Kora: A Physics-Based Fire Pipeline and Toolset

Alexey Stomakhin
Weta FX
Mililani, HI, USA
st.alexey@gmail.com

John Edholm
Weta FX
Vadstena, Sweden
jedholm@wetafx.co.nz

Murali Ramachari
Weta FX
Wellington, New Zealand
muralir@wetafx.co.nz

Aleksandr Isakov
Weta FX
Wellington, New Zealand
aisakov@wetafx.co.nz

Zahra Forootaninia
Weta FX
Wellington, New Zealand
zahraf@wetafx.co.nz

Marcus Schoo
Weta FX
Wellington, New Zealand
mschoo@wetafx.co.nz

Nicholas Illingworth
Weta FX
Wellington, New Zealand
nillingworth@wetafx.co.nz

Joe Letteri
Weta FX
Wellington, New Zealand
letteri@wetafx.co.nz

Figure 1: Fire shots from the climactic sequence of *Avatar: Fire and Ash* completed with Kora. ©2025 20th Century Studios / Lightstorm Entertainment. All Rights Reserved.

Abstract

We present *Kora*[1], a fire system for visual effects production. It consists of (i) a weakly compressible, sparse, spatially adaptive, MPI-distributed physics-based combustion solver, featuring a novel *energy cascade turbulence* model and support for *adiabatic cooling*; (ii) a Houdini-centric abstraction layer that exposes the advanced combustion routines through intuitive, artist-friendly interfaces. Key components include customizable premixed fuel sourcing, liquid-gas coupling and vaporization, control and art-direction utilities, as well as physics-based shading and render-graph integration. We describe the architecture of Kora, its solver foundations, and how we put it into practice in *Avatar: Fire and Ash*. We were honored to be awarded the Visual Effects Society (VES) 2026 Emerging Technology Award for this work.

CCS Concepts

• **Computing methodologies** → **Physical simulation**; *Simulation types and techniques.*

[1]Kora is te reo Māori for spark.

DigiPro '26, Los Angeles, CA, USA

ACM ISBN 979-8-4007-2777-1/26/07
https://doi.org/10.1145/3819990.3820026

Keywords

fire, explosion, combustion, flamethrower, simulation, pipeline

ACM Reference Format:
Alexey Stomakhin, John Edholm, Murali Ramachari, Aleksandr Isakov, Zahra Forootaninia, Marcus Schoo, Nicholas Illingworth, and Joe Letteri. 2026. Kora: A Physics-Based Fire Pipeline and Toolset. In *The Digital Production Symposium (DigiPro '26), July 18, 2026, Los Angeles, CA, USA.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3819990.3820026

1 Introduction

The 1000+ ambitious fire shots in *Avatar: Fire and Ash*, ranging from flaming arrows and flamethrowers to massive explosions and fire tornadoes (Figure 1), necessitated a system that would deliver believable results on an unprecedented scale. To achieve the required level of visual fidelity and consistency we developed a physics-based combustion solver incorporating accurate chemistry, thermodynamics, and thin flame models. On-set references for torches, flame bars, and burning tents guided the work across the Simulation, Look Development, and Rendering departments. Using chemical properties of real-world fuels and tuning pre-mixing with oxygen enabled us to capture compelling effects, such as oxygen starvation and flickering thin flames, leading to remarkable quality in matching real-world references.

To expand the solver adoption beyond experts in chemistry and compressible fluids, enable intuitive authoring within a familiar Houdini setting, and facilitate rapid onboarding, we further designed an artist-friendly pipeline and toolset built around our combustion solver and renderer, see Figure 2.

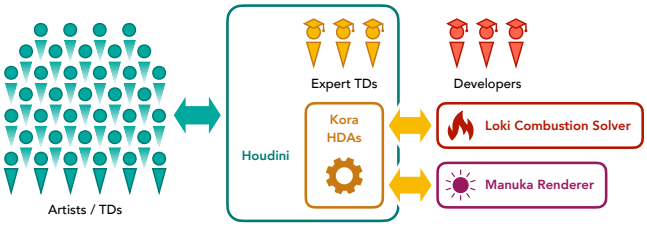

Figure 2: Kora overview. A layer of curated HDAs, designed by developers and expert TDs, abstracts away the complexities of our physics-based combustion solver and renderer, making them readily accessible to artists. ©2026 Wētā FX Ltd.

We begin by going over the related works in §2. In §3 we provide motivation for the development of our Kora fire system and list our contributions. §4 does a deep dive into the solver implementation. §5 is devoted to the pipeline and toolset architecture. Finally, in §6 we describe some of our workflows and production examples from *Avatar: Fire and Ash*, and finish with lessons and limitations in §7.

2 Previous work

Up until recently, many of the combustion solvers used in VFX—including commercial tools (e.g., SideFX Houdini) and proprietary implementations (e.g., [Aguilera and Johansson 2019])—relied primarily on phenomenological or heuristic models. These systems emphasized ease of use via numerous decoupled controls (e.g., buoyancy and expansion, which in reality are tightly coupled physically) and ad hoc modifiers such as dissipation, turbulence injection, and shredding forces [Maury et al. 2010]. Physics-inspired elements (e.g., wrinkled flames [Hong et al. 2007]) appeared in places, while additional post-processing heuristics were often layered on top: temperature smoothing and masking to approximate radiative heating and cooling [Cioroba et al. 2018; Feldman et al. 2003; Melek and Keyser 2002], sharp color-gradient remapping of soot fields at render time to emulate thin reaction zones, and dissipation-based heuristics to mimic soot oxidation [Horvath and Geiger 2009].

Although this approach granted broad creative freedom, it frequently ignored the fundamental interconnectivity between thermomechanical quantities and did not faithfully capture chemistry and flame front evolution, with the latter often being approximated with finite-rate chemical kinetics [Ihm et al. 2004; Kang et al. 2007; Stam and Fiume 1995]. The underlying real-world relationships were thus absent or misrepresented, shifting the burden to manually "re-create physics" through exhaustive trial-and-error tuning by artists. As explicitly noted in [Nielsen et al. 2022, 2019], even the most talented artists could produce realistic results only through great skill and a significant amount of tweaking time. This resulted in heavy cognitive load, inconsistent looks across large teams delivering high shot counts, long iteration cycles, and a proliferation of shot-specific parameters.

Turbulence. Another important requirement for VFX combustion solvers is the ability to inject or amplify turbulence and vorticity, though existing approaches each come with notable limitations. Curl noise [Bridson et al. 2007] offers flexible multi-scale velocity perturbations, but typically requires extensive per-shot tuning of amplitudes and frequencies to achieve believable results. Vorticity confinement [Fedkiw et al. 2001] boosts existing vortices in the flow, yet it is not physically grounded and can easily introduce visual artifacts when over-amplified, making it hard to control robustly. Wavelet turbulence [Kim et al. 2008] adds fine-scale detail as a post-process, but using more than one or two octaves often produces unnatural, easily recognizable patterns.

Physics-based combustion. A progression toward physics-based combustion simulation has long been advocated for [Feldman et al. 2003; Melek and Keyser 2002; Stam and Fiume 1995], with notable contributions in thin-flame modeling [Nguyen et al. 2002], solid-fuel burn and expansion [Losasso et al. 2006], and shock waves [Kwatra et al. 2010; Sewall et al. 2009; Yngve et al. 2000]. Most pertinently, [Nielsen et al. 2022, 2019] introduced a complete production-ready physics-based combustion system implemented within Autodesk's Bifrost, including realistic fuel-oxidizer mixtures, thin-flame propagation, soot formation/oxidation, simplified isobaric thermodynamics with ideal-gas expansion, and numerical improvements, balancing efficiency with limited compressibility (support for contraction/expansion, but not shock waves).

3 Motivation and contributions

3.1 Solver features

Our work was strongly inspired by the physics-based combustion solver presented in [Nielsen et al. 2022, 2019], which demonstrated that faithful chemistry-driven fire simulation is both feasible and highly valuable for production. While we admired both the quality and the design philosophy of that system, adopting it directly was not a practical option in our pipeline, as it is tightly coupled to Autodesk Maya—an environment that is largely unfamiliar to FX artists accustomed to Houdini-centric workflows. In addition, we sought to fully leverage the capabilities of our in-house framework Loki [Lesser et al. 2022], in particular its robust multiphysics infrastructure for coupled interactions with solids and liquids (Figure 3), as well as its sparse adaptive data structures designed for large-scale, distributed simulation using MPI (Figure 4).

While our design borrows a number of conceptual ideas from [Nielsen et al. 2022, 2019], it also differs in several key aspects: (i) we store and evolve absolute chemical amounts—rather than relative fractions—for improved mass tracking; (ii) we enforce the ideal gas law directly as a constraint, instead of using its differential form, to robustly accommodate volatile artistic inputs; (iii) we replaced

Figure 3: Kora builds on Loki's coupling capabilities to augment gaseous flames with entrained solid sparks. ©2025 20th Century Studios / Lightstorm Entertainment. All Rights Reserved.

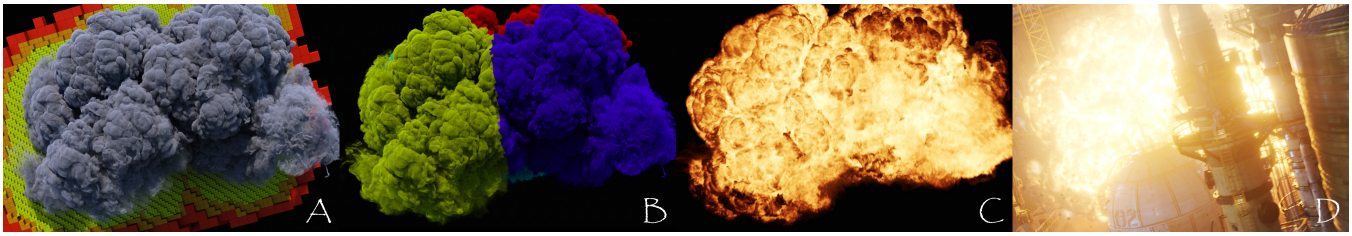

Figure 4: An explosion shot breakdown: (a) adaptive simulation grid's bucket structure, (b) color-coded ranks of the MPI partitioning, (c) render of the explosion element, and (d) the final compositing result.

the isobaric approximation with a hybrid isochoric solve followed by adiabatic expansion, capturing rapid expansion with *adiabatic cooling* more faithfully.

To enhance vortical flow structure in a physically grounded way with minimal artistic tuning, we developed a multi-scale *energy cascade turbulence* model. The approach restores dissipated kinetic energy by injecting turbulence according to Kolmogorov's spectrum, yielding rich multi-scale dynamics and visually compelling detail even at lower resolutions.

The new system gave rise to a variety of desirable and previously hard-to-achieve effects, such as oxygen starvation and flickering of thin flames, delivering consistent, reference-matching results, and was successfully deployed in *Avatar: The Way of Water*, as reported in [Edholm et al. 2023].

3.2 Production usability

Despite the solver's fidelity, practical adoption in production was limited by the expertise required to operate it. The solver proved effective when matching real-world references—where known chemical compositions and physical conditions produced results consistent with controlled experiments. In practice, however, VFX workflows rarely function in this manner. Artists typically receive feedback in terms of the desired final appearance, without direct knowledge of the underlying parameters governing combustion. Determining how to modify fuel composition or solver settings to achieve a particular visual outcome can therefore be challenging.

Moreover, multiple independently developed Loki-based combustion setups quickly emerged across our FX department. Created by different artists to solve immediate shot needs, these configurations spread and accumulated over time, resulting in a fragmented and increasingly difficult-to-navigate FX landscape. With no single authoritative configuration, best practices were absorbed informally and inconsistently, making it challenging for junior artists to achieve reliable results without extensive guidance.

Loki itself exposes a large number of expert-level parameters intended for multiphysics research and development, many of which should remain fixed for robustness or physical correctness. In reality, though, the users could inadvertently modify such settings, leading to divergent results, increased debugging time, and the need for repeated specialist intervention. This made the solver powerful but fragile when used outside a controlled context.

To address this gap between research-oriented physics-based simulation and artist-driven workflows, we developed Kora, a centralized framework that makes our new solver available through an intuitive set of tools within the familiar Houdini environment, mimicking what the Pahi system [Stomakhin et al. 2023] did for water. Specifically, the Kora toolset organizes its functionality into distinct complementary areas: sourcing, simulation control and art-direction, and rendering.

Sourcing. Kora automates the setup of premixed fuel-oxidizer combinations using stoichiometric and equivalence ratios. It supports volumetric stamping, particle-based, and fluxed sourcing approaches to accommodate various emission needs. Leveraging Loki's coupling capabilities, it configures multi-material simulations in which liquid fuels interact with air and vaporize into combustible gases while conserving mass and momentum; a similar approach supports interactions with solid particles such as sparks and soot.

Simulation control. Kora exposes only the solver controls that are meaningful for artistic iteration, while locking down—or automatically managing—expert parameters that should remain consistent across shots. By abstracting much of the underlying complexity, including thermodynamics and combustion chemistry, Kora enables artists to efficiently author sophisticated simulations and focus their time on creative iteration rather than low-level technical setup.

Art-direction. To retain physical plausibility while enabling direction, Kora provides minimally invasive steering mechanisms: (i) buoyancy manipulation via warped gravity fields, (ii) spatially varying Coriolis forces for controllable vortex formation, (iii) wind boundary constraints, and (iv) frequency-domain velocity guiding [Forootaninia and Narain 2020] adapted to sparse grids. These techniques allow artists to shape macroscopic behavior without suppressing salient dynamic features.

Rendering. Kora integrates with the Manuka renderer [Fascione et al. 2018], resolving complex volume compositing at render time. Combustion reaction fields drive advanced shading effects, facilitating the creation of sophisticated looks, while Kelvin temperatures are used directly to compute accurate black-body radiation.

4 Combustion solver

In this section we present our Eulerian physics-based combustion solver. We begin with the governing thermomechanical equations, highlighting how our approximations and tracked variables differ from those in [Nielsen et al. 2022, 2019], which enables a more robust expansion response to highly volatile artistic inputs and captures effects such as adiabatic cooling. We further describe our combustion reaction and flame front models, followed by discretization and implementation details of the solver routines. Finally, we introduce our novel energy cascade turbulence technique.

4.1 Governing equations

Let us recall the Eulerian form of continuum mechanics equations for compressible fluids. For more in-depth exposition we refer the reader to [Gonzalez and Stuart 2008].

Conservation of momentum. For a fluid with density ρ, velocity $\boldsymbol{u}$, and pressure p, in the presence of gravity $\boldsymbol{g}$, the Eulerian form of Newton's second law reads

$$\rho \frac{D\boldsymbol{u}}{Dt} = -\nabla p + \rho \boldsymbol{g}. \tag{1}$$

Here $\frac{D}{Dt} \equiv \frac{\partial}{\partial t} + \boldsymbol{u} \cdot \nabla$ is the *material derivative* operator. We ignore viscosity effects for simplicity of exposition.

Conservation of mass. The continuity equation describing the evolution of a density field ρ in a velocity field $\boldsymbol{u}$ is

$$\frac{D\rho}{Dt} = -\rho \nabla \cdot \boldsymbol{u}. \tag{2}$$

Note that for incompressible fluids this simplifies to zero-divergence condition $\nabla \cdot \boldsymbol{u} = 0$, which together with equation (1) forms the system of Euler equations.

Equation of state. For compressible fluids we need to establish a dependency between density and other state variables. We use the *ideal gas law* to relate density ρ to pressure p, molar mass M, and temperature T

$$\rho = \frac{pM}{RT}, \tag{3}$$

where R is the thermodynamic constant.

Temperature evolution. There is another advection-diffusion equation describing the evolution of temperature

$$\rho C \frac{DT}{Dt} = \kappa \nabla^2 T + W, \tag{4}$$

which together with equations (1), (2) and (3) forms a closed system in variables $\boldsymbol{u}$, p, ρ, and T. W represents energy sources which in our case are the result of chemical reactions, κ is thermal conductivity of the fluid, and C is its specific heat capacity. The latter depends on the fluid evolution regime. For instance, one may distinguish between isochoric C_V, isobaric C_p, or other variations.

4.2 Assumptions and simplifications

The system of equations (1), (2), (3), and (4) has severe numerical stability limitations associated with compressibility and is non-trivial to integrate directly.

4.2.1 Isobaric approximation. Following [Nielsen et al. 2022] one may replace variable p with fixed atmospheric pressure $p_{\text{atm}} = 101.3\text{KPa}$ in equation (3) and also consider temperature evolution in equation (4) to be isobaric (happening at constant atmospheric pressure). This simplification is based on the assumption that local variations in pressure ∇p driving mechanical evolution of the fluid are small enough not to introduce a considerable disturbance to the large absolute pressure of the atmospheric layer. It eliminates the possibility of shock waves and constrains simulatable scenarios to relatively slow contractions and expansions, but also makes the problem much easier to deal with numerically. The system of continuous equations to be solved becomes

$$\begin{cases} \rho \dfrac{D\boldsymbol{u}}{Dt} &= -\nabla p + \rho \boldsymbol{g}, \\ \nabla \cdot \boldsymbol{u} &= -\dfrac{1}{\rho} \dfrac{D\rho}{Dt}, \\ \rho &= \dfrac{p_{\text{atm}} M}{RT}, \\ \rho C_p \dfrac{DT}{Dt} &= \kappa \nabla^2 T + W. \end{cases} \tag{5}$$

Note the use of *isobaric* (constant pressure regime) specific heat capacity C_p.

While imperfect, the isobaric approach is a good first step and aligns with the existing state-of-the-art literature. Later in §4.4 we introduce an improved adiabatic approximation that handles rapid contractions, expansions and pressure variations more faithfully.

4.2.2 Operator splitting. A common approach to solve system (5) is to use operator splitting. It allows separating advection/convection into its own step of temporal integration. With that, the temperature evolution equation (4) becomes a parabolic heat diffusion

$$\rho C \frac{\partial T}{\partial t} = \kappa \nabla^2 T + W, \tag{6}$$

and can be readily evolved forward on its own, assuming other quantities are fixed, see §4.7.2. So in what follows, we will consider it to be solved, and the temperature field to be predefined. Molar mass can be viewed as predefined as well, given the results of the

Figure 5: Setting splashes of liquid fuel on fire in *Avatar: Fire and Ash*. Preview render (top) and the final compositing result (bottom).

combustion reaction step, see §4.5. Consequently, we are left with

$$\begin{cases} \rho \dfrac{\partial \boldsymbol{u}}{\partial t} &= -\nabla p + \rho \boldsymbol{g}, \\ \nabla \cdot \boldsymbol{u} &= -\dfrac{1}{\rho}\dfrac{\partial \rho}{\partial t}, \\ \rho &= \dfrac{p_{\text{atm}} M}{RT}. \end{cases} \tag{7}$$

4.3 Expansion and concentration

Rather than utilizing the ideal gas law in its differential form by substituting it into the density evolution equation (2), see [Nielsen et al. 2022], we enforce it as a constraint. This allows for more precise mass tracking by avoiding error accumulation and subsequent drift in the course of temporal integration. Additionally, this guarantees a robust response to user interventions, when sudden, non-differentiable changes in simulation quantities—due to mass or temperature stamping, or equilibrium adjustments as in [Flynn et al. 2024]—are required artistically.

Let ρ_{prev}, M, and T be density, molar mass, and temperature of the simulated fluid. Note that the combination may not necessarily satisfy the ideal gas law with pressure p_{atm}, indicating a potential for expansion or contraction. We can however determine the density that would satisfy it as

$$\rho_\star = \frac{p_{\text{atm}} M}{RT}. \tag{8}$$

The change in density from ρ_{prev} to the desired $\rho_\star$ over the duration of a timestep δt defines the expansion

$$-\frac{1}{\rho}\frac{\partial \rho}{\partial t} = -\frac{\partial \ln \rho}{\partial t} \simeq -\frac{1}{\delta t}(\ln \rho_\star - \ln \rho_{\text{prev}}) = -\frac{1}{\delta t} \ln s, \tag{9}$$

where we have introduced density scaling

$$s = \frac{\rho_\star}{\rho_{\text{prev}}}. \tag{10}$$

With that, system (7) becomes

$$\begin{cases} \rho \dfrac{\partial \boldsymbol{u}}{\partial t} &= -\nabla p + \rho \boldsymbol{g}, \\ \nabla \cdot \boldsymbol{u} &= -\dfrac{1}{\delta t} \ln s, \end{cases} \tag{11}$$

and can be solved using a pressure projection, see §4.7.1.

4.3.1 Measure of concentration. While being reasonable from a scientific standpoint, density [kg/m^3] or even moles per unit volume [1/m^3] are not the most convenient quantities for artists to deal with when it comes to specifying the amount of a chemical contained in a region of space. We introduce a new dimensionless measure

$$c = \frac{\rho}{\rho_{\text{atm}}}, \tag{12}$$

which we call *concentration*. Here ρ is the current density of a chemical and ρ_{atm} is its density in the thermodynamic equilibrium at pressure p_{atm} and room temperature $T_{\text{atm}} = 288.15$K

$$\rho_{\text{atm}} = \frac{p_{\text{atm}} M}{R T_{\text{atm}}}, \tag{13}$$

where M is the molar mass of the chemical. Substituting (13) into (12) gives

$$c = \frac{\rho R T_{\text{atm}}}{p_{\text{atm}} M}, \tag{14}$$

which means concentration of a chemical is simply its moles per unit volume ρ/M scaled by the $RT_{\text{atm}}/p_{\text{atm}}$ constant.

Concentration is intuitive for artists to use, as $c = 1$ means a chemical is exactly at the thermodynamic equilibruim with $p = p_{\text{atm}}$ and $T = T_{\text{atm}}$, and values greater or less than 1 indicate expansion or contraction, respectively.

Chemical mixtures. Given chemicals with concentrations c^i and equilibrium densities ρ^i_{atm}, the density of the mixture is then

$$\rho = \sum_i c^i \rho^i_{\text{atm}}. \tag{15}$$

With this, concentrations are similar to chemical fractions in [Nielsen et al. 2022], but they do not necessarily add up to 1, with the sum $c = \sum_i c^i$ serving as an indicator of a potential expansion or contraction. If molar masses of the chemicals are M^i, the molar mass of the mixture can be computed as

$$M = \frac{1}{c} \sum_i c^i M^i. \tag{16}$$

It is then easy to check that equation (14) holds for mixtures of chemicals as well.

4.3.2 Concentration change with expansion. Previously we have connected expansion to density change via equations (9) and (10). We can now rewrite it in terms of concentration and temperature. Assuming $c_\star$ and c_{prev} correspond to $\rho_\star$ and ρ_{prev}, as described by equation (14), respectively

$$s = \frac{\rho_\star}{\rho_{\text{prev}}} = \frac{c_\star}{c_{\text{prev}}} = \frac{1}{c_{\text{prev}}} \frac{\rho_\star R T_{\text{atm}}}{p_{\text{atm}} M} = \frac{1}{c_{\text{prev}}} \frac{T_{\text{atm}}}{T}. \tag{17}$$

The computation within the solver timestep enforcing expansion and mass/concentration conservation then proceeds in accordance with Algorithm 1 (isobaric regime).

4.3.3 Delayed expansion. Instead of performing immediate expansion to achieve the equilibruim concentration $c_\star$, one may wish to spread the effect over a period of time. We introduce a new parameter τ called *expansion relaxation time*, and rather than aiming for $c_\star$ by the end of a timestep we choose to target

$$\tilde{c} = c_\star + (c_{\text{prev}} - c_\star) \exp\left(-\frac{\delta t}{\tau}\right). \tag{18}$$

This makes fluid concentration approach $c_\star$ exponentially, reducing the discrepancy by a factor of e over time τ. The corresponding update to the Algorithm 1 is to replace equation (17) with

$$s = \frac{\tilde{c}}{c_{\text{prev}}} = \frac{1 + \left(\frac{c_{\text{prev}} T}{T_{\text{atm}}} - 1\right) \exp\left(-\frac{\delta t}{\tau}\right)}{\frac{c_{\text{prev}} T}{T_{\text{atm}}}}. \tag{19}$$

4.4 Adiabatic approximation

The isobaric approach described in §4.2.1 works well when $p \simeq p_{\text{atm}}$ is maintained at all times. This assumption, however, excludes certain important scenarios. For instance, an artist may want to stamp a large concentration of fuel at high temperature with the sole purpose of getting $p \gg p_{\text{atm}}$ to achieve rapid expansion of an explosion. Algorithm 1 (isobaric regime) will of course convert the excessive pressure into expansion, but, alas, incorrectly, as the former was introduced externally and not accounted for in the

Algorithm 1 The order of operations during a timestep of size δt. In *isobaric* regime the solver utilizes isobaric specific heat capacities $C = C_p$ of chemicals for all thermal updates; while in *adiabatic* regime isochoric specific heat capacities $C = C_V$ are used with an additional adjustment for cooling and expansion ($\gamma = C_p/C_V$).

Dissipation ▷ §4.7.5
Emission ▷ §5.1
Update domain buckets and load balance (MPI) ▷ §4.6
Mass diffusion ▷ §4.7.2
Combustion reaction ▷ §4.5
Radiative cooling ▷ §4.7.4
Thermal conduction ▷ §4.7.2
Energy cascade turbulence ▷ §4.8
Compute the sum c_{prev} of all chemical concentrations
Compute s from c_{prev}, T and T_{atm}, using equation (17)
if regime is *adiabatic* **then** ▷ §4.4
    $s \leftarrow s^{1/\gamma}$ ▷ expansion correction
    $T \leftarrow Ts^{\gamma-1}$ ▷ adiabatic cooling
end if
Apply external forces ▷ §5.3.1
Pressure projection with divergence $-\frac{1}{\delta t}\ln s$ ▷ §4.7.1
Velocity-based guiding ▷ §5.3.2
Multiply all concentrations by s to account for expansion
Advection ▷ §4.7.3

other stages of the timestep. Specifically, cooling—a prominent phenomenon accompanying expansion, associated with the system performing mechanical work—is missing from Algorithm 1 (isobaric regime), leaving temperature unaffected.

In order to rectify this issue, we abandon the isobaric assumption. Instead, we perform thermodynamic computations at constant volume, using *isochoric* specific heat capacities C_V, intentionally avoiding any and all expansion effects. Once we get to the pressure projection we perform a single *adiabatic* expansion—an appropriate choice for a closed thermodynamic system.

Adiabatic process for an ideal gas is constrained by the polytropic relation $pV^\gamma = \text{const}$, where V is volume and $\gamma = C_p/C_V$ is the adiabatic index. Since $V \propto \frac{1}{\rho}$, we can rewrite it as

$$p \propto \rho^\gamma \tag{20}$$

Thus, in order to equalize pressure through expansion the required density scaling must now be $s^{1/\gamma}$. From the ideal gas law and equation (20) one can infer that $T \propto \rho^{\gamma-1}$. Hence, the temperature should be scaled by $s^{(\gamma-1)/\gamma}$ implying cooling alongside expansion. The full procedure is summarized in Algorithm 1 (adiabatic regime).

4.5 Combustion

Note: italicized quantities introduced in this section correspond to established physical and chemical concepts documented in the literature and are exposed as solver parameters with real-world default values taken from [NIST 2025].

4.5.1 Chemical reactions. The solver models the burning of hydrocarbon fuels by explicitly tracking interactions of those with oxygen, nitrogen, carbon dioxide, and water vapor. The system supports multiple common real-world fuels, including methane, propane, and complex compounds such as diesel, each defined by their thermochemical properties. Nitrogen and existing combustion products act as chemically inert, diluting the gas mixture and participating only thermomechanically by means of added concentration and heat capacity.

The dominant chemical process is *fuel oxidation*, in which hydrocarbon fuel reacts with available oxygen to form carbon dioxide and water vapor while releasing chemical energy as heat. Physically, this heat corresponds to the *enthalpy of combustion*. It raises the local temperature, driving volumetric expansion via the equation of state and, consequently, generating buoyant motion.

A key concept governing combustion behavior is the *stoichiometric ratio*, which defines the ideal proportion of fuel and oxygen required for complete combustion, leaving neither species in excess. The *equivalence ratio* is defined as the actual fuel-oxygen mixture relative to this stoichiometric case. Mixtures with an equivalence ratio below one are fuel-lean and contain excess oxygen, whereas mixtures above one are fuel-rich and oxygen-deficient.

As an example consider the balanced chemical equation for complete combustion of propane

$$C_3H_8 + 5O_2 \rightarrow 3CO_2 + 4H_2O + \text{heat}. \tag{21}$$

It implies that 5 moles of oxygen are required for complete oxidation of 1 mole of propane. Hence the stoichiometric ratio of propane is (1:5). With this, a (1:10) ratio, corresponding to the equivalence ratio of (1:2), would imply a fuel-lean combustion with excess oxygen. On the other hand (2:5) ratio, corresponding to equivalence ratio of (2:1), would create a fuel-rich, oxygen deficient mixture.

Fuel-rich conditions give rise to *oxygen starvation*, in which combustion becomes locally oxygen-limited and flame fronts intermittently ignite and extinguish as fresh oxygen is entrained from the surrounding flow. This naturally produces visual phenomena known as choked flames, pulsation, and flickering. Because the solver tracks chemicals and models reactions explicitly, these behaviors emerge directly from the local availability of reactants rather than from heuristic noise or temporal modulation.

In addition to the primary oxidation reaction, the solver models *soot formation* as a subsequent process occurring in fuel-rich regions. When excess fuel remains after oxygen has been locally depleted, carbon-rich intermediates can nucleate into soot particles. Since soot represents solid particulate matter rather than a gas, it does not add pressure to the system. Accordingly, soot concentration is excluded from the equation of state, ensuring that its presence only influences density, but not expansion.

Soot oxidation provides the opposing mechanism to soot formation. In high-temperature, oxygen-rich regions—typically near flame cores—soot particles can be consumed through oxidation, converting them into gaseous combustion products.

Within the Eulerian solve, fuel oxidation is carried out on a per-voxel basis following equation (21): the reactant concentrations are reduced, the product concentrations are increased, and the released heat is converted into temperature change, using the heat capacity of the mixture. The amount of soot formed from the excess fuel in a fuel-rich mixture is controlled by a *soot formation rate* parameter. In our implementation soot carries chemical, thermal and mechanical properties of the fuel it came from, and consequently the soot oxidation process follows the same equation (21).

Figure 6: Flamethrower reference footage (bottom), matched by a Kora simulation (middle), and the resulting shot from *Avatar: Fire and Ash* (top). ©2025 20th Century Studios / Lightstorm Entertainment. All Rights Reserved. ©2026 Wētā FX Ltd.

4.5.2 Flame front. Flame front defines the region of space where the combustion reaction takes place, and we represent it using a signed distance field (SDF). It is initiated by voxels which meet both ignition criteria: (i) the temperature is greater or equal to the *ignition temperature* of the fuel; (ii) the equivalence ratio is within the *flammability limits*.

The SDF is propagated in the normal direction based on the *flame speed* v

$$\delta\phi/\delta t = -v, \tag{22}$$

which allows the burning region to expand to new areas where the fuel and oxygen mixture is sufficient for combustion. Similar to [Nielsen et al. 2022], this propagation is done using SDF dilation followed by re-normalization. The SDF is only discretized in the narrowband of 5 voxels on each side of the zero interface, which limits the amount of dilation that can be done. In order to support large flame speeds, the dilation and re-normalization is done in multiple iterations.

4.6 Spatial discretization

4.6.1 Data structure. Eulerian data is traditionally discretized on a three-dimensional Cartesian grid. We employ a sparse discretization that decomposes space into uniform blocks, or *buckets* [Bridson 2003], allocating memory only in regions of interest. The bucket size is controlled by a user-defined parameter $\mathcal{B}$. Regions of interest are typically initialized from emitter bounding boxes or user-specified volumes and are updated dynamically based on the quantities they contain as the simulation progresses. An illustration of the bucket structure can be found in Lesser et al. [2022, Figure 10].

Every simulation quantity is represented as a *channel*, defined by a name, data type (e.g., float, vec3f, integer), subdivision level (4^3, 8^3, 16^3), and an initial value. Within a bucket, each channel defines a local grid of values according to its subdivision. We use a finer 8^3 subdivision for channels representing chemical species and thermal quantities, and a coarser 4^3 subdivision for mechanical quantities. As a result, chemical reactions are resolved at twice the spatial resolution of the fluid dynamics, a trade-off that has proven effective in balancing simulation cost and rendered visual fidelity.

Padding is used to dilate the sparse simulation domain beyond the immediate region of interest. This additional margin is necessary to capture surrounding velocity flow and to allow the simulation to expand naturally into newly activated regions. Insufficient padding can cause the fluid to become artificially constrained, as advection lacks available buckets into which quantities may propagate, while excessive padding yields little additional visual benefit relative to the increased computational cost. In practice, a padding of three buckets has proven to be a reliable default.

User-defined *bounds* restrict the spatial extent within which buckets may be created and are essential for preventing unbounded domain growth, keeping simulation size and runtime under control.

4.6.2 Adaptivity. To concentrate computational effort in regions of interest, the solver employs spatial adaptivity by varying the discretization resolution. This is achieved through buckets of different sizes, where each bucket size is an integer power-of-two multiple of the base resolution $\mathcal{B}$, with the finest resolution corresponding to $\mathcal{B}$ itself. Although the implementation does not rely on an explicit octree data structure, equivalent grading constraints are enforced to limit resolution transitions between neighbouring buckets to at most a factor of two. Smooth and stable data transfer across resolution boundaries is ensured using the moving least squares (MLS) interpolation strategy described in [Ando and Batty 2020].

We employ three complementary methods for controlling spatial adaptivity, all of which can be combined within a single simulation. The first method extends standard padding with *adaptive padding*, in which progressively coarser buckets are introduced around already discretized regions of interest. This significantly extends the solver's ability to capture large-scale ambient air flow around combustion regions (Figure 4a) within a fixed computational budget; insufficient padding can otherwise lead to artificially constrained, overly damped explosions. The second method drives adaptivity using a user-defined control field. A common example is a camera-frustum field, where bucket resolution decreases with increasing distance from the camera, concentrating detail where it contributes most to the final image. The third method relies on heuristics derived from the simulation state. These heuristics are expressed as user-defined volume expressions, allowing the resolution to adapt dynamically in response to evolving flow features.

4.6.3 MPI. Distributed computing enables the simulation to be partitioned across multiple machines or ranks, with each rank operating on a subset of the domain. This allows simulations to scale beyond the memory limits of a single machine and, when sufficient parallelism is available, to reduce wall-clock time by distributing work across multiple nodes. In addition to enabling sparsity, the bucket-based discretization naturally supports domain decomposition for distributed execution. Buckets provide a convenient unit

for partitioning simulation data among ranks and for exchanging boundary data between neighbouring buckets located on different ranks. Buckets are partitioned and load-balanced using the approach described in [Wretborn et al. 2026], ensuring efficient utilization of computational resources (Figure 4b).

4.7 Solver routines

4.7.1 Pressure projection. We solve system (11) using the variational pressure formulation of [Batty et al. 2007], which employs voxel face area fractions to accurately represent kinematic solid boundaries. Buoyancy arises from combining spatially varying density with hydrostatic pressure Dirichlet boundary conditions, enforced on the outer layer of voxels of the sparse domain. The expansion term resulting from the updated concentrations and temperature (see §4.3.3) is incorporated as a modifier to the divergence constraint. The resulting linear system is solved using conjugate gradient method with algebraic multigrid preconditioner (Lesser et al. [2022, §6.5]).

4.7.2 Temperature and mass diffusion. We initially discretized and solved the heat equation (6) using an approach analogous to the pressure projection described in §4.7.1. However, because the linear system needed to be solved for quantities discretized at twice the resolution of the pressure grid, this approach proved too costly for production use. Instead, we approximate the solution using convolution with Gaussian kernels, which we found to yield negligible visual differences while significantly reducing computational overhead. To prevent excessive smoothing in actively burning regions, we use the flame-front signed distance field together with the interior flame temperature as a Dirichlet boundary condition.

A similar strategy is employed for mass diffusion of fuel and oxygen. Each species is diffused independently using their corresponding mass-diffusivity coefficient.

4.7.3 Advection. The convection terms of system (5) are handled using a standard semi-Lagrangian approach, applied uniformly to all advected quantities, including chemical species, temperature, the flame-front SDF, and velocity. The solver supports multiple combinations of integration schemes—first-, second-, and third-order Runge-Kutta—and interpolation methods, including linear, quadratic, and cubic interpolation. These can be augmented with error-correction techniques such as MacCormack [Selle et al. 2008], BFECC [Kim et al. 2005], and advection-reflection [Zehnder et al. 2018]. The different combinations yield subtly different visual characteristics and expose a trade-off between numerical accuracy, visual fidelity, and computational cost.

4.7.4 Radiative cooling. Radiative heat emission from hot volumes can be modeled using the Stefan-Boltzmann law

$$\mathcal{M} = \sigma \epsilon T^4, \tag{23}$$

where $\mathcal{M}$ is the *radiant exitance* (energy radiated per unit surface area per unit time, $[\mathrm{J}/(\mathrm{s}\cdot\mathrm{m}^2)]$), T is the local temperature, ϵ is the dimensionless emissivity coefficient, and σ is the Stefan-Boltzmann constant. Accurately modeling radiative emission and subsequent re-absorption is inherently non-local and typically requires ray-based or transport methods, which are computationally expensive for large-scale volumetric simulations.

Our goal is to capture the characteristic visual effects of radiative cooling without explicitly simulating full radiative transfer. In particular, we seek to preserve the temperature dependence of the Stefan-Boltzmann law, with emitted energy proportional to σT^4, while allowing emissivity to vary spatially based on soot concentration c_{soot}, as denser soot regions radiate more efficiently. Additionally, the formulation must express radiative losses as heat removal per unit volume, $[\mathrm{J}/(\mathrm{s}\cdot\mathrm{m}^3)]$.

From dimensional considerations, an expression of the form

$$\nabla\Big(\sigma c_{\mathrm{soot}}\left(T^4 - T_{\mathrm{amb}}^4\right)\Big) \tag{24}$$

satisfies these requirements. The subtraction of the ambient temperature T_{amb} prevents radiative losses below ambient conditions. Expanding the gradient yields our final radiative cooling formula

$$Q = \alpha\,\sigma\left(T^4 - T_{\mathrm{amb}}^4\right)\|\nabla c_{\mathrm{soot}}\| + 4\beta\,\sigma c_{\mathrm{soot}} T^3 \|\nabla T\|, \tag{25}$$

where α and β are user-defined emissivity coefficients. The separation into two terms allows artistic control over whether cooling is driven primarily by soot gradients or by thermal structure.

For each voxel, temperature and soot gradients are computed using finite differences. The resulting heat loss rate is converted into a temperature rate of change as

$$\dot{T} = \frac{Q}{\rho C} \quad [\mathrm{K/s}], \tag{26}$$

where ρC is the total volumetric heat capacity of all chemical species in the voxel. Temperature is then evolved using an exponential decay formulation

$$T \leftarrow T_{\mathrm{amb}} + (T - T_{\mathrm{amb}})\exp\left(-\frac{\dot{T}\,\delta t}{T}\right), \tag{27}$$

which offers improved numerical stability for large timesteps and prevents overshooting.

4.7.5 Dissipation. Dissipation provides an artistic control that enables gradual decay of a chemical concentration c over time

$$c \leftarrow c\exp\left(-\frac{\delta t}{\tau_d}\right), \tag{28}$$

where τ_d is a user-defined dissipation relaxation time specific to that chemical species. To prevent the formation of locally depressurized regions as material is removed, the dissipated amount is redistributed to ambient species, such as nitrogen or oxygen.

4.8 Energy cascade turbulence

Energy cascade turbulence (ECT) is a physically motivated turbulence enrichment method that injects divergence-free motion across multiple spatial scales in order to restore kinetic energy lost to numerical dissipation. Traditionally, this loss has been addressed by manually adding various noises at a multitude of scales, which requires careful tuning of amplitudes and frequencies and does not adapt automatically to the evolving simulation state. ECT removes this tuning burden by estimating how much kinetic energy is missing at each scale and injecting only the amount of turbulence needed to restore a physically plausible energy distribution. The method has been used in multiple generations of our in-house combustion solvers over the last decade and was originally inspired by the cascade-ratio approach of [Ishimuroya and Kanai 2016].

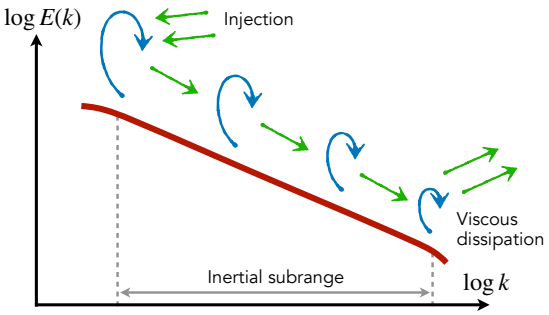

Figure 7: Kolmogorov spectrum formalizes the notion that kinetic energy enters the system at low frequencies and subsequently cascades toward higher frequencies. ©2026 Wētā FX Ltd.

The fundamental building block of ECT is curl noise [Bridson et al. 2007], which generates incompressible velocity fields by applying the curl operator to a vector noise potential. In practice, the partial derivatives required to compute the curl are evaluated using finite differences, and the resulting field acts as an acceleration term in the simulation.

ECT uses the Kolmogorov energy spectrum as a reference model for turbulent energy distribution. In a fully developed flow, the energy density $E(k)$ at spatial frequency k follows

$$E(k) \propto k^{-\frac{5}{3}}, \tag{29}$$

see Figure 7. ECT treats this spectrum as a target distribution: deviations from it are interpreted as scale-dependent energy loss due to numerical dissipation.

To estimate these losses, the velocity field needs to be decomposed into a set of frequency bands [Chui 1992] for which we employ recursive smoothing. Starting from the original velocity field $\boldsymbol{u}^0 = \boldsymbol{u}$, successive filtered fields are computed as

$$\boldsymbol{u}^{i+1} = [\underbrace{\mathcal{J} \circ \ldots \circ \mathcal{J}}_{2^i}](\boldsymbol{u}^i), \tag{30}$$

where $\mathcal{J}$ is a $3 \times 3 \times 3$ box convolution operator in voxel space, applied 2^i times to achieve a coarsening factor of 2 between consecutive frequencies. Band-limited velocity components are then obtained by differencing adjacent levels

$$\boldsymbol{u}^i_{\text{band}} = \boldsymbol{u}^i - \boldsymbol{u}^{i+1}. \tag{31}$$

For each band i, the local kinetic energy $E^i = \frac{1}{2}\|\boldsymbol{u}^i_{\text{band}}\|^2$ is estimated and compared to that of the next coarser band E^{i+1}. If the observed energy ratio E^i/E^{i+1} falls below the falloff $r_e = 2^{-5/3}$ implied by equation (29), the deficit is interpreted as missing turbulent energy at that scale. ECT compensates for this via added acceleration

$$\boldsymbol{a}^i_{\text{turb}} = \begin{cases} \mathcal{R}_{\text{turb}}(l_i)\, A_i\, \nabla \times \boldsymbol{N}^i, & \text{if } E^i/E^{i+1} < r_e, \\ 0, & \text{otherwise,} \end{cases} \tag{32}$$

where $\boldsymbol{N}^i$ is a curl noise potential matched to the band feature size $l_i = 2^i \Delta x$ (Δx is the voxel size) and A_i is the amplitude, defined as

$$A_i = \sqrt{2 r_e E^{i+1}} - \sqrt{2E^i}. \tag{33}$$

$\mathcal{R}_{\text{turb}}(l)$ is a user-provided curve that maps feature size to turbulence strength. This curve provides artistic control for selectively emphasizing or suppressing turbulence at certain scales.

The turbulence contributions from all frequency bands are accumulated and applied back to the velocity field

$$\boldsymbol{u} \leftarrow \boldsymbol{u} + \sum_{i=0}^{n-1} \boldsymbol{a}^i_{\text{turb}}\, \delta t, \tag{34}$$

where n is the number of frequency bands. Increasing n expands the range of resolved turbulent scales but incurs higher computational cost, as the number of convolution operations doubles with each additional band. In practice, we have found that using five bands provides a robust balance between visual richness and performance. The visual comparison of ECT with wavelet turbulence and vorticity confinement is shown in Figure 8.

5 Kora architecture

As outlined in the introduction, the development of the Kora toolset was triggered by rising usability concerns around our physics-based combustion model, whose physically grounded controls often felt unintuitive in practice. At the same time, the underlying low-level Loki setups—particularly those involving multi-material coupling—had grown increasingly complex, making them difficult for artists to manage directly.

Kora provides a collection of carefully crafted HDAs, developed by expert TDs in close collaboration with researchers and developers. These HDAs encapsulate the complex Loki graphs and abstract away the intricacies of the combustion model, exposing only the artist-relevant controls through clear and intuitive interfaces. In addition, they streamline and standardize the preparation of inputs for the solver, and its outputs for the renderer. From a workflow perspective, Kora's functionality divides naturally into the following distinct stages: sourcing, simulation control and art-direction, and rendering (Figure 9).

5.1 Sourcing

The influence of emission on the final appearance of a simulation is at least as significant—if not greater—than the settings of the combustion model itself. Providing artists with clear, intuitive sourcing tools was therefore paramount.

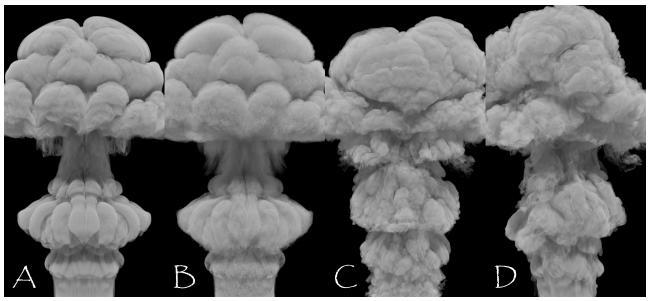

Figure 8: Comparison of turbulence models. (a) No turbulence, (b) wavelet turbulence with 3 octaves of fine scale noise, (c) vorticity confinement with strength 6 (which took many iterations to dial in for this example), (d) energy cascade turbulence with default amplitude multiplier of 1. ©2026 Wētā FX Ltd.

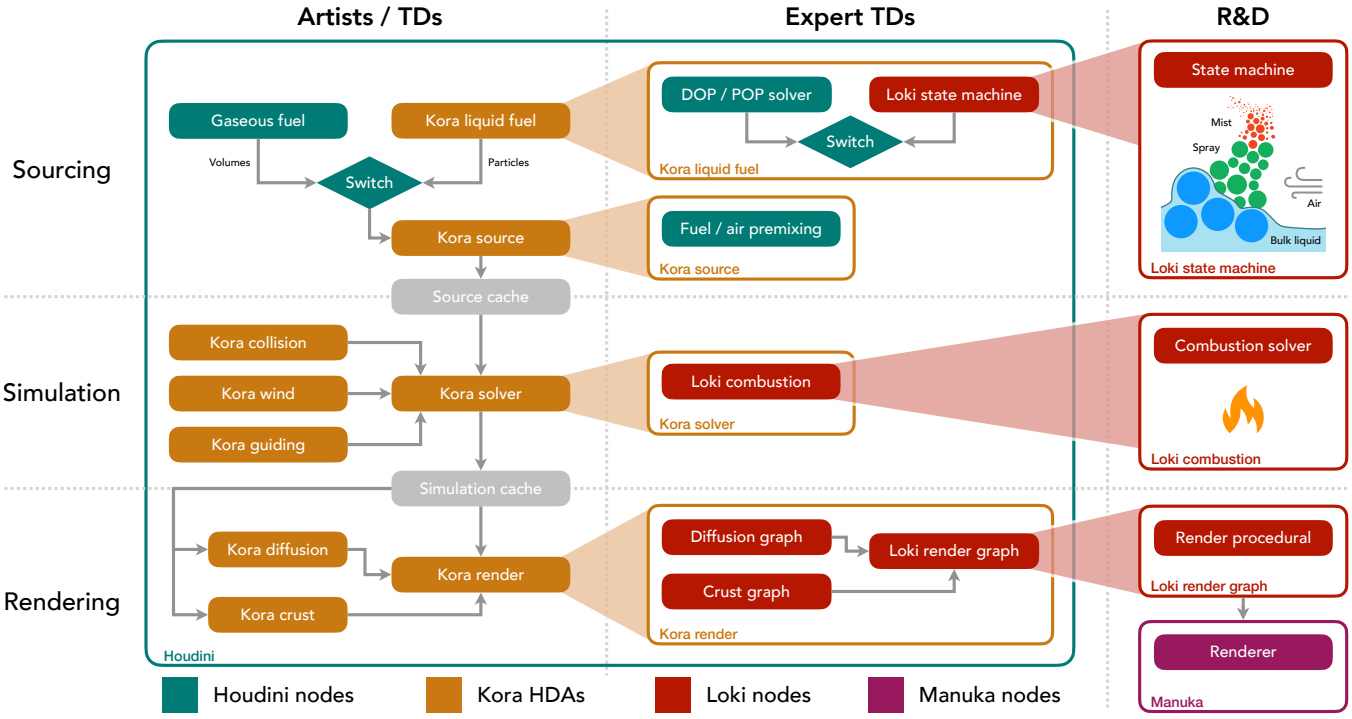

Figure 9: Kora architecture. Kora consists of low-level Loki solvers and render graphs wrapped into artist-friendly HDAs, as well as additional Houdini-centric tools. The diagram illustrates the data flow from sourcing to simulation, to rendering (top-to-bottom) and encapsulation / authoring hierarchy from artists to expert TDs, to researchers and developers (left-to-right). ©2026 Wētā FX Ltd.

5.1.1 Pre-mixing. The solver operates on absolute chemical quantities expressed as dimensionless concentrations, and it expects sourcing inputs in that form. However, asking artists to reason about fuel and oxygen amounts in terms of stoichiometry—and to map those values to the resulting flame appearance—introduces unnecessary friction. To avoid this, the *Kora source* node automates fuel-oxygen pre-mixing: artists simply specify the total amount of emitted mixture and choose how much oxygen is present relative to the stoichiometric optimum. This provides intuitive, direct control over the sootiness or cleanliness of the resulting flame without requiring them to manipulate absolute concentrations, see Figure 10. Mathematically, the oxygen pre-mixing ratio equals to the inverse of the equivalence ratio.

5.1.2 Emission variants. Kora allows artists to choose among the various emission mechanisms supported by the solver. The most flexible and commonly used option is volumetric stamping, where the user defines an emission region via an implicit SDF, where the solver's temperature, velocity and chemical concentrations are combined with the values of the corresponding input fields. The combine operation can be configured to overwrite existing values, add to them, or take the minimum or maximum. When the incoming fields are spatially bounded and already define their own footprint, the distance field can be omitted.

Stamped emission gives artists a lot of creative freedom and is useful for rapid iteration on one-off shots. But because it bypasses physical reasoning and directly edits the fields, the amount of mass, momentum, and energy it injects cannot be meaningfully specified or even assessed upfront. As a result, moving such a setup to a shot with different numerical settings or flow conditions often leads to unexpected behavior and requires re-tuning.

A more physically grounded alternative is flux-based emission [Stomakhin and Selle 2017]. Although Kora implements this approach, it tends to have limited practical value unless the source has a clean, well-defined inlet.

With the above limitations in mind, we designed a particle-based emission approach. An artist provides a set of particles carrying velocity, temperature, and pre-mixed chemical composition produced by the Kora source node. The solver then performs an MPM-style [Stomakhin et al. 2013] rasterization of these particles into the temperature, velocity, and chemical fields, ensuring conservation of mass, momentum, and thermal energy. The particles themselves

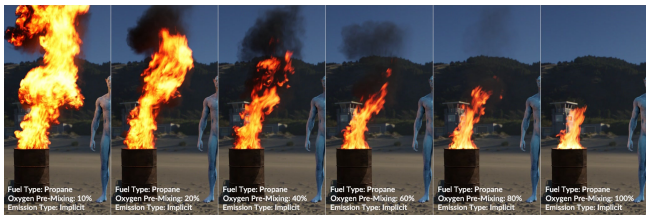

Figure 10: Fuel-oxygen pre-mixing. Increasing a source's oxygen pre-mixing ratio (left-to-right) leads to shorter, due to less overall emitted fuel, less sooty flames. At 10% pre-mixing, 90% of fuel is initially oxygen-starved and only ignites once it mixes with ambient air; at 100% pre-mixing, the source emits a perfectly stoichiometric fuel-oxygen mixture. ©2025 20th Century Studios / Lightstorm Entertainment. All Rights Reserved. ©2026 Wētā FX Ltd.

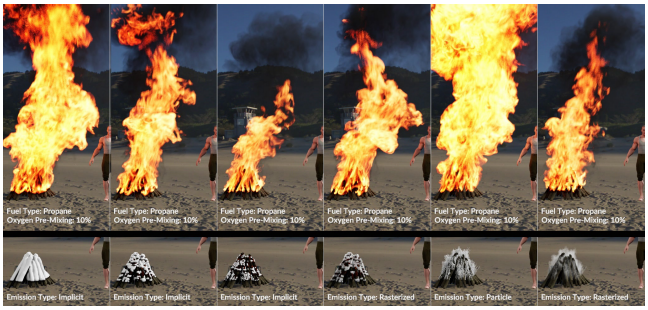

Figure 11: Emission types. Variation in look across emission types and input shapes. *Implicit* and *rasterized* represent stamped emission with and without an implicit SDF. ©2026 Wētā FX Ltd.

can be generated procedurally, via a Houdini POP simulation, or by any other means. Examples of the different emission types are shown in Figure 11.

5.1.3 Coupling. While pure gaseous combustion can be compelling, a lot of dramatic visual complexity arises when gaseous flows interact with other, submerged materials—such as sparks entrained within flames, see Figure 3. Kora leverages Loki's coupling capabilities to set up and manage these kinds of interactions.

Liquid fuels are a clear example of essential non-gaseous materials. Their higher density means they experience less deceleration from aerodynamic drag, producing effects like dripping or spitting fire (Figure 5), and allowing flamethrowers to shoot much farther than a gaseous jet ever could (Figure 6). We model liquid fuels using the Loki state machine [Stomakhin et al. 2023]. A stream of liquid fuel is emitted using flux-based emission and then breaks up into spray and eventually mist droplets, all of which are two-way coupled with the surrounding gaseous volume. Once in the mist phase, the droplets can be "vaporized" by splatting them into the volume using our particle-based emission approach, and subsequently ignite. To enable flexible production workflows, the Loki liquid-fuel state machine and the volumetric combustion solver may be executed separately in sequential simulation stages.

5.2 Simulation control

The *Kora solver node* exposes a carefully curated subset of the underlying Loki combustion solver's controls, balancing artistic flexibility with physical robustness. While the full Loki solver graph contains hundreds of parameters, many correspond to fixed physical constants or low-level numerical configuration choices that should remain invariant across shots. Examples include air density, gravitational acceleration, and solver tolerances, which rarely benefit from artistic adjustment and can easily destabilize simulations. Kora therefore exposes only those parameters which have a clear, predictable impact on the visual character of fire and smoke and that are commonly adjusted in response to creative direction. We list some of the most prominent ones below.

Combustion rate controls how quickly fuel reacts with oxygen, directly affecting fire intensity and brightness. *Flame speed* determines how fast the combustion front propagates into combustible regions. *Emissivity* governs how efficiently thermal energy is lost through radiation. *Expansion relaxation time* sets how quickly the fluid expands or contracts toward thermodynamic equilibrium. *Mass diffusivity* and *thermal diffusivity* control the spread of chemical species and heat. *Energy cascade strength* controls the amount of kinetic energy injected into underrepresented turbulent scales. *Soot formation rate* determines how quickly soot is generated during combustion. *Soot dissipation rate* controls how rapidly soot is removed from the simulation. *Soot oxidation rate* governs how quickly soot is consumed in high-temperature regions.

In contrast, advanced numerical settings—such as discretization schemes and internal stabilization parameters—remain hidden and are managed automatically. By exposing only intuitive, visually meaningful controls and handling expert-level settings internally, Kora reduces cognitive load while preserving physical plausibility and consistency. Artists can efficiently iterate on fire looks with confidence in the solver's stability and robustness.

5.3 Art-direction

Realistic fire behavior is governed by complex physical processes—including combustion, heat transfer, buoyancy, and turbulence—all interacting dynamically. Kora is designed to capture these phenomena with high fidelity, delivering accurate and believable fire simulations. However, visual effects work often demands more than realism. Creative direction may require flames to twist into specific shapes, follow precise paths, or exhibit magical qualities that defy natural laws. In such cases, strict physical accuracy alone cannot achieve the desired result.

To bridge this gap, Kora introduces guiding mechanisms, allowing artists to steer the simulation while preserving its natural complexity. These mechanisms fall into two main categories—force-based and velocity-based—each integrating differently with the solver and offering distinct advantages for artistic direction. In what follows, we outline these two categories with specific examples implemented in Kora.

5.3.1 Force-based guiding. Force-based controls modify the momentum equation (1) by adding artist-defined external force terms

$$\rho \frac{D\boldsymbol{u}}{Dt} = -\nabla p + \rho \boldsymbol{g} + \mathbf{F}_{\text{ext}}. \tag{35}$$

Even when the forces themselves are supernatural, their application integrates smoothly with the rest of the solve, preserving the natural turbulent character of the flow and producing believable motion

Figure 12: A fire tornado created using a warped gravity field for a local updraft along the specified trajectory and Coriolis force to induce rotation. ©2025 20th Century Studios / Lightstorm Entertainment. All Rights Reserved. ©2026 Wētā FX Ltd.

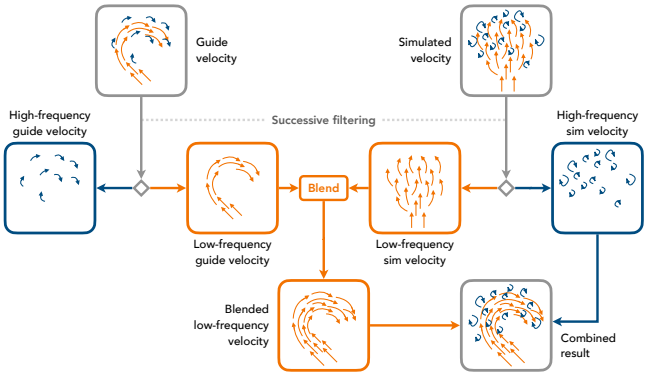

Figure 13: Schematic representation of the frequency-domain guiding approach. ©2026 Wētā FX Ltd.

that fits seamlessly within the combustion processes. The trade-off is that achieving a specific target velocity field can be difficult, if not impossible.

Updrafts. Local updrafts along complex trajectories are common in real-world fires, driven by spatial variations in buoyancy. Because buoyancy is itself a gravity-driven force, Kora creates warped gravity fields that steer the flow along specified paths without compromising its natural turbulent motion.

Coriolis forces. Inspired by the formation of cyclones and anticyclones, Kora introduces a truncated form of the Coriolis force, dropping non-local terms

$$\mathbf{F}_{\text{Coriolis}} = \rho \boldsymbol{\omega} \times \boldsymbol{u}. \tag{36}$$

This force twists buoyant flows into spinning vortices, producing effects ranging from subtle spirals to full fire-tornado motion, see Figures 1(right), 12, and 17. Artists shape the vortex by defining the angular-velocity field $\boldsymbol{\omega}$, either as a uniform value for global rotation or as a spatially varying field for localized swirling behavior.

5.3.2 Velocity-based guiding. Velocity-based controls perform

$$\boldsymbol{u} \leftarrow \boldsymbol{f}(\boldsymbol{u}, \boldsymbol{u}_{\text{guide}}), \tag{37}$$

where $\boldsymbol{f}$ combines the simulated velocity $\boldsymbol{u}$ with the artist-provided target velocity field $\boldsymbol{u}_{\text{guide}}$. While this gives artists complete creative control over the flow, it wipes out the solver's generated dynamics and tends to produce unnatural motion if applied indiscriminately. For this reason, we use velocity-based control selectively—for example, only at the domain boundaries or restricted to the low-frequency components of the velocity field.

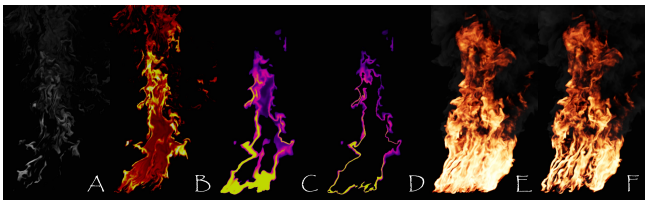

Figure 14: Slices of channels produced by the combustion solver: (a) soot, (b) temperature, (c) heat, and (d) hollow flame. Shading results (e) and (f) demonstrate the use of heat and hollow flame channels as flame alpha, respectively. ©2026 Wētā FX Ltd.

Wind. Wind control injects artist-authored velocity fields as Neumann boundary conditions, enforced on the outer layer of voxels of the sparse domain in the pressure projection (§4.7.1), and uses them to initialize new sparse-grid regions and out-of-domain advection backtraces. Applied sparsely, it guides the large-scale motion while preserving solver-driven dynamics.

Frequency-domain guiding. Frequency-domain velocity guiding [Forootaninia and Narain 2020] decouples a fluid's macroscopic motion from its microscopic detail. Guiding is applied exclusively to the low-frequency component of the velocity field, enabling artists to steer the overall trajectory of flames or smoke plumes without suppressing the fluid's natural complexity. This approach respects the cascade principle, allowing energy to transfer naturally from large to small scales, in line with our energy cascade turbulence technique (see §4.8).

To adapt frequency-domain guiding to sparse grids, we implemented the frequency decomposition using convolution filtering rather than Fourier transforms. The velocity field is successively smoothed via averaging $\mathcal{J}$ over a $3 \times 3 \times 3$ box in voxel space

$$\boldsymbol{u}_{\text{low}} = [\mathcal{J} \circ \ldots \circ \mathcal{J}](\boldsymbol{u}), \tag{38}$$

$$\boldsymbol{u}_{\text{high}} = \boldsymbol{u} - \boldsymbol{u}_{\text{low}}. \tag{39}$$

The resulting low-frequency velocity component $\boldsymbol{u}_{\text{low}}$ is blended with the corresponding low-passed guide field using a weight and a spatial mask, while the preserved high-frequency component $\boldsymbol{u}_{\text{high}}$ is re-introduced back to form the final result (see Figure 13)

$$\boldsymbol{u} \leftarrow \text{Blend}(\boldsymbol{u}_{\text{low}}, \boldsymbol{u}_{\text{guide}}) + \boldsymbol{u}_{\text{high}}. \tag{40}$$

A note on up-resing. The wind control can be extended into a general fluxed-boundary constraint by specifying the influx of additional simulation quantities—not just velocity—from outside the domain. This enables workflows such as re-simulating a region of an existing combustion cache at higher resolution by feeding the cached channels into the region's boundary constraint. Because changes in resolution can alter the resulting flow behavior, frequency-domain guiding can be used to better match the re-simulated version to the original.

5.4 Rendering

5.4.1 Shading. Kora drives shading effects by combining the output fields from the Loki combustion solver, as shown in Figure 15. Photorealistic flame colors are computed from the Kelvin temperature channel T using accurate black-body radiation, while flame alpha is derived from the heat channel H, which stores the enthalpy of combustion. To make flames appear sharper—particularly important for close-up shots—we further modulate the heat channel using the equivalence-ratio field ϕ, smoothly suppressing values

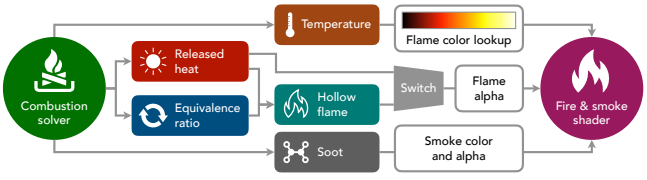

Figure 15: The channels produced by the combustion solver are used to drive fire and smoke shading. ©2026 Wētā FX Ltd.

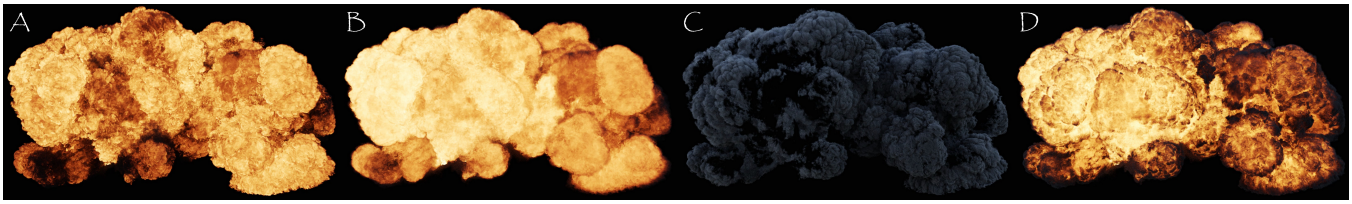

Figure 16: A demonstration of Kora's render-time operations: (a) raw Loki output, (b) Kora diffusion, (c) Kora crust, and (d) the final result combining diffusion and crust. ©2026 Wētā FX Ltd.

near 0 and 1

$$\zeta = \left(1 - (2\phi - 1)^4\right) H. \tag{41}$$

We refer to this derived quantity as the *hollow flame*, see Figure 14. Soot is shaded straightforwardly by using its concentration to drive alpha and mapping a shade of gray to its color.

5.4.2 Volume compositing. Kora leverages Loki's direct integration with our in-house renderer Manuka to support volume compositing at render time through a powerful node-based render graph. Unlike traditional workflows limited to shader-level adjustments, this system enables complex volume operations—such as ramp-based remapping, multi-level filtering and resampling, and custom volume scripting—to be evaluated directly during rendering. This gives artists significantly greater control and flexibility when shaping the final look of combustion effects. Of particular interest are two render graph techniques we developed specifically to give our explosions a sharper, more "crusty" look, as shown in Figure 16. We refer to these methods as *Kora diffusion* and *Kora crust*, and their details are described below.

Kora diffusion. We apply a multi-level blur pass to the temperature channel and add it back with the original. This blends the hot combustion interior with the cooler surrounding air, lowering the temperature along the outer shell of the explosion and exaggerating the appearance of radiative cooling.

Kora crust. We blur the soot volume and subtract it from the original soot field. This isolates soot near convex regions of the explosion surface, where density remains relatively high after subtraction. The resulting layer darkens the outer shell of the explosion while allowing the emissive core to shine through the concave "cracks", producing the characteristic crusty appearance.

6 Production use

Kora was used extensively in the delivery of *Avatar: Fire and Ash*. In total, over 1000 shots were delivered containing individual Kora fire elements. These elements ranged from flamethrowers, fires and explosions, to a flux-vortex-driven fire tornado. Kora allowed a team of 100+ artists across 8 teams to create consistent and physical results while addressing notes from the client. Several sequences were especially impacted by Kora such as: The Ash Attack, Varang's Flamethrower, The Burning of the Metkayina Village, The Third Act Battle and many more. Alongside *Avatar: Fire and Ash*, Kora has been used on several more in-house productions.

The Ash Attack. The Ash Attack sequence is one of the film's major set pieces which provided significant technical challenges. The Sully family's convoy of trading ships is swamped and ambushed by the Mangkwan Clan, also known as the Ash people, led by their vicious leader Varang. They attack with burning arrows, fire bombs, and even a kamikaze targeting the gas filled medusoid which explodes. A defining characteristic of their armaments was designed to disperse flammable liquid upon impact, with the resulting splatter accelerating combustion on adjacent surfaces (Figure 5).

The fire arrows utilized a hybrid simulation approach designed to balance computational efficiency with physical accuracy across distinct phases of motion. During the ballistic flight phase where heavy motion blur reduced the need for fine detail, lightweight Houdini Pyro simulations provided a cost-effective solution. Upon surface contact, a FLIP fluid solver was introduced to accurately model the hydrodynamic behaviour of the accelerant, capturing the splashing and spreading dynamics inherent to liquid ignition. The resulting fluid simulation then served as the driving input for a high-fidelity Kora fire simulation, ensuring that the final combustion effect was grounded in physically plausible dynamics.

The Kora framework proved instrumental in maintaining look consistency and deterministic simulation outputs across the production pipeline. A key advantage of this approach is that the resolved simulation fields are passed directly to the render procedural with minimal post-simulation artistic intervention. This property facilitated the distribution of the Ash Attack sequence and other comparably complex sequences across multiple visual effects teams. Shared simulation templates ensured cross-team look continuity without redundant re-development of core effect parameters.

The Flux Devil. Working within the world of James Cameron typically means grounding every effect deeply in physics, and our pipeline and tooling are built around that expectation. The Flux Devil, however, sits at a delicate intersection of the physical and the

Figure 17: A Coriolis force term was incorporated into the momentum equation to impart a supernatural rotational component, resulting in a large-scale explosion being advected into a vortex structure. ©2025 20th Century Studios / Lightstorm Entertainment. All Rights Reserved.

fantastical: while the underlying phenomenon borrows from real physical principles, its scale, shape, and behaviour are ultimately a creative invention, guided as much by narrative and visual intent as by the laws that inspired it. Our approach was therefore to ground the simulation in real physics wherever possible, and to reserve deliberate, controlled departures for the places where the filmmakers' vision required them.

The story describes the phenomenon as driven by strong naturally occurring magnetic forces, producing an intricate double helix of plasma and water, generating a localized wind field as the displaced water pulls air towards the vortex. Any nearby metal is drawn into the structure, including the destroyed Sea Dragons left crippled by the Tulkun Elders. The sequence reaches its climax when the 700ft long Factoryship, felled by the Toruk, is pulled into the Flux Devil by magnetic attraction (Figure 17), spilling its liquid fuel across the ocean surface and feeding a fire tornado that rises a thousand feet into the air.

Our shot work spanned an unusually wide range of scales, from small firebursts passing close to camera to high aerial views looking down at a burning ocean surface, with the tornado twisting through the frame. Physical reference was difficult to source at the scales required, so we drew on a combination of amateur footage of fire vortices formed over steel drums containing fuel fires, and documentary footage of large forest fires, to inform the gross behaviour of the simulation. Those references gave us a physically plausible foundation, but the scale and choreography of the sequence still required controlled deviations from pure physics. To that end, warped gravity fields and art-directable Coriolis forces allowed the shape and rotation of the tornado to be tuned per shot while preserving the internal combustion behaviour that Kora was designed to produce, see Figures 1(right) and 12.

The Third Act Battle. The Third Act Battle sequence prioritized production efficiency through extensive library asset utilization. A new suite of Kora library assets were developed encompassing aerial explosions, flaming oil dispersal on water surfaces along with vehicular fire effects (Figure 18). As the sequence incorporates an eclipse event, all assets were engineered to function across both diurnal and nocturnal lighting conditions. The primary technical distinctions between configurations involved adjustments to the volumetric absorption and scattering coefficients of the smoke component, alongside targeted exposure corrections to preserve the high intensity luminous core characteristic of nighttime combustion. The resulting standardized asset library was then distributed across contributing visual effects teams, enforcing look consistency throughout the sequence.

Figure 18: Library fires, produced by Kora, placed in a shot context.

7 Lessons and limitations

While Kora has proven effective for the combustion effects required by *Avatar: Fire and Ash*, it is important to acknowledge several limitations encountered during production. Built on top of the Loki framework, Kora inherits the assumptions and constraints of a physically grounded solver, which can present challenges in certain scenarios. In particular, objects moving at very high velocities remain difficult to simulate accurately: resolving their motion demands temporal resolutions whose substep requirements quickly become prohibitive, making such shots impractical within typical turnaround times. More broadly, Kora performs best when supplied with inputs that align with the physical assumptions underlying its solvers. Production needs, however, often call for highly stylized effects featuring exaggerated velocities, stylized source geometry, or forces tuned for visual rather than physical plausibility. These cases place significant strain on physics-based components and can lead to heavy simulations with slow iteration cycles.

Another lesson concerns the use of fuel dissipation. While effective for shaping flame lifespan and size, it has frequently been overused as a corrective control despite its limited physical basis. Many such cases could be addressed more robustly through improved fuel sourcing design. Similarly, visually effective post-simulation processes—such as hollow flame masking, Kora diffusion and Kora crust—highlight opportunities to capture more of this behavior directly at simulation time.

User feedback has also identified usability concerns. Some parameters still remain overly complex for less experienced artists, and acceptable value ranges are not always clear, leading to reliance on institutional knowledge. Improving parameter discoverability and simplifying controls would further lower the barrier to entry. In addition, simulation runtimes can be long, especially for high-fidelity or heavily directed setups, limiting iteration speed despite extensive use of sparsity, adaptivity, and distributed execution.

We share these limitations not as shortcomings of the approach, but as honest observations about the trade-offs inherent in bringing a physically based combustion solver into a production pipeline, and in the hope that they inform both future development of Kora and related work by the wider community.

Acknowledgments

We wish to express our sincere gratitude to James Cameron for the extraordinary opportunity to contribute to the Avatar universe. The world of Pandora has provided an unparalleled canvas against which to develop, stress test, and advance the tools and techniques described in this paper. Work of this nature does not emerge in a vacuum; it is shaped profoundly by the vision and demands of the filmmaker driving it.

Thank you as well to our Technology, Research, and FX leadership for their support; to Ken Museth for the initial investigation and inspiration; to Rook Bridson and Michael Nielsen for insightful discussions; and to Sarah Coombs for help with the submission.

Many creative and dedicated individuals contributed their expertise to this project: Steve Lesser, Sean Flynn, Noh-hoon Lee, Joel Wretborn, and Markus Wabro — from Simulation; Chris George, Vincent Pegoraro, Tomáš Davidovič, and Marc Droske — from Rendering; Dániel Bukovec, James Robinson, Adrien Rollet, Anthony Arnoux, Rahul Deshprabhu, David Caeiro Cebrian, Florian Hu, Tanē MacDonald, Ivel Hernández Martínez, Tomohiro Okita, Jordan Cario, Yas Arahori, and Caitlin Prosser — from FX.

This work stands as a testament to the enduring belief that genuine innovation is achieved through the disciplined integration of scientific inquiry and artistic vision.

References

Gerardo Aguilera and John Johansson. 2019. Avengers: Endgame, a new approach for combustion simulations. In *ACM SIGGRAPH 2019 Talks* (Los Angeles, California) *(SIGGRAPH '19)*. Association for Computing Machinery, New York, NY, USA, Article 39, 2 pages. doi:10.1145/3306307.3328203

Ryoichi Ando and Christopher Batty. 2020. A practical octree liquid simulator with adaptive surface resolution. *ACM Trans. Graph.* 39, 4, Article 32 (Aug. 2020), 17 pages. doi:10.1145/3386569.3392460

Christopher Batty, Florence Bertails, and Robert Bridson. 2007. A fast variational framework for accurate solid-fluid coupling. In *ACM SIGGRAPH 2007 Papers* (San Diego, California) *(SIGGRAPH '07)*. Association for Computing Machinery, New York, NY, USA, 100–es. doi:10.1145/1275808.1276502

Robert Bridson. 2003. *Computational aspects of dynamic surfaces*. Ph. D. Dissertation. Stanford University, Stanford, CA, USA.

Robert Bridson, Jim Houriham, and Marcus Nordenstam. 2007. Curl-noise for procedural fluid flow. *ACM Trans. Graph.* 26, 3 (July 2007), 46–es. doi:10.1145/1276377.1276435

Charles K Chui. 1992. *An introduction to wavelets*. Vol. 1. Academic press.

Mihai Cioroba, Rick Hankins, Miguel Perez Senent, and Huai Yuan Teh. 2018. Star Wars: The Last Jedi - effects simulation: industrial light and magic. In *ACM SIGGRAPH 2018 Talks* (Vancouver, British Columbia, Canada) *(SIGGRAPH '18)*. Association for Computing Machinery, New York, NY, USA, Article 76, 2 pages. doi:10.1145/3214745.3214778

John Edholm, Alexey Stomakhin, Rahul Deshprabhu, David Caeiro Cebrian, Florian Hu, and Caitlin Pope. 2023. Fire and Explosions in Avatar: The Way of Water. In *ACM SIGGRAPH 2023 Talks* (Los Angeles, CA, USA) *(SIGGRAPH '23)*. Association for Computing Machinery, New York, NY, USA, Article 59, 2 pages. doi:10.1145/3587421.3595406

Luca Fascione, Johannes Hanika, Mark Leone, Marc Droske, Jorge Schwarzhaupt, Tomáš Davidovič, Andrea Weidlich, and Johannes Meng. 2018. Manuka: A Batch-Shading Architecture for Spectral Path Tracing in Movie Production. *ACM Trans. Graph.* 37, 3, Article 31 (aug 2018), 18 pages. doi:10.1145/3182161

Ronald Fedkiw, Jos Stam, and Henrik Wann Jensen. 2001. Visual simulation of smoke. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '01)*. Association for Computing Machinery, New York, NY, USA, 15–22. doi:10.1145/383259.383260

Bryan E. Feldman, James F. O'Brien, and Okan Arikan. 2003. Animating suspended particle explosions. *ACM Trans. Graph.* 22, 3 (July 2003), 708–715. doi:10.1145/882262.882336

Sean Flynn, Alexey Stomakhin, Joel Wretborn, and Daniel White. 2024. Art Directable Underwater Explosion Simulation. In *ACM SIGGRAPH 2024 Talks* (Denver, CO, USA) *(SIGGRAPH '24)*. Association for Computing Machinery, New York, NY, USA, Article 55, 2 pages. doi:10.1145/3641233.3664314

Zahra Forootaninia and Rahul Narain. 2020. Frequency-domain smoke guiding. *ACM Transactions on Graphics (TOG)* 39, 6 (2020), 1–10.

Oscar Gonzalez and Andrew M. Stuart. 2008. *A First Course in Continuum Mechanics*. Cambridge University Press. doi:10.1017/CBO9780511619571

Jeong-Mo Hong, Tamar Shinar, and Ronald Fedkiw. 2007. Wrinkled flames and cellular patterns. *ACM Trans. Graph.* 26, 3 (July 2007), 47–es. doi:10.1145/1276377.1276436

Christopher Horvath and Willi Geiger. 2009. Directable, high-resolution simulation of fire on the GPU. In *ACM SIGGRAPH 2009 Papers* (New Orleans, Louisiana) *(SIGGRAPH '09)*. Association for Computing Machinery, New York, NY, USA, Article 41, 8 pages. doi:10.1145/1576246.1531347

Insung Ihm, Byungkwon Kang, and Deukhyun Cha. 2004. Animation of reactive gaseous fluids through chemical kinetics. In *Proceedings of the 2004 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Grenoble, France) *(SCA '04)*. Eurographics Association, Goslar, DEU, 203–212. doi:10.1145/1028523.1028550

Masato Ishimuroya and Takashi Kanai. 2016. Adding visual details based on low-resolution energy cascade ratios for smoke simulation. In *ACM SIGGRAPH 2016 Posters* (Anaheim, California) *(SIGGRAPH '16)*. Association for Computing Machinery, New York, NY, USA, Article 16, 2 pages. doi:10.1145/2945078.2945094

Byungkwon Kang, Yoojin Jang, and Insung Ihm. 2007. Animation of chemically reactive fluids using a hybrid simulation method. In *Proceedings of the 2007 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (San Diego, California) *(SCA '07)*. Eurographics Association, Goslar, DEU, 199–208.

ByungMoon Kim, Yingjie Liu, Ignacio Llamas, and Jarek Rossignac. 2005. FlowFixer: using BFECC for fluid simulation. In *Proceedings of the First Eurographics Conference on Natural Phenomena* (Dublin, Ireland) *(NPH'05)*. Eurographics Association, Goslar, DEU, 51–56.

Theodore Kim, Nils Thürey, Doug James, and Markus Gross. 2008. Wavelet turbulence for fluid simulation. *ACM Trans. Graph.* 27, 3 (Aug. 2008), 1–6. doi:10.1145/1360612.1360649

Nipun Kwatra, Jón T. Grétarsson, and Ronald Fedkiw. 2010. Practical animation of compressible flow for shock waves and related phenomena. In *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Madrid, Spain) *(SCA '10)*. Eurographics Association, Goslar, DEU, 207–215.

Steve Lesser, Alexey Stomakhin, Gilles Daviet, Joel Wretborn, John Edholm, Noh-Hoon Lee, Eston Schweickart, Xiao Zhai, Sean Flynn, and Andrew Moffat. 2022. Loki: A Unified Multiphysics Simulation Framework for Production. *ACM Trans. Graph.* 41, 4, Article 50 (jul 2022), 20 pages. doi:10.1145/3528223.3530058

Frank Losasso, Geoffrey Irving, Eran Guendelman, and Ron Fedkiw. 2006. Melting and Burning Solids into Liquids and Gases. *IEEE Transactions on Visualization and Computer Graphics* 12, 3 (May 2006), 343–352. doi:10.1109/TVCG.2006.51

Olivier Maury, Dan Piponi, Florent Andorra, and Craig Hammack. 2010. Bending Fire with Plume, a CUDA Based 3D Fluid Solver and Volume Renderer. In *ACM SIGGRAPH 2010 Talks* (Los Angeles, California) *(SIGGRAPH '10)*. Association for Computing Machinery, New York, NY, USA.

Zeki Melek and John Keyser. 2002. Interactive Simulation of Fire. In *Proceedings of the 10th Pacific Conference on Computer Graphics and Applications (PG '02)*. IEEE Computer Society, USA, 431.

Duc Quang Nguyen, Ronald Fedkiw, and Henrik Wann Jensen. 2002. Physically based modeling and animation of fire. *ACM Trans. Graph.* 21, 3 (July 2002), 721–728. doi:10.1145/566654.566643

Michael B. Nielsen, Morten Bojsen-Hansen, Konstantinos Stamatelos, and Robert Bridson. 2022. Physics-Based Combustion Simulation. *ACM Trans. Graph.* 41, 5, Article 176 (may 2022), 21 pages. doi:10.1145/3526213

Michael B. Nielsen, Konstantinos Stamatelos, Morten Bojsen-Hansen, and Robert Bridson. 2019. Physics-based combustion simulation in bifrost. In *ACM SIGGRAPH 2019 Talks* (Los Angeles, California) *(SIGGRAPH '19)*. Association for Computing Machinery, New York, NY, USA, Article 40, 2 pages. doi:10.1145/3306307.3328149

NIST. 2025. *Chemistry WebBook*. doi:10.18434/T4D303

Andrew Selle, Ronald Fedkiw, Byungmoon Kim, Yingjie Liu, and Jarek Rossignac. 2008. An Unconditionally Stable MacCormack Method. *J. Sci. Comput.* 35, 2–3 (June 2008), 350–371. doi:10.1007/s10915-007-9166-4

Jason Sewall, Nico Galoppo, Georgi Tsankov, and Ming Lin. 2009. Visual simulation of shockwaves. *Graph. Models* 71, 4 (July 2009), 126–138. doi:10.1016/j.gmod.2009.03.002

Jos Stam and Eugene Fiume. 1995. Depicting fire and other gaseous phenomena using diffusion processes. In *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '95)*. Association for Computing Machinery, New York, NY, USA, 129–136. doi:10.1145/218380.218430

Alexey Stomakhin, Steve Lesser, Joel Wretborn, Sean Flynn, Johnathan Nixon, Nicholas Illingworth, Adrien Rollet, Kevin Blom, and Douglas Mchale. 2023. Pahi: A Unified Water Pipeline and Toolset. In *Proceedings of the Digital Production Symposium* (Los Angeles, CA, USA) *(DigiPro '23)*. Association for Computing Machinery, New York, NY, USA, Article 11, 13 pages. doi:10.1145/3603521.3604291

Alexey Stomakhin, Craig Schroeder, Lawrence Chai, Joseph Teran, and Andrew Selle. 2013. A material point method for snow simulation. *ACM Trans. Graph.* 32, 4, Article 102 (July 2013), 10 pages. doi:10.1145/2461912.2461948

Alexey Stomakhin and Andrew Selle. 2017. Fluxed Animated Boundary Method. *ACM Trans. Graph.* 36, 4, Article 68 (jul 2017), 8 pages. doi:10.1145/3072959.3073597

Joel Wretborn, Marcus Schoo, Noh-Hoon Lee, Christopher Batty, and Alexey Stomakhin. 2026. A practical partitioner for distributed simulations on sparse dynamic domains using optimal transport. *ACM Trans. Graph.* 3787521 (Jan. 2026).

Gary D. Yngve, James F. O'Brien, and Jessica K. Hodgins. 2000. Animating explosions. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '00)*. ACM Press/Addison-Wesley Publishing Co., USA, 29–36. doi:10.1145/344779.344801

Jonas Zehnder, Rahul Narain, and Bernhard Thomaszewski. 2018. An advection-reflection solver for detail-preserving fluid simulation. *ACM Trans. Graph.* 37, 4, Article 85 (July 2018), 8 pages. doi:10.1145/3197517.3201324