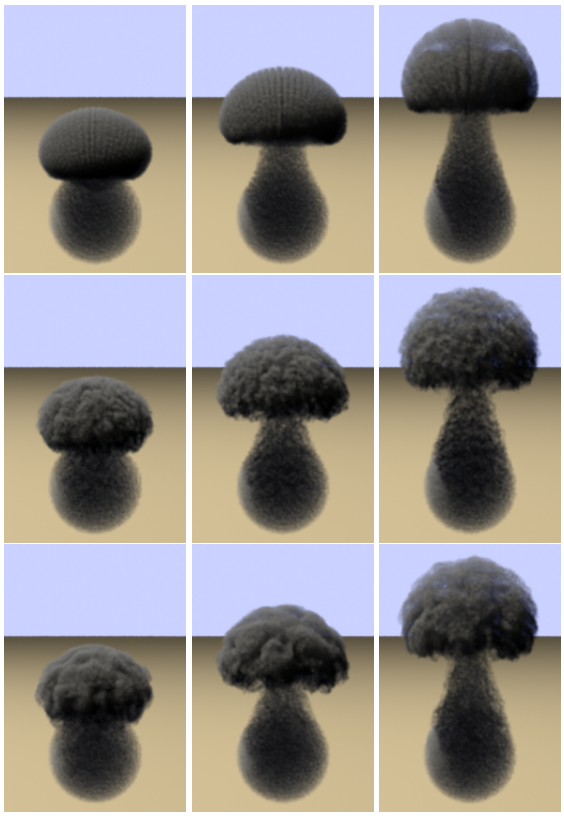


Figure 4: *Top row: large-scale motion only. Middle row: turbulence added, but without evolution. Bottom row: our full turbulence evolution model.*

7. Results

We ran several demonstrations of the sub-grid turbulence model in 3D. The time added to the simulation for tracking the extra turbulence quantities was negligible, giving roughly two seconds per frame for a $30\times120\times30$ grid on a 2.2Ghz Core Duo. Running the marker particles through the total velocity field with three octaves of noise scaled linearly with particle count, at about .3 seconds per frame for 1000 particles; this became a bottleneck for higher quality renders with hundreds of thousands of particles, but as we noted earlier should allow for trivial parallelization not to mention other optimization possibilities.

Figure 1 shows frames from a simulation of a smoke plume generated by a continuous source of hot smoke, with the full turbulence model in place. Marker particles were seeded in each frame, then rendered with self-shadowing.

Figure 4 demonstrates the effect of layering the turbulence model on top of existing large-scale motion, with or without the evolution (diffusion and spectral cascade). Without turbulence the motion is flat and boring, at least in these early frames. With turbulence but without evolution, we

were forced to include energy in all octaves right from the start; there thus appears to be too much fine-scale detail initially, but not enough mid-scale turbulent mixing. The full evolution model instead shows the spectral cascade, with the smallest-scale vortices only appearing naturally as energy reaches them, and the spatial diffusion lets the turbulent mixing extend further into the flow breaking up symmetries better.

8. Conclusions

We presented both a new sub-grid turbulence evolution model and a new predictor step for fluid simulation, with the goal of getting closer to high quality smoke simulation. At present our model remains too simplistic for scientific validity, but we argue it captures much of the qualitative look of real turbulence.

There are many directions for future work, beyond simply improving the accuracy of our model: in particular, we highlight the question of how to automatically pull energy from the large-scale simulation into the first turbulent octave. Ideally a solution to this would not perturb stable laminar regions of flow, but naturally cause a transition to turbulence when an instability is detected.

Acknowledgements

This work was supported in part by a grant from the Natural Sciences and Engineering Research Council of Canada, and by a scholarship from BC Innovation Council and Precarn Incorporated.

References

[AN05] ANGELIDIS A., NEYRET F.: Simulation of smoke based on vortex filament primitives. In *Proc. ACM SIGGRAPH/Eurographics Symp. Comp. Anim.* (2005).

[BHN07] BRIDSON R., HOURIHAN J., NORDENSTAM M.: Curl-noise for procedural fluid flow. *ACM Trans. Graph. (Proc. SIGGRAPH) 26*, 3 (2007).

[BMF07] BRIDSON R., MÜLLER-FISCHER M.: Fluid simulation, 2007. ACM SIGGRAPH 2007 courses.

[CD05] COOK R. L., DEROSE T.: Wavelet noise. *ACM Trans. Graph. (Proc. SIGGRAPH) 24*, 3 (2005), 803–811.

[FSJ01] FEDKIW R., STAM J., JENSEN H. W.: Visual simulation of smoke. In *Proc. SIGGRAPH* (2001), pp. 15–22.

[Gam95] GAMITO M. N.: Two dimensional Simulation of Gaseous Phenomena Using Vortex Particles. In *Proc. of the 6th Eurographics Workshop on Comput. Anim. and Sim.* (1995), Springer-Verlag, pp. 3–15.

[HN67] HARLOW F. H., NAKAYAMA P. I.: Turbulence transport equations. *Phys. Fluids 10*, 11 (1967), 2323–2332.

[KH04] KNISS J., HART D.: Volume effects: modeling smoke, fire, and clouds, 2004. Section from ACM SIGGRAPH 2004 courses, *Real-Time Volume Graphics*, http://www.cs.unm.edu/ jmk/sig04_modeling.ppt.

[KLLR05] KIM B., LIU Y., LLAMA I., ROSSIGNAC J.: FlowFixer: using BFECC for fluid simulation. In *Proc. Eurographics Workshop on Natural Phenomena* (2005).

[KTJG08] KIM T., THÜREY N., JAMES D., GROSS M.: Wavelet turbulence for fluid simulation. *ACM Trans. Graph. (Proc. SIGGRAPH)* (2008).

[Ney03] NEYRET F.: Advected textures. In *Proc. ACM SIGGRAPH/Eurographics Symp. Comp. Anim.* (2003), pp. 147–153.

[PK05] PARK S. I., KIM M. J.: Vortex fluid for gaseous phenomena. In *Proc. ACM SIGGRAPH/Eurographics Symp. Comp. Anim.* (2005).

[PN01] PERLIN K., NEYRET F.: Flow noise. In *ACM SIGGRAPH Technical Sketches and Applications* (2001), p. 187. http://www-evasion.imag.fr/Publications/2001/PN01/.

[Pop05] POPE S. B.: *Turbulent Flows*. Cambridge University Press, 2005.

[RNGF03] RASMUSSEN N., NGUYEN D., GEIGER W., FEDKIW R.: Smoke simulation for large scale phenomena. *ACM Trans. Graph. (Proc. SIGGRAPH) 22* (2003), 703–707.

[SF92] SHINYA M., FOURNIER A.: Stochastic motion: Motion under the influence of wind. In *Proc. Eurographics* (1992), pp. 119–128.

[SF93] STAM J., FIUME E.: Turbulent wind fields for gaseous phenomena. In *Proc. ACM SIGGRAPH* (1993), pp. 369–376.

[SRF05] SELLE A., RASMUSSEN N., FEDKIW R.: A vortex particle method for smoke, water and explosions. *ACM Trans. Graph. (Proc. SIGGRAPH)* (2005), 910–914.

[Sta99] STAM J.: Stable fluids. In *Proc. SIGGRAPH* (1999), pp. 121–128.

[TL72] TENNEKES H., LUMLEY J. L.: *A first course in turbulence*. MIT Press, 1972.

[YUM86] YAEGER L., UPSON C., MYERS R.: Combining physical and visual simulation—creation of the planet jupiter for the film 2010. In *Proc. ACM SIGGRAPH* (1986), pp. 85–93.

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