Retiming of Fluid Simulations for VFX: Distributed Non-Linear Fluid Retiming by Sparse Bi-Directional Advection-Diffusion

Ken Museth
Weta Digital

Figure 1: Our fluid retiming technique applied to both smoke (left image) and fire (right image) simulations. Smoke (right): Input generated by forward fluid simulation at 24 fps. Smoke (mid): Output generated by 5X retiming to 120 fps. Smoke (left): Output generated by 10X retiming to 240 fps. Fire (left): Input generated by forward fluid simulations at 24 fps. Fire (right): Output generated by 4X retiming to 96 fps.

ABSTRACT

We present a novel approach to retiming of fluid simulations, which is a common yet challenging practice in visual effects productions. Unlike traditional techniques that are limited to dense simulations and only account for bulk motion by the fluid velocities, our approach also works on sparse simulations and attempts to account for two (vs one) of the fundamental processes governing fluid dynamics, namely the effect of hyperbolic advection and parabolic diffusion, be it physical or numerical. This allows for smoother transitions between the existing and newly generated simulation volumes, thereby preserving the overall look of a retimed fluid animation, while significantly outperforming both forward simulations, which tend to change the look, and guided inverse simulations, which are known to be computationally expensive.

CCS CONCEPTS

• **Computing methodologies** → **Physical simulation**;

KEYWORDS

Fluid simulation, re-timing, advection, inverse diffusion, VFX, VDB

ACM Reference Format:

Ken Museth. 2019. Retiming of Fluid Simulations for VFX: Distributed Non-Linear Fluid Retiming by Sparse Bi-Directional Advection-Diffusion. In *Proceedings of SIGGRAPH '19 Talks.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3306307.3328163

SIGGRAPH '19 Talks, July 28 - August 01, 2019, Los Angeles, CA, USA

ACM ISBN 978-1-4503-6317-4/19/07.
https://doi.org/10.1145/3306307.3328163

MOTIVATION

Fluid simulations like fire, smoke, clouds, explosions and water are ubiquitous in VFX productions. It is well known that these simulations tend to be computationally expensive, but equally problematic is the fact that they are also notoriously difficult to art-direct or sculpt. This is partly a consequence of the fact that the governing Navier-Stokes equations of fluid mechanics generate non-linear effects at multiple scales (think cascade of energy in turbulence) and are exceedingly difficult to solve accurately. In practice this implies that, at least in computer graphics, these fluid equations are rarely solved to actual numerical convergence and hence even small changes in temporal or spatial discretizations can lead to significant changes of the final look. This problem is further exacerbated when artists run fluid simulations at low resolutions (in space and time) in an effort to speed up turnarounds for their look development. This often leads to the following dilemma where an artist has finally hit a desired "look and feel" of a fluid simulation, only to be asked by the vfx supervisor to please slow it down, either to create the illusion of a larger scale, or simply to change the pace of the shot. Due to the reasons outline above, the solution is rarely to re-simulate at the new desired time-scale, since that is certain to change the look. This essentially rules out full forward simulation, so the natural alternative would seem to be an inverse simulation. However, target-drive fluid simulations are typically slower than even forward simulations and hence deemed impractical for VFX.

The preferred solution to this common dilemma is dubbed "retiming", which conceptually amounts to interpolation of the cached primary simulation volumes. This is also analogous to what animators call "inbetweening". The computational process by which the interpolated volumes are generated typically involves a simplified secondary "fluid-like" simulation that performs forward, backward

or bi-directional advection, based on the existing cached fluid fields like velocities, densities, etc. Forward and backward advections are relatively straightforward for dense velocity fields, but are often accompanied by distinct non-smooth changes in motion and intensity when a key-frame, or more precisely key-volume, is crossed. This can be mitigated somewhat by blending bi-directional advection from the two neighboring key-volumes, but unless they are temporally close to each other, the overlaps can be off, which produces artifacts like popping or pulsing. Most VFX studios, and even some commercial DCCs, have their own implementation of this idea, but at Weta Digital these existing options were found to be inadequate. This of course motivated the current work, which consists of the following contributions: 1) sparse bi-directional advection, 2) sparse bi-directional diffusion, 3) distributed computation, and 4) non-linear retiming – each of which will be detailed below.

1 SPARSE BI-DIRECTIONAL ADVECTION

Bi-directional advection, i.e. bulk motion forward and backward in time relative to the flow velocities cached in key volumes, is usually the main computational technique found in existing implementations of retiming. If the key volumes, or more precisely the flow fields, are reasonably similar this forward and backward advection aligns spatially and can be mixed or blended without introducing severe artifacts. However, if the key volumes are not sampled closely enough this approach can introduce blurring or temporal popping, a problem that we shall revisit in the next section. For now, a bigger issue is that most implementations of fluid advection are limited to dense grids. This stems from the fact that they are based on semi-Lagrangian advection schemes. In essence, they work by tracing information back in time (along characteristics of the hyperbolic PDE) and then resample quantities from the underlying Eulerian grid. While unconditionally stable, this approach assumes that velocity values are already available at grid points to which future quantities are transported, an assumption that only holds true for dense grids. However, with the advance of compact data structures like VDB[Museth 2013], it is now common to have fluid caches with sparse velocity fields. To address this we have implemented a sparse advection scheme that works in two stages. Initially, velocity-self-advection is performed using a hybrid approach inspired by FLIP schemes that solve for advection by means of a mixed Lagrangian and Eulerian representation. However, we have deviated somewhat from the classic FLIP technique by implementing higher-order polynomial particle-to-grid interpolation, which does not require high particle densities. It works as follows: Grid points in the cached sparse velocity volumes are self-advected (forward or backward) using RK schemes. Then their new coordinate positions are used to build a VDB point acceleration structure, PointIndexGrid, which in turn is used to construct a new voxel grid (after performing morphological operations to close holes due to particle divergence). Velocity values are then sampled onto the active voxels in the new grid by means of Moving Least Squares (MLS) interpolation,which uses the PointIndexGrid for fast point-lookup. This defines a new self-advected sparse velocity field. Finally, all remaining sparse fluid fields, such as density, temperature and fuel, are passively advected by regular semi-Lagrangian advection (after their sparse grid topology has been advected similarly to the space velocity grid).

2 SPARSE BI-DIRECTIONAL DIFFUSION

While advection certainly accounts for one of the more prominent characteristics of fluid dynamics, namely bulk motion or transport of physical quantities, it is certainly not the only contributor. Most notably, virtually all fluid simulations are also affected by some degree of diffusion, either through actual physical diffusion or as numerical viscosity from discretization errors. As mentioned above, artificial diffusion, in the form of blurring, can also be introduced when blending forward and backward volumes that are misaligned, e.g due to the underlying assumption of constant velocities from key volumes. In all these cases the net effect is a blurring or smearing out of details or gradients. The problem is, however, that inverse diffusion, which would be required to reverse this effect during backwards propagation, is numerically unstable (think inverse heat equation). Our solution draws inspiration from image processing, where diffusion is easily shown to be equivalent to blurring or smoothing by convolution. It then follows that inverse diffusion, or at least a small amount of it, can be approximated as inverse blurring, which of course amounts to sharpening. These ideas translate trivially from 2D to 3D as well as from dense to sparse grids. Specifically we perform forward and time-reversed, i.e. inverse, diffusion by means of proven techniques from image processing, namely 3D convolution by respectively median-value filters and unsharp masking. Mean-value filters preserves boundaries, and unsharp masking, unlike actual inverse diffusion, is unconditionally stable (though it might still amplify noise if applied too aggressively).

3 DISTRIBUTED COMPUTATIONS

Since the simulations outlined above are frame independent they can trivially be multi-threaded and even distributed on the "render wall". Thus, we can perform retimings in a fraction, typically less than a minute, of the time that it took to generate the input caches.

4 NON-LINEAR RETIMING

As explained earlier, the crux of fluid retiming is to remap time and create the missing simulation frames. However, this remapping of time can be either linear, as in changing the fps, or non-linear, e.g. speed up and then slow down an explosion. Our implementation supports both, essentially allowing for arbitrary retiming through the use of artist-defined curves that map input time to output time.

5 CONCLUSION AND LIMITATIONS

We have combined all of the ideas outlined above to form a system that is capable of generating in-between volumes, which more closely match existing key-volumes, and also works on sparse fluid caches. However, our solution is certainly not a "silver bullet" in the sense that, similarly to all other attempts at retiming, it will of course fail for sufficiently coarse temporal sampling of the input volumes. This is a simple consequence of the fact that retiming is fundamentally an ill-posed inverse problem and like any interpolation method has lower bounds on the input sampling rates.

REFERENCES

Ken Museth. 2013. VDB: High-resolution Sparse Volumes with Dynamic Topology. *ACM Trans. Graph.* 32, 3, Article 27 (July 2013), 22 pages. https://doi.org/10.1145/2487228.2487235