Real-Time Physics-Based Visualization of FDS/WFDS Simulations in Unreal Engine 5 Using Sparse Volume Textures

David Caballero*[1]

[1]*MeteoGrid. Modesto Lafuente 45, 28010, Madrid, Spain*[2]

**Corresponding author. Email: david@meteogrid.com*

Keywords

Wildland fire simulation, FDS/WFDS, OpenVDB, Real-time volumetric rendering, Sparse Volume Texture

Abstract

This paper presents a novel end-to-end pipeline for the physics-based, real-time visualization of wildland fire simulations performed with FDS/WFDS in Unreal Engine 5 (UE5). The workflow converts volumetric SMOKE3D output from FDS into OpenVDB sequences on the same HPC cluster where the simulations are executed, exploiting parallel processing to reduce conversion times to minutes even for large multi-mesh domains. The resulting VDB files are imported into UE5 as Sparse Volume Textures (SVT) and rendered in real time through a custom volumetric ray marching material that maps three physically meaningful FDS channels — soot density, gas temperature, and water vapour mass fraction — to optically distinct rendering behaviours: light extinction, blackbody thermal emission, and single-scattering for white smoke, respectively. A standardized normalization scheme with fixed physical reference values enables direct cross-case comparison. The pipeline is validated on three FDS/WFDS test cases of increasing complexity executed on the MeluXina and LUMI EuroHPC supercomputers within the HIDALGO2 project. The compatibility of SVT-based volumetric rendering with Neural Radiance Fields (NeRFs) scene reconstruction is demonstrated; showing that mutual occlusion between synthetic fire volumes and NeRF-reconstructed real-world environments is correctly handled, unlike Gaussian Splatting representations. To our knowledge, this is the first published workflow connecting physics-based wildland fire simulation output to real-time volumetric rendering in a game-engine environment.

PREPRINT

1. Introduction

The physical, chemical and fluid-dynamic phenomena involved in wildland fires are notoriously difficult to communicate to firefighters, incident commanders, policymakers and the general public. High-fidelity physics-based simulation with tools such as the Fire Dynamics Simulator (FDS) (McGrattan et al., 2023) and its wildland-urban interface extension (WFDS) (Mell et al., 2007, 2009, 2011) can capture these phenomena with high accuracy, but the standard post-processing environment, Smokeview, offers limited visualization capabilities that are not suited for high-quality rendering, training simulations or real-time decision support.

Existing approaches to visualizing fire in interactive environments typically rely on billboard sprites onto which video clips of flames or smoke are projected (Latta & Kolb, 2004; Fuller et al., 2007), or on particle systems driving animated billboards. These methods are computationally inexpensive but optically incorrect: the smoke produced is not volumetric, the interaction with scene lighting is not physically modeled, and no connection to simulation data exists. More advanced game-engine implementations use physically-based volumetric materials (Fuller et al., 2007; Crane et al., 2007), but these are driven by procedural noise or artist-authored fields, not by actual simulation results.

The approach proposed here is fundamentally different. We use FDS/WFDS as the source of the volumetric data — soot density, gas temperature and water vapor mass fraction — and feed these fields directly into the real-time volumetric renderer of Unreal Engine 5 (UE5), a frontline game development platform increasingly used for scientific visualization (Caballero, 2024). The result is a visualization that is both physically grounded and visually convincing, and that can be produced for any FDS/WFDS case with no manual artistic intervention.

The broader research context is a complete pipeline connecting procedurally generated vegetation in UE5 (via Procedural Content Generation, PCG), exported as per-facet point clouds with physical fuel attributes, to FDS/WFDS simulations on EuroHPC supercomputers, and back to UE5 for visualization (Caballero, 2026).

This paper focuses exclusively on the downstream visualization step: the conversion of FDS volumetric output to OpenVDB and its real-time rendering in UE5 as Sparse Volume Textures.

2. Justification and objectives

The need for high-fidelity visualization of wildland fire simulations arises in at least three distinct operational contexts. In firefighter training, realistic three-dimensional representations of fire behavior in complex terrain and vegetation are far more effective than two-dimensional diagrams or low-fidelity animations. In public risk communication, the ability to show how a fire would behave under specific weather and fuel conditions in a recognizable landscape is a powerful tool for evacuations and prescribed burn acceptance. In scientific dissemination, interactive visualization of simulation results allows researchers to explore and communicate complex spatio-temporal dynamics that are impossible to convey with static figures.

Smokeview, the standard FDS visualization tool, meets none of these needs. Its volumetric rendering is functional for scientific inspection but produces outputs that are unsuitable for training, communication or dissemination purposes: the visual quality is low, lighting is pre-computed and static, and there is no integration with real-world scene data.

The intermediate format chosen to bridge FDS output and UE5 is VDB, implemented with the library OpenVDB, an open-source hierarchical data structure and toolkit for the representation of high-resolution sparse volumetric data, introduced by Museth et al. (2013) and now an Academy Software Foundation standard widely adopted in visual effects and scientific visualization. OpenVDB organizes voxel data in a B+tree-like hierarchy of nodes with adaptive resolution, storing only the active (non-background) voxels and their spatial topology.

This sparse representation is particularly well suited to wildland fire simulation output: the smoke, flame and hot gas occupy only a small fraction of the total computational domain at any given time step, and a dense volumetric array would waste the majority of its storage on background cells. In practice, OpenVDB file sizes for the cases presented here range from 3 MB per frame at early ignition — when only a small combustion zone is active — to 15 MB at peak fire spread, compared to the tens of gigabytes that a dense *float32* array would require for the same domain. This compression makes it practical to transfer complete simulation sequences from HPC clusters to local workstations via standard file transfer protocols in minutes rather than hours, and to store multiple simulation scenarios simultaneously in GPU memory as Sparse Volume Textures in UE5. Each VDB file in the sequence carries three named *FloatGrid* objects — density, temperature and scatter — together with a linear spatial transform encoding the physical voxel size and domain origin, so that the correct metric geometry of the simulation domain is preserved through the entire pipeline without manual intervention.

Game engines such as UE5 provide the required rendering quality and real-time performance, and their volumetric rendering systems — in particular Sparse Volume Textures (SVT) and the Heterogeneous Volume (HV) actors — are capable of handling the scale and resolution typical of FDS output. The missing piece is a conversion pipeline connecting FDS binary output to the VDB format natively consumed by UE5, and a physically motivated material model that maps FDS physical quantities to optically meaningful rendering parameters.

The specific objectives of this work are: (1) to define the FDS output configuration required to export the necessary volumetric quantities; (2) to develop and validate a parallel Python converter from FDS SMOKE3D binary format to OpenVDB; (3) to establish a standardized normalization scheme that enables cross-case comparison; (4) to design a physically motivated volumetric ray marching material for UE5; and (5) to demonstrate the compatibility of this approach with Neural Radiance Field (NeRF) scene reconstruction.

PREPRINT

3. Material and methods

3.1. FDS/WFDS simulation configuration

FDS 6.10.1, compiled from the master branch with OpenMPI 5.0.7 and GCC 14.2.0, was used for all simulations in MeluXina and LUMI HPC clusters. When chemical reactions are active (N_REACTIONS > 0), FDS automatically reserves three SMOKE3D channels regardless of user declarations: channel 1 (SOOT DENSITY), channel 2 (HRRPUV), and channel 3 (EFFECTIVE FLAME TEMPERATURE). User-defined channels are appended sequentially after these three. Two additional channels are declared in the FDS input file to obtain the quantities required for visualization:

```
&SM3D QUANTITY='TEMPERATURE' /                              ! Channel 4
&SM3D QUANTITY='MASS FRACTION', SPEC_ID='WATER VAPOR' /     ! Channel 5
```

The WATER VAPOR species is already declared as an individually tracked species by the WFDS vegetation combustion model (*vegetation_model.txt*), which uses the chemistry C=2.10, H=6.20, O=2.16, HOC=17425 kJ/kg. No additional &SPEC declaration is required in the main input file.

The SMOKE3D binary format stores volumetric data as FORTRAN unformatted sequential records, using Run-Length Encoding (RLE) for compression. Decoding was implemented by reading the FDS source directly (smvv.f90, SUBROUTINE SMOKE3D_TO_FILE and SUBROUTINE RLE). Two file types exist for channels with DISPLAY_TYPE=='GAS' (soot and water vapour): the *.s3d* file stores the extinction-optimized (logarithmically scaled) representation used by Smokeview, while the *.s3dd* file stores the linearly scaled density field. For temperature, only the *.s3d* file is written, with linear scaling over the fixed FDS range [0, 2500°C]. The accompanying *.s3d.sz* metadata file records, for each frame: simulation time, uncompressed size, maximum optical value (MAX_VAL), and — for GAS channels — maximum physical density value (MAX_DENSITY_VAL, column 6). It is this last quantity that is used to dequantize the .s3dd data into physical units.

Table 1- SMOKE3D channel mapping and file types for the FDS output used in this pipeline.

Channel	Quantity	Data file	Dequantization
1	SOOT DENSITY (kg/m³)	_1.s3dd	byte / 254 × MAX_DENSITY_VAL
2	HRRPUV (kW/m³)	_2.s3d	Not used in VDB export
3	Effective flame temperature	_3.s3d	Not used in VDB export
4	TEMPERATURE (°C)	_4.s3d	byte / 254 × 2500 °C
5	WATER VAPOR mass fraction	_5.s3dd	byte / 254 × MAX_DENSITY_VAL

3.2. Parallel OpenVDB conversion

The converter *fds2vdb_parallel.py* was developed to handle multi-mesh FDS cases of arbitrary size and mesh arrangement. It operates as follows. First, the Smokeview *.smv* file is parsed to recover the physical extents of each mesh (PDIM entries), from which the global domain dimensions and mesh arrangement (NI × NJ × NK) are determined automatically. Second, for each simulation frame, the .s3d.sz metadata file is read to obtain the per-frame maximum density values and frame timestamps. Third, the RLE-compressed binary data for each mesh and each of the three channels of interest (channels 1, 4 and 5) are decoded and dequantized into physical units. Fourth, the per-mesh arrays are assembled into the global volumetric array, with shared boundary nodes overwritten by the neighboring mesh (physically correct, as they represent the same spatial location). Fifth, the three assembled volumes are normalized to [0, 1] using fixed reference values and exported as named OpenVDB *FloatGrid* objects.

Parallel processing of frames is implemented with Python's *multiprocessing.Pool*, with one worker process per available CPU core. This approach scales linearly with core count. On a single 128-core node of

PREPRINT

MeluXina (AMD EPYC 7H12), a 240×240×20 m domain with 144 meshes and 971 frames is converted in under 4 minutes. File sizes range from approximately 3 MB per frame for sparse early-ignition frames to 15 MB at peak fire spread, reflecting the sparse storage advantage of OpenVDB.

OpenVDB 12.0.0 was compiled from source on both MeluXina and LUMI supercomputers, with Python bindings (*openvdb.cpython*). Three named *FloatGrid* objects are written per output frame: density (soot), temperature, and scatter (water vapor). Each grid carries a linear spatial transform encoding the physical voxel size and domain origin, ensuring correct metric placement when the VDB sequence is imported into UE5.

3.3. Standardized normalization scheme

UE5 SVT assets operate on values in the [0, 1] range. A consistent normalization scheme is essential to allow a single rendering material to be used across simulation cases without modification, and to enable direct visual comparison between fire scenarios. The following fixed reference values are suggested for all cases:

Table 2- Suggested normalization values of the selected physical quantities for the generation of VDB files.

Channel	Physical quantity	Reference max	Basis
density	Soot density	0.001 kg/m³	Empirical calibration, optically dense soot
temperature	Gas temperature	1800 °C	Near-maximum flame temperature, WFDS model
scatter	H_2O mass fraction	0.10 kg/kg	Stoichiometric max, WFDS vegetation combustion

Values exceeding the reference are clipped to 1.0. The ambient water vapor background (~0.07 normalized units) present throughout the computational domain is subtracted before normalization, yielding a residual field wv_excess = max(wv − 0.07, 0) that isolates combustion-generated water vapor from ambient humidity. This prevents the ambient background from contributing spurious white-smoke scattering across the entire domain.

3.4. UE5 import and SVT configuration

The VDB frame sequence is imported into UE5 via the native VDB importer, which maps named grid names to SVT attribute channels. The three grids (density, temperature, scatter) are assigned to SVT attribute slots as follows: density → Attribute A / channel R; temperature → Attribute B / channel R; scatter → Attribute B / channel G. SVT attributes are stored as 16-bit float, which provides 65536 quantization levels — sufficient to avoid banding artifacts in smooth smoke gradients while maintaining acceptable memory footprint for multi-hundred-frame sequences.

The VDB spatial transform encodes the physical domain origin and voxel size in meters, but several coordinate-system adjustments are required when placing the HV actor in UE5. First, since FDS uses meters as its base unit and UE5 uses centimeters, a uniform scale factor of 100 must be applied to the actor. Second, FDS follows a right-handed coordinate convention whereas UE5 uses a left-handed system; a sign flip on the appropriate axis is therefore necessary to obtain the correct spatial orientation of the volume. Third, the pivot point of the HV actor is referenced to the centroid of the occupied voxel bounding box of the last frame in the sequence rather than to the domain origin, so a manual position offset must be applied to align the volume with the correct world-space location in the scene.

3.5. Volumetric ray marching material

A custom volumetric material was developed in the UE5 Material Editor for use with the Heterogeneous Volume actor. The material reads two SVT channels — soot density and gas temperature — and maps them to distinct optical behaviors. The soot density channel serves a dual role: it drives the extinction coefficient, producing darker and more opaque smoke as density increases, and it simultaneously drives the single-scattering albedo, modulating the proportion of incident light that is scattered rather than absorbed. This dual use of a single physical field is physically justified because soot particles both absorb and scatter light, and

PREPRINT

because the spatial distribution of soot is a reliable proxy for the overall smoke column regardless of whether the dominant optical regime is absorption or scattering. The gas temperature channel drives self-emission through a UE5 Blackbody material node: the normalized temperature value is remapped to the physical range [0, 1800 °C] and passed to the Blackbody node, which returns the spectral color of a perfect emitter at that temperature, producing the characteristic progression from dark red (~500 °C) through orange and yellow to near-white (~1500 °C).

Water vapor mass fraction (channel 5) was exported and evaluated as a candidate for a dedicated scattering channel, with the physical motivation that combustion-generated H_2O is most concentrated above the flame zone where gas temperatures fall below the condensation threshold, precisely the region where condensed droplets and organic aerosols produce the white smoke characteristic of vegetation fires. In practice, however, physically correct use of this field as a scattering driver requires cross-referencing with the temperature channel to estimate where condensation actually occurs, since water vapor remains optically inactive above the condensation threshold. This temperature-dependent gating adds substantial complexity to the material graph and introduces calibration parameters — condensation temperature, aerosol scattering coefficient — that are not directly available from the FDS output. Comparative rendering tests showed that the visual gain over the single-channel soot approach was marginal for the fire scenarios evaluated, while the material complexity increased significantly. The water vapor channel was therefore retained in the VDB export for completeness and future use, but is not used in the current rendering material.

The material uses UE5's hardware-accelerated single-scattering ray marching with a single directional light source. The Heterogeneous Volume renderer does not support indirect illumination or skylight, which produces a persistent shadowed zone on the unlit side of the smoke column. This is a known limitation of the current UE5 rendering architecture for heterogeneous volumes and is documented as such.

4. Results

4.1. Test cases

Three FDS/WFDS cases of increasing domain complexity were used to validate the pipeline. Table 2 summarizes their key parameters.

Table 3- FDS/WFDS test cases used to validate the pipeline.

Case	Domain (m)	Cell size (m)	Meshes	Frames	HPC cluster
Douglas fir tree (2 m)	2.4×2.4×9.6	0.10	8	977	MeluXina
T- hedge (2 m)	8×6×5	0.05	240	801	MeluXina / LUMI
CSIRO Case F19 (grassland)	240×240×20	0.50	144	171	MeluXina

The Douglas fir tree case uses the WFDS Lagrangian particle model with four fuel classes (foliage, small, medium and large roundwood) and a ring ignitor of spherical particles with HRRPUA = 60 kW/m². The T-hedge case extends the same approach to a 2-meter-tall conifer T-shaped hedge with 240 MPI meshes at 5 cm resolution. The CSIRO Case F19 is a standard validation case for grassland fire spread against Australian field experimental data.

PREPRINT

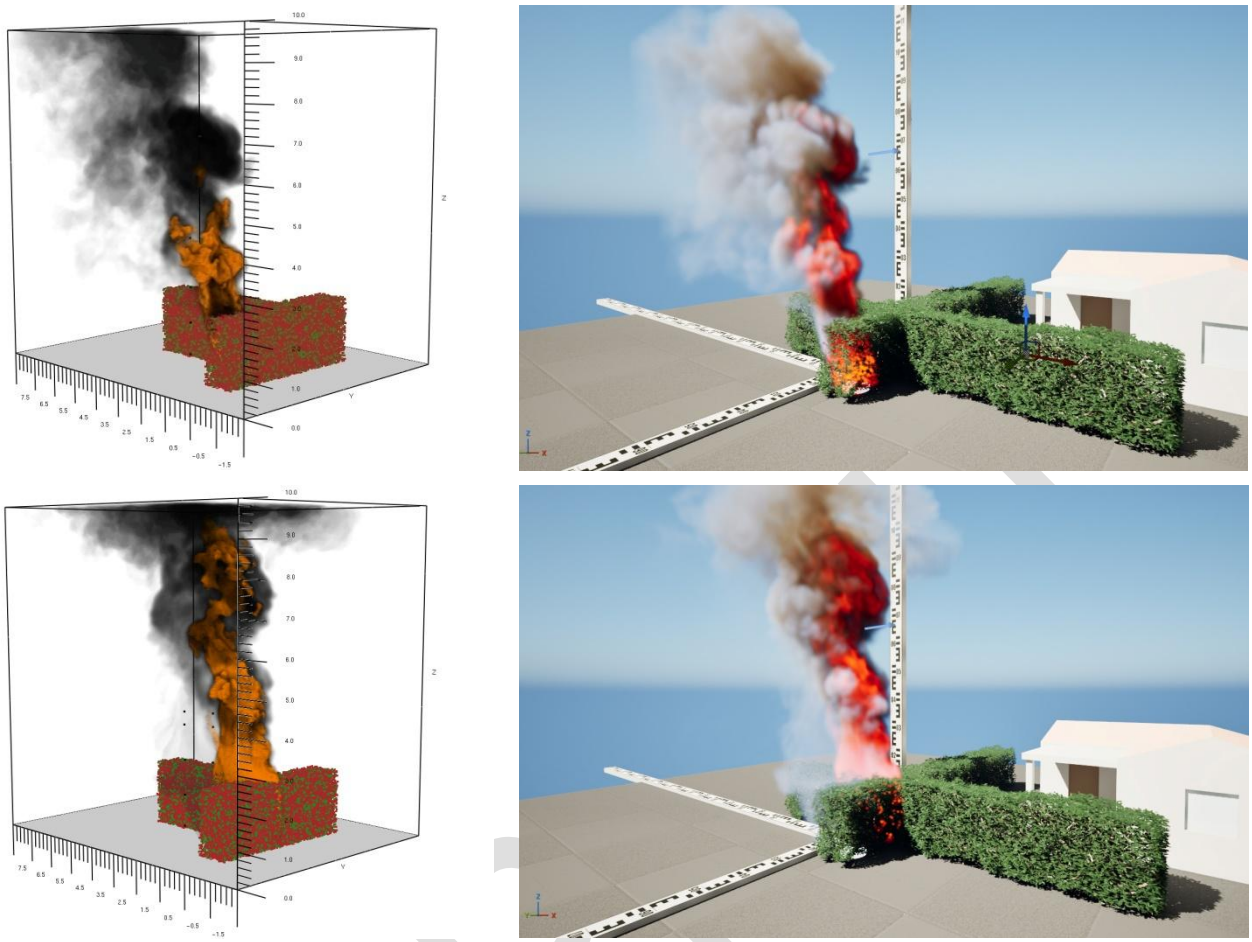


Figure 1 – T-hedge case rendered in Smokeview (left) and UE5 (right), at frames 190 (up) and 800 (bottom). The horizontal extent of the computational domain (8 × 6 m) is correctly reproduced at physical scale. The white scattering zone near ground level corresponds to the water vapour excess field; the dark absorbing region above corresponds to the soot density field. Flame region is self-emissive.



Figure 2 – CSIRO Case F19 grassland fire experiment rendered in UE5. The 240×240×20 m domain comprises 144 MPI meshes at 0.5 m resolution. The fire front propagation and associated smoke column are visible.

4.2. Pipeline performance

Conversion throughput on a single 128-core MeluXina CPU node (AMD EPYC 7H12, 32 worker processes allocated by SLURM) is summarized in Table 3. Transfer times from HPC to local workstation via FTP (WinSCP) are also reported; these are typically the dominant wall-clock cost in the workflow.

Table 4- Pipeline performance metrics. Conversion performed on a single MeluXina CPU node with 32 worker processes.

Case	Conversion time (min)	VDB size GB	FTP Transfer FTP time (min)
Douglas fir tree (977 frames)	< 3 min	~4 GB	~8 min
T- hedge (801 frames)	< 4 min	~7 GB	~14 min
CSIRO Case F19 (171 frames)	< 1 min	~1.2 GB	~3 min

5. Neural Radiance Field and Gaussian Splatting integration

Neural Radiance Fields (NeRF) (Mildenhall, 2020) and Gaussian Splatting (GS) (Kerbl et al., 2023) are two common methods for photogrammetric 3D reconstruction of real-world scenes from multi-view photograph collections. Both represent the scene as a volumetric or point-based radiance field that can be rendered in novel views. Their compatibility with physics-based SVT fire volumes is relevant for applications where simulated fire must be composited over a real reconstructed environment.

Because both SVT and NeRF volumes are rendered through ray marching, they can share the same depth buffer and correctly handle mutual occlusion: scene geometry reconstructed by NeRF occludes fire volumes behind it, and dense fire smoke occludes NeRF geometry behind the smoke column. This compositing is physically correct and was verified in UE5 using a NeRF scene reconstructed from drone imagery.

Gaussian Splatting, by contrast, represents the scene as a collection of semi-transparent 3D Gaussians sorted and alpha-composited in screen space. This representation is fundamentally incompatible with ray marched volumetric rendering: depth ordering between Gaussian splats and volumetric smoke cannot be resolved correctly in screen space, producing visible compositing artifacts. For applications requiring fire visualization over real-world reconstructed scenes, NeRF is therefore the preferred reconstruction method.

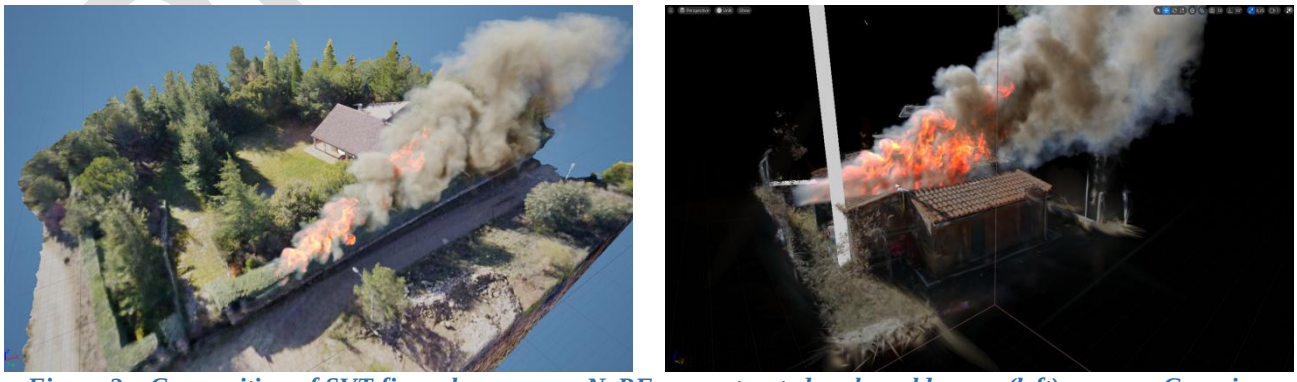


Figure 3 – Compositing of SVT fire volume over a NeRF-reconstructed real-world scene (left) versus a Gaussian Splatting reconstruction (right). Mutual occlusion is physically correct with NeRF; depth-ordering artefacts are visible with Gaussian Splatting).

6. Discussion

PREPRINT

The principal novelty of this work is the end-to-end connection between physics-based wildland fire simulation and real-time volumetric rendering in a game-engine environment. Each design decision in the pipeline is grounded in physical reasoning. Soot density drives both the extinction coefficient and the scattering albedo, serving as the sole proxy for visible smoke: it captures the spatial extent and optical density of the smoke column in both its absorbing and scattering regimes, and produces visually convincing results across all three test cases without requiring additional fields or temperature-dependent modulation. Gas temperature drives thermal emission through a physically correct blackbody spectrum, producing the characteristic color progression of the flame zone.

The standardized normalization scheme is a deliberate choice to prioritize scientific comparability over per-case visual optimization. Dynamic normalization — scaling each frame or each case to its own maximum — would produce visually saturated results in all cases but would make inter-case comparison meaningless. The fixed reference values chosen here (0.001 kg/m³ for soot, 1800 °C for temperature, 0.10 kg/kg for water vapor) are grounded in the combustion stoichiometry of the WFDS model and empirically validated against the test cases: no case exceeds these limits in any frame, confirming that the reference values correctly bracket the physical range without premature saturation.

Water vapor mass fraction was evaluated as a candidate for a dedicated scattering channel, with the physical motivation that combustion-generated H_2O is most concentrated above the flame zone where gas temperatures fall below the condensation threshold. In practice, physically correct use of this field as a scattering driver requires cross-referencing with the temperature channel to estimate where condensation actually occurs, since water vapor remains optically inactive above the condensation threshold. This temperature-dependent gating introduces calibration parameters — condensation temperature, aerosol scattering coefficient — that are not directly available from the FDS output, and adds substantial complexity to the material graph. Comparative rendering tests showed that the visual gain over the single-channel soot approach was marginal for the scenarios evaluated. The water vapor channel is therefore retained in the VDB export for completeness and future use, but is not used in the current rendering material.

The incompatibility of Gaussian Splatting with SVT ray marching is a fundamental architectural mismatch, not a calibration issue: Gaussian Splatting performs alpha compositing in screen space, which cannot respect the depth relationships between point-based representations and volumetric grids that span the same spatial region. NeRF, as a volumetric method, does not have this limitation. This finding has practical implications for WUI fire visualization applications, where the fire simulation domain must be composited over photogrammetric reconstructions of real buildings, roads and vegetation.

The current implementation has two known limitations. First, the UE5 Heterogeneous Volume renderer supports only single-scattering with a single directional light; there is no support for indirect illumination or skylight. This produces a persistent shadow zone on the unlit side of the smoke column that is not physically realistic for diffuse-sky conditions. This is a limitation of the UE5 rendering architecture at the time of writing, not of the conversion pipeline. Second, using soot density as the sole scattering proxy does not distinguish between the optically dark behavior of heavily loaded soot and the white scattering behavior of condensed water droplets and organic aerosols: both phenomena are represented by the same field, and their separation requires either aerosol-specific output from FDS or a more sophisticated material model. This is identified as the primary avenue for future improvement.

7. Conclusions

A complete, open and validated pipeline has been presented for the physics-based real-time visualization of FDS/WFDS wildland fire simulations in Unreal Engine 5. The key contributions are:

- Identification of the correct FDS namelist (&SM3D) for user-defined SMOKE3D channels, and the exact decoding procedure for the *.s3dd* density companion files, derived directly from the FDS Fortran source code.
- A parallel Python converter (*fds2vdb_parallel.py*) that handles multi-mesh FDS cases of arbitrary geometry, with throughput sufficient to convert hundreds of frames in minutes on a single HPC node.

PREPRINT

- A physically motivated two-channel rendering strategy in which soot density serves as the sole proxy for visible smoke — driving both light extinction and single-scattering albedo — while gas temperature drives thermal emission through a blackbody material node. Water vapor mass fraction is exported and retained for future use but was found to require prohibitive material complexity to use correctly as a scattering driver.
- A standardized cross-case normalization scheme with fixed physical reference values grounded in the WFDS vegetation combustion stoichiometry, enabling direct visual comparison between fire scenarios without modifying the rendering material.
- A demonstration that NeRF-based scene reconstruction is compatible with SVT volumetric rendering for composited WUI fire visualization, correctly handling mutual occlusion, whereas Gaussian Splatting is not due to a fundamental screen-space compositing incompatibility.

The pipeline is immediately applicable to any FDS/WFDS case using the WFDS vegetation combustion model, and its normalization reference values are transferable to other fuel types with minor calibration. The conversion scripts are open and will be made available for community use.

8. Acknowledgements

This work was conducted within the HIDALGO2 project (EuroHPC JU). Computing resources were provided by the MeluXina supercomputer (LuxProvide, Luxembourg) and the LUMI supercomputer (CSC, Finland).

9. References

Alexander, M.E., Cruz, M.G. (2013) Limitations on the accuracy of model predictions of wildland fire behaviour: a state-of-the-knowledge overview. Forestry Chronicle, 89(3), 372–383.

Caballero, D. (2024) A method for importing and drawing SHP files in Unreal Engine. Technical Notes Series (preprint). DOI: 10.13140/RG.2.2.26576.48648/1

Caballero, D. (2026) From 3D vegetation models to physics-based wildfire simulation: a fractal barycentric method for Lagrangian particle initialization in FDS/WFDS. Advances on Forest Fire Research – 2026 D. X. Viegas & L.M. Ribeiro (Ed.) In press.

Cheney, N.P., Gould, J.S., Catchpole, W.R. (1998). Prediction of fire spread in grasslands. International Journal of Wildland Fire, 8(1), 1–13.

Crane, K., Llamas, I., Tariq, S. (2007) Real-Time Simulation and Rendering of 3D Fluids. In: Nguyen, H. (ed.) GPU Gems 3, Chapter 30, pp. 633–676. Addison-Wesley Professional.

Fuller, A.R., Krishnan, H., Mahrous, K., Hamann, B., Joy, K.I. (2007). Real-time procedural volumetric fire. Proceedings of the 2007 Symposium on Interactive 3D Graphics and Games (I3D 2007), ACM, pp. 175–180.

Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G. (2023). 3D Gaussian Splatting for real-time radiance field rendering. ACM Transactions on Graphics, 42(4).

Latta, L., Kolb, A. (2004). Homogeneous Bubbles for Fast and Robust Fire Rendering. In: Fernando, R. (ed.) GPU Gems: Programming Techniques, Tips and Tricks for Real-Time Graphics, Chapter 6. NVIDIA / Addison-Wesley.

McGrattan, K., Hostikka, S., McDermott, R., Floyd, J., Weinschenk, C., Overholt, K. (2023). Fire Dynamics Simulator User's Guide, NIST Special Publication 1019, 7th edition. National Institute of Standards and Technology, Gaithersburg, MD.

Mell, W., Jenkins, M.A., Gould, J., Cheney, P. (2007). A physics-based approach to modelling grassland fires. International Journal of Wildland Fire, 16(1), 1–22.

PREPRINT

Mell, W., Maranghides, A., McDermott, R., Manzello, S.L. (2009). Numerical simulation and experiments of burning Douglas fir trees. Combustion and Flame, 156(10), 2023–2041.

Mell, W., McNamara, D., Maranghides, A., McDermott, R., Forney, G., Hoffman, C., Ginder, M. (2011). Computer modelling of wildland-urban interface fires. Fire and Materials Conference, San Francisco, CA.
Museth, K. (2013). VDB: High-resolution sparse volumes with dynamic topology. ACM Transactions on Graphics, 32(3), 1–22.

Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R. (2020). NeRF: Representing scenes as neural radiance fields for view synthesis. Proceedings of ECCV 2020, Lecture Notes in Computer Science, vol. 12346.

Museth, K. 2013. VDB: High-resolution sparse volumes with dynamic topology. ACM Trans. Graph. 32, 3, Article 27 (June 2013) 22 pages.

Museth, K., Lait, J., Johansson, J., Nafis, D., Pacanowski, R., Hutchins, D., Bailey, J., Bogart, D., Crawford, D., Gentry, R., Museth, K., Miller, G. (2013). OpenVDB: An open-source data structure and toolkit for high-resolution volumes. SIGGRAPH 2013 Course Notes.

PREPRINT