Real-Time Grid-Based Fluid Simulation

Abstract

This report shall provide an insight into the implementation details of a real-time fluid simulation. More specifically, the grid-based approach proposed by Jos Stam (2003), including a critical analysis of the methodology. An alternative particle-based approach to the simulation will also be very briefly discussed, as well as a performance comparison of a CPU vs GPU implementation.

Introduction

The realistic motion of fluid can be described by a set of partial differential equations, known as Navier-Stokes equations (Figure 1). In simple terms, these equations account for all momentum exchange possibilities within a fluid, such as: acceleration, convection, gravity and viscosity (Foster and Metaxas 1996). These equations lay the foundations of simulating natural phenomena, such as flowing rivers and rising smoke, seen commonly in video games (Stam 2003; Wikipedia 2021).

$$\frac{\partial \mathbf{u}}{\partial t} = -(\mathbf{u} \cdot \nabla)\mathbf{u} + \nu \nabla^2 \mathbf{u} + \mathbf{f}$$

$$\frac{\partial \rho}{\partial t} = -(\mathbf{u} \cdot \nabla)\rho + \kappa \nabla^2 \rho + S$$

Figure 1: Navier-Stokes equations: Top - Velocity in a compact vector notation. Bottom - Density moving through a velocity field (Stam 2003, p. 2).

Implementation

A grid-based fluid simulation is represented by a grid of cells, whereby each cell contains information on the fluid's properties within its bounds, such as its: velocity, density, and temperature etc. All of which properties are a product of the cell they belong to and its direct neighbours, which are calculated via three major stages of the simulation: Diffusion, Advection and Projection.

Firstly, it's important to identify that Stam's (2003) method models fluid as if you were observing from within, as opposed to without. Try to imagine the implementation akin to air swirling around the inside of a room, instead of water sloshing inside a cup (Cowboy Programming 2008). What the simulation is actually visualising is the suspension of a substance within the fluid, like ink in water or smoke in the air. The measurement of such substance refers to the fluid's density.

Diffusion

Diffusion is the process that describes the change in density among the grid cells (Figure 2). Stam's (2003) implementation used incompressible (or mass conserving) fluid, meaning the space it occupied could not be changed. This meant that the total amount of fluid leaving a cell, had to be equal to the total amount of fluid entering a cell. Stam (2003) implemented his solution using Gauss-

Seidel relaxation, describing it as “the simplest iterative solver which works well in practice” (p. 6). The result being each cells density value gradually becoming the average of those surrounding it.

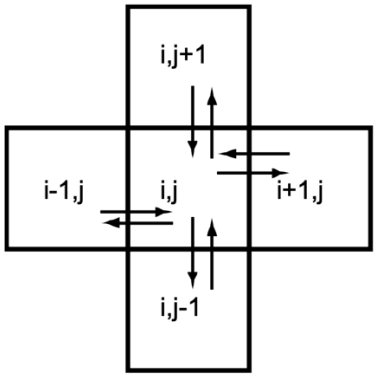

Figure 2: Visualisation of diffusion (Stam 2003, p. 5).

Advection

Advection is the process of moving the fluids data across the grid, through its velocity field. This step is required due to the grid cells being in a fixed position, meaning the data needs to be transferred across to other cells. This is unlike a particle-based implementation, where each particle stores its own data and can move freely (Braley and Sandu 2009). Rather than intuitively tracing each cell forwards in time, Stam (2003) used “linear backtracing” (p. 7) which instead traced the cells backwards in time, by inverting their velocities. These previous values would then be used to represent the values forward in time, with greater accuracy and less computation (Figure 3).

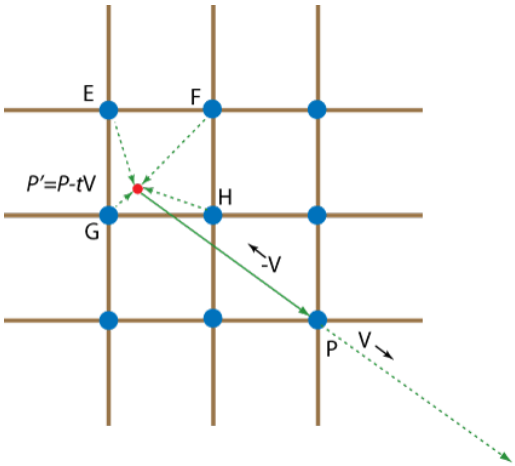

Figure 3: Linear Backtracing: Value P’ is a product of E, F, G, H and is used to calculate point P (Cowboy Programming 2008).

Projection

Ensuring the fluid is incompressible is a prerequiste for both methods of Diffusion and Advection mentioned above, thus handled by the Projection step. Since fluid cannot be created nor destroyed, once the simulation has begun, divergence within the velocity field must be filtered out. This can typically be solved by using Helmholtz decomposition (Inspecto 2021; Wikipedia 2021) although Stam (2003) mentions an approach using Hodge Decomposition and a Poisson equation. Essentially,

a divergence-free velocity field is the result of the original velocity field, subtract the curl-free field (Figure 4).

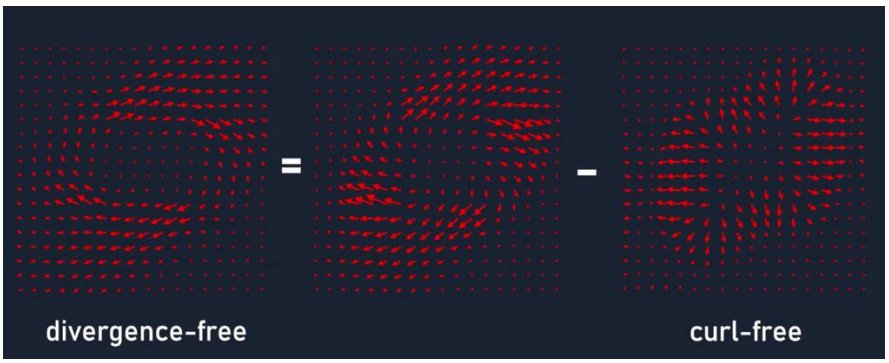

Figure 4: Visualisation of divergence-free calculation (Inspecto 2021).

Finally, to contain the fluid within a set of boundaries, each boundary cell must calculate an exact negative velocity of the incoming velocity – directing the fluid to flow in the opposite direction. This can be better visualised by toggling on implemented debug options that highlight the fluids velocity and each cells boundary. Additionally, a user may adjust the resolution of the simulation, causing the grid dimensions to change by a scale factor of two.

Critical Analysis

Despite grid-based simulations typically being very accurate when compared to their particle-based counterpart, they also tend to be a lot slower (Braley and Sandu 2009). Unlike Stam's (2003) CPU implementation, academics have seen a great deal of success with using utilising the GPU and its fragment shading pipeline to maximize simulation performance (Liu et al 2004). GPU implementations take advantage of the huge parallel processing potential a card can offer, leading to the performance increases of up to almost 15 times (Figure 5).

Grid scale	Average GPU time (ms)	Average CPU time (ms)	Speedup
64*64	0.76	3.15	4.1X
128*128	1.15	13.07	11.4X
256*256	30.00	332.30	11.1X
512*512	49.00	719.19	14.7X
1024*1024	201.00	2819.48	14.0X

Figure 5: GPU vs CPU performance (Liu et al 2004, p. 8).

GPU Gems (2004) features an article explaining how their grid-based approach is implemented by using textures as an alternative data structure to store and better process the fluid data, instead of using arrays (Figure 6).

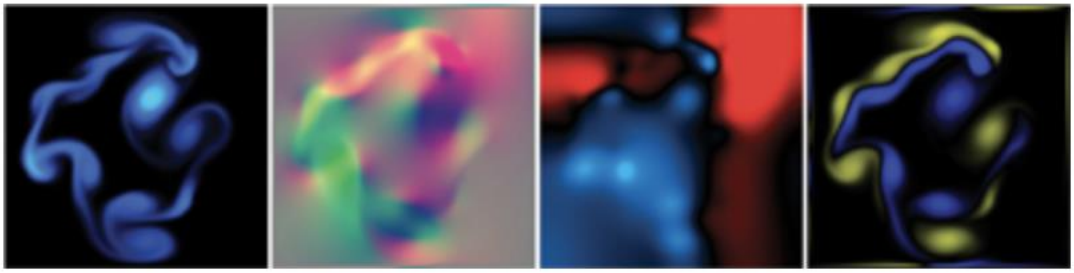

Figure 6: Fluid data stored as separate textures (GPU Gems 2004).

Conclusion

Grid-based fluid simulations are a great proof of concept when demonstrating real-world fluid mechanics but can be held back in both detail and performance if not implemented by fully utilising the GPU. Their application, especially Stam's (2003) implementation, is only relevant within two-dimensional space. However, as Ash (2005) demonstrates, the concepts that drive a 2D simulation can be very easily upscaled to 3D and thus be used more appropriately within games and films.

References

Ash, M., 2005. *Simulation and Visualization of a 3D Fluid* [online]. Available from: https://www.mikeash.com/thesis/thesis-en.pdf [Accessed 18 March 2021].

Braley, C. and Sandu, A., 2009. *Fluid Simulation For Computer Graphics: A Tutorial in Grid Based and Particle Based Methods* [online]. Available from: https://cg.informatik.uni-freiburg.de/intern/seminar/gridFluids_fluid-EulerParticle.pdf [Accessed 18 March 2021].

Bridson, R., 2015. *Fluid Simulation for Computer Graphics*. 2nd Edition. CRC Press.

Cowboy Programming., 2008. *Practical Fluid Mechanics* [online]. Available from: http://cowboyprogramming.com/2008/04/01/practical-fluid-mechanics/ [Accessed 18 March 2021].

Foster, N. and Metaxas, D., 1996. Realistic Animation of Liquids. *Graphical Models and Image Processing* [online], 58 (5), 471-483.

Inspecto, 2020. *But HOW Do Fluid Simulations Work?* [video, online]. YouTube. Available from: https://youtu.be/qsYE1wMEMPA [Accessed 18 March 2021].

NVIDIA Corp., 2004. *GPU Gems* [online]. NVIDIA Corp.

Stam, J., 2003. *Real-Time Fluid Dynamics for Games* [online]. Available from: https://www.autodesk.com/research/publications/real-time-fluid-dynamics [Accessed 18 March 2021].

Wikipedia., 2021. *Helmholtz decomposition* [online]. Available from: https://en.wikipedia.org/wiki/Helmholtz_decomposition [Accessed 18 March 2021].

Wikipedia., 2021. *Navier-Stokes equations* [online]. Available from: https://en.wikipedia.org/wiki/Navier%E2%80%93Stokes_equations [Accessed 18 March 2021].

Wu, E., Liu, Y. and Liu, X., 2004. An improved study of real-time fluid simulation on GPU. *Computer Animation and Virtual Worlds.* 15 (34), 139-146.