A Physics-Based Forest Fire Simulation using Semantic Gaussian Splatting

Nienke Driessen

Delft University of Technology

TU Delft

A Physics-Based Forest Fire Simulation using Semantic Gaussian Splatting

by

Nienke Driessen

Nienke
Driessen

Thesis advisor:	Michael Weinmann
Daily co-supervisor:	Joris Rijsdijk
External Advisor:	Dominik L. Michels
Project Duration:	September, 2025 - April, 2026
Faculty:	Faculty of Computer Science and Engineering, Delft

Cover:	Forest fire simulation render created by Nienke Driessen
Style:	TU Delft Report Style, with modifications by Nienke Driessen

TU Delft

Preface

Firstly, I would like to thank my primary supervisor, Michael Weinmann, for his feedback and guidance throughout this project. And thank you for letting me use the computer remotely; I certainly wouldn't have gotten this far without it. I would also like to thank Joris Rijsdijk, who joined midway through the thesis as my second supervisor, and put me on the right path in the first meeting with the advice to create my own fire simulation instead. His feedback provided very useful insights throughout the thesis as well. I really appreciate that both supervisors consistently made time for weekly meetings.

My thanks also go to Dominik Michels for providing code that served as an initial inspiration for this thesis, as well as for his valuable suggestions regarding the evaluation.

My academic background consists of a Bachelor's degree in Computer Science and Engineering, and a Bachelor's degree in Industrial Design Engineering. During my Master's in Computer Science, I became increasingly interested in computer graphics, because it is a combination of complex computational challenges with strong visual and practical applications. This project spanned a lot of my favourite subjects, from scene modelling, simulations, rendering, and vision-language models. This helped a lot with my enthusiasm and motivation during my thesis.

In the summer of 2024, I visited the west coast of Canada, where, as in many recent years, large-scale forest fires were occurring. This experience sparked my initial interest in the topic. From discussions with researchers at ESA to others involved in wildfire-related projects, the direction for my thesis gradually took shape. I am proud of the final result presented in this work.

Nienke Driessen
Delft, April 2026

Summary

Wildfires are increasing in both frequency and severity due to climate change, which increases the need for accurate and scalable fire modelling techniques. While existing simulation methods range from statistical models to physics-based approaches, most high-fidelity models rely on idealized, synthetic environments with complete scene knowledge. This limits their applicability to real-world forests, as well as evaluation opportunities.

This thesis addresses this gap by investigating wildfire simulation directly on real-world scene reconstructions using 3D Gaussian Splatting (3DGS). On the Feature Splatting framework, a complete pipeline is developed that reconstructs forest scenes from aerial imagery and augments them with semantic and material information.

A preprocessing framework is introduced to refine these reconstructions. It includes noise filtering, semantic classification, procedural stem insertion, and material column completion. On this representation, a physics-based, particle-driven fire simulation model is implemented, which models heat transfer, ignition, combustion, and fire spread directly on Gaussian primitives. The system is integrated into the Nerfstudio framework.

The evaluation demonstrates that the proposed simulation reproduces several characteristic wildfire behaviours reported in the literature. Fire spread increases with vegetation density and is strongly influenced by wind direction and speed. Slope experiments confirm that fire propagates faster uphill than downhill, with an accelerating increase in spread rate for steeper upward slopes. Wind-driven behaviour is qualitatively realistic, including faster burnout under stronger wind conditions. The simulation is stable across repeated runs and performs consistently on synthetic scenes.

In addition to standard evaluations, this thesis also introduces novel testing scenarios, such as forest obstruction experiments and burnt mass estimation. Results indicate that wind influence is slightly overestimated, particularly in firebreak scenarios where residual flammable material enables fire to cross barriers at lower wind speeds than expected. Burnt mass estimation reveals that the current model underestimates total biomass loss. Further calibration of material properties and combustion parameters is required.

Finally, the simulation is applied to real-world drone data. The system is capable of running wildfire simulations on reconstructed forest scenes, but challenges remain, such as noise in the data, limitations in semantic classification, and scalability issues, as these scenes are significantly larger

Overall, this work presents a novel and robust pipeline for wildfire simulation on Gaussian Splatting-based scene representations.

Contents

1 Introduction

Due to climate change, there is an increase in both the number and severity of forest fires globally [14]. This problem is expected to intensify in the coming decades [46]. Accurately modelling the behaviour of forest fires is therefore crucial for forest fire prevention and management, as well as risk mitigation.

Over the years, various methods have been developed to simulate or predict forest fires. Some of the main techniques include statistical modelling [104] [41] [17], and cellular automata [117] [47] [32] [123] [25]. Even though these models are insightful, most lack the spatial and physical detail to simulate fire behaviour in realistic, complex environments like forests. In recent years, there has been a shift towards physics-based fire modelling [80] [74] [112] [35] [53] where interactions between vegetation structure, combustion processes, heat transfer, and environmental conditions are explicitly simulated. *Fire in Paradise* [35] and its successor *Scintilla* [53] are two noteworthy recent papers that model elements such as tree geometry, ground moisture, plant combustion dynamics, heat transfer, char insulation, mass loss, and fine fuel distribution. All these components improve realism and robustness in the fire behaviour prediction.

In spite of these developments, an important drawback still exists. Most physics-based simulations run on ideal, synthetic scene data with complete knowledge of terrain and vegetation structures. Models work in a digital environment with complete knowledge of every tree, terrain feature, and other parameters. Real captured data is often taken in less ideal conditions, where most of the forest is hidden from view, weather conditions vary, and scene-capturing equipment has a restricted resolution. Therefore, thorough modelling is not feasible for real-world applications. An example where this kind of modelling is not realistic is the boreal forests in Canada, which are vast and dense, and ever-changing. In these forests, NASA found the largest increase in extreme fire behaviour [14]. Most existing high-fidelity fire simulations cannot be directly applied to real-world environments, since there is no translation between the data structures and synthetic data they are based on, and a real forest. A second challenge is evaluation. Without the ability to model real-world scenarios, evaluation by means of comparison with real wildfire observations remains difficult.

Recent progress in novel view synthesis offers a potential solution. 3D Gaussian Splatting (3DGS) is a photorealistic scene representation technique that represents a scene with anisotropic 3D Gaussians. Even though the primary use is novel view synthesis, its explicit representation proves to be useful for scene modelling. Its speed, efficiency, and quality exhibit great potential for canopy modelling [82] [103] [36].

This thesis investigates the simulation of wildfires in Gaussian splatting-based scene representations to create a fire simulation that is able to run on real forest data without switching frameworks. For this purpose, we extend the semantics-informed Gaussian Splatting framework called Feature Splatting [86] with material representations, scene refinements, and create a physics-based particle-driven combustion model to run a fire simulation on the Splatted scene. This will be done within the Nerfstudio [102] Interface, to make it as accessible as possible. The goal is to create a pipeline to predict fire spread in real forests, based on realistic scene capture footage. This thesis will mainly focus on modelling North

American boreal forests since these are subject to an increase in wildfires [14], and they are relatively homogeneous in terms of tree species, mostly coniferous [11].

This study will investigate the following research questions

1. *How can 3D Gaussian Splatting be used to reconstruct individual trees and forest patches with material properties for fire behaviour modelling?*
2. *How can a physics-based fire simulation be implemented on reconstructed forest scenes based on Gaussian Splatting?*
3. *How well does the fire behaviour simulated on splatted reconstructions match the outcomes of known digital and real-world controlled burns?*

The main contributions of this thesis include:

- A complete pipeline for simulating forest fire dynamics from aerial drone-style imagery using 3D Gaussian Splatting reconstructions.
- A preprocessing framework to prepare reconstructed forest scenes for combustion simulation, including semantic classification, noise filtering, procedural stem insertion, and material column filling.
- A physics-based particle-driven combustion model operating directly on Gaussian primitives, modelling heat transfer, ignition, fuel combustion, and fire propagation.
- Integration of the wildfire simulation into the Nerfstudio [102] framework, suitable for interactive visualization.
- An implementation of a voxel hash data structure for acceleration.
- Novel qualitative and quantitative evaluation methods of wildfire spread behaviour in reconstructed forest environments.
- Application of the simulation on real drone data from the Open Forest Observatory [76].

2

Related works

The objective of this research is to bridge the gap between physically-based forest fire simulations and practical applicability to real forest locations. We will provide a brief overview of relevant research that has led up to this point.

2.1. General scene modelling

Scene representation focuses on mapping an existing 3D scene to a digital representation. It has applications in fields such as remote sensing, medical imaging, robotics, construction, and gaming. This makes it an important and widely applicable field of research. As we need to map an existing forest patch into a simulation, we provide a brief overview of related work.

Initially, traditional Computer Vision (CV) based methods were the preferred approach [100]. These methods mostly consisted of the steps of feature extraction, feature encoding, and classification. These methods use basic image components like colours, shapes, and lines to understand the underlying structure. Some early works include the work of Laveau and Faugeras [55], published in 1994, which presented a method that makes use of existing relations between images, rather than a three-dimensional model of the scene.

Structure-from-Motion (SfM) is a method that reconstructs a 3D structure by identifying common features across multiple 2D images. Shimon Ullman introduced the first formal theory [109] in 1979. The first applications were self-calibrating metric reconstruction systems [69]. According to Beardsley et al. [10] methods varied from sequential [5] [8] [9] [39] [90] [122] to batched [40] [70] [105]. These methods were later applied to unordered internet photo collections and other scenes. After this, later works increased the method to hundreds of thousands and millions of photos [94]. Schonberger et al. [94] introduced a general-purpose method for SfM, as well as an implementation named COLMAP. In 2024, Wang et al. introduced an alternative called DUST3R [114], which leverages deep neural networks to directly predict dense correspondences and relative poses between image pairs.

In parallel to the research on recovering explicit geometry, methods were invented that store the appearance of a scene directly [34] [13] instead of the geometry, by sampling and reconstructing a 4D function they call a Lumigraph. It describes the flow of light at all positions in all directions.

From 2019 onwards, novel view synthesis using radiance fields was introduced [64] [79]. The most notable approach has been the introduction of Neural Radiance Fields (NeRF) in 2020 by Mildenhall et al. [67]. In this method, an implicit radiance field parametrized by a neural network represents light distribution in a scene without explicitly defining the geometry of the scene. From this implicit representation, new views can be generated by querying the neural network. Even though the novel views can be very realistic, NeRF requires a lot of computational power, so it can be slow to train and render, as the neural network needs to be queried each time. Additionally, because of its implicit representation, it is difficult to modify scenes, as well as to apply simulations to them. These limitations have motivated the development of alternative representations that aim to preserve the visual quality

while improving rendering efficiency and scene editability.

3D Gaussian Splatting (3DGS) [48] was presented by Kerbl et al. As opposed to the implicit neural representation of NeRF, a scene consists of 3D Gaussian distributions (splats). Not only does this method still obtain photorealistic results, but the explicit, pointcloud-like representation makes it lightweight, fast, and editable.

A notable extension to 3DGS is called Feature Splatting (FS) [86]. The FS pipeline enriches the 3DGS model with semantic information from vision-language models. This is done by distilling object-centric vision language features (CLIP [87], DINO [77], SAM [50]) into the 3D Gaussians, which allows for scene decomposition based on text queries. Additionally, the decomposed scene is synthesized using physics-based dynamics and material properties. Then it is simulated and rendered using a particle-based simulator. The semantic decomposition is very useful in our case, since different parts of a forest (e.g. stone, wood, leaves) behave differently in combustion. Since the pipeline presented in this thesis is built on such semantics and informed 3DGS represntations, we provide more detailed explanations of 3DGS and FS in Section 3.

Recently, several applications of 3DGS to forest reconstruction have been presented. ForestSplat [97] presents a large-scale forest monitoring system using 3DGS with consumer-grade drone footage. The method generates canopy height maps with an accuracy comparable to LiDAR but at much lower cost. Though this method is relevant to our research, its focus lies on monitoring tree growth over time, over large land masses of hundreds of acres. Since our research has no need for monitoring over time, the purpose differs, and it will not be used. DRAGON [38] combined drone and ground-level images for building reconstruction using 3DGS. It addressed the viewpoint gap by extrapolating intermediate views. Although this method is focused on urban scenes, it is relevant for combining multi-elevation forest data.

The aforementioned methods usually provide appearance-based and semantics- or physics-informed representations, but none consider the combustion and fire spread in addition to this. The method presented in this thesis considers this as well.

2.2. Tree modelling

The 3D representation of plants has been categorised into three categories by Godin et al. [33], namely global, modular, and multi-scale representations. In global representations, plants are viewed as a whole, either in shapes such as tree crowns represented as ellipses and trunks as cylinders, or as functional blocks, like carbon pools or water resistance. In modular representations, plants are decomposed into repeating units like a graph. Thirdly, multi-scale representations describe the plant at hierarchical levels of detail. In larger scenes, it might be useful to apply a multi-scale representation in order to increase scalability, both for rendering as well as fire simulation purposes.

Among some notable examples of 3D plant representations, Bailey et al. [6] presented Helios, a plant and environmental modelling framework with adaptable plug-ins of state-of-the-art biophysical models. Metsanis et al. [68] developed a framework for modelling agricultural digital twin plants, where the growth of a plant is observed and mapped in a digital twin. While this thesis does not focus on the growth of plants, down the line, a forest fire digital twin could be created. Another work by Kochi et al. [52] reconstructed plants as a point cloud using images with an improved SfM-MVS method. A different approach was followed by Liang et al. [58] with the use of hyperspectral imaging, which helps in segmenting plants from their background to build 3D plant models. Although this could help forest scene recreation, for now, this work focuses purely on image-based input in order to make the system as accessible as possible.

The tree reconstruction method developed by Li et al. [57] applies 3DGS to reconstruct leafless trees and extract structural features like the Diameter at Breast Height (DBH). It achieves high accuracy on detailed branching. While our focus is on leafy trees, the technique shows that 3DGS is effective for fine-scale vegetation modelling.

Since conifer tree species have distinctive branch structures, several approaches focused on conifer-specific representations. Meyer and Neyret [65] presented a cone-based pine tree model, where each needle is a cylinder with a certain angle to a branch. Zhang et al. [121] presented a leaf polygon

decimation method. Pirk et al. [80] represented leaves as 2D surfaces with a specified mass and area. Upon combustion, the leaves' surfaces shrink proportionally to their mass loss. The work of Mendoza et al. [63] is potentially the most applicable to our research, as they dispersed Lagrangian particles within a conical volume to present a conifer specifically. This point cloud-like representation can be matched to a 3DGS object, as shown in Figure 2.1.

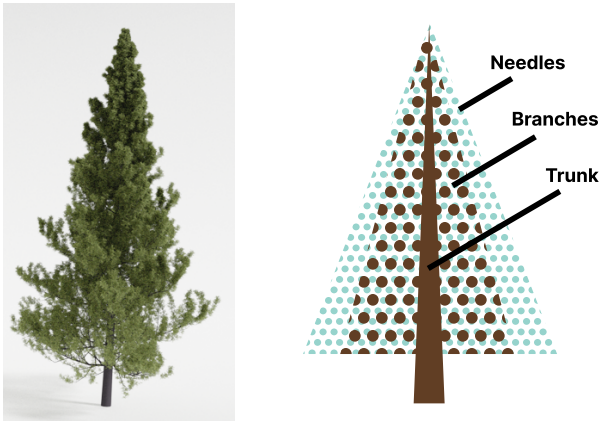


Figure 2.1: Simplified model of a conifer tree as presented by Mendoza et al. [63]

.

2.3. Forest modelling

Moving from modelling single trees to modelling forests, new challenges arise as forests are very complex systems. They are large, dense, and ever-changing. It is essential to develop accurate digital models of forests [72] to be able to simulate and predict forest fires. In addition, realistic models of forests have applications beyond science, such as in training, education, the planning of urban or rural environments, and (serious) games.

Forest geometry is often obtained using 3D measurement techniques, such as airborne or terrestrial LiDAR, digital elevation models, or photogrammetry from aerial photography [72]. We can discern between two branches of forest modelling.

One branch is the reconstruction of the forest from real-world measurements. An example is *3D Forest* by Trochta et al. [106], a terrestrial laser scanning application for detailed structural forest modelling. Their application extracts important parameters of forest structure from the terrestrial laser scanning data, as well as enabling tree and crown segmentation. Qiu et al. [85] presented the vision of the forest digital twin paradigm for analysing forest spatial structure and monitoring tree growth. It must be noted that the use of LiDAR point cloud data is not always feasible due to cost or location [57].

The second branch involves simulating forest behaviour, based on ecological knowledge and growth models. This can be used for controlled experimentation and the study of forest dynamics under different conditions. An example of this is a paper written by Makowski et al. [61] which presents a method to design large-scale ecosystems based on plant growth and interactions, tropism types, and resource competition, which basically grows a forest from scratch. Ecoclimates [78] focused on local variations in climate by modelling the feedback loops between vegetation, soil, and atmosphere.

The FOR-instance dataset introduced by Puliti et al. [82] provided manually annotated tree segments. The dataset distinguishes semantic classes specific to conifer forests, such as stem, woody branches, live branches, terrain, and low vegetation. This level of detail is abstract yet detailed enough for the combustion in our work. A later contribution by Puliti et al. [81] was the FOR-species20K dataset, which focused on tree species classification. Since our study is limited to conifers without distinguishing between subspecies, it will not make use of the FOR-species20K dataset.

2.4. Combustion

Over the years, several approaches have been proposed to simulate fire and combustion. A common approach is physics-based fluid simulation, where fire is treated as a reactive fluid and modelled by solving the Navier–Stokes equations coupled with temperature, density, and combustion terms [120] [124][45]. Even though these methods can produce very realistic results, they are computationally expensive and typically require dense volumetric discretization.

Several works explicitly model heat transfer mechanisms. Melek and Keyser [62] proposed a linear combustion model in which fuel, oxygen, and combustion products are represented separately. The heat transfer is modelled in the form of convection and conduction. But in large-scale outdoor fire scenarios, radiation plays an important role in ignition and fire spread over distance. Therefore, a realistic representation of radiative heat transfer should be taken into account. De Ris et al. [21] proposed a method of radiation fire modelling. It preserved full-scale flame heat fluxes by increasing the ambient pressure as the scale is reduced. However, this approach was designed for controlled scaling experiments and did not directly address heterogeneous, large-scale natural environments.

Another fire modelling approach is particle-based simulation. In 1983, the first systematic method for particle systems expanding fire simulations was proposed by Reeves et al. [88]. Particle systems are a technique used to model fuzzy objects, which are the basis of many fire simulation algorithms [116]. However, early particle systems were primarily designed for visual realism rather than physically-based combustion or material-dependent fuel consumption.

Fuzzy objects, often sourced from remote sensing data, are objects that can only be defined and monitored with a certain level of certainty. Molenaar et al. [71] proposed a method to estimate object behaviour through field data sampled at different times by identifying fuzzy objects and their dynamics. Our work exhibits uncertainty in both the geometric characteristics and the features extracted from digital images. If a scene is not completely accurately defined, the fire simulation should behave stochastically and account for local variability. Therefore, for this work, we chose a particle system to simulate fire.

Particle systems are based on a number of particles, each consisting of a set of attributes. Attributes can be location, lifespan, size, transparency, shape, speed, direction, acceleration, etc. The ever-going motion of the particles over time will continuously change their properties.

One example of a particle-based fire simulation was presented by [42], who proposed a combination of coarse particle grid simulation with GPU-based fine, view-oriented refinement simulations. While this method produces visually convincing flames, it is not useful for fuel combustion or fire propagation as these factors are simplified. For example, mass decayed exponentially instead of using material-dependent combustion dynamics.

Nielsen et al. [73] argued that to model the underlying physics, simulations will look much more realistic, and propose a physics-based combustion model, using mathematical models for the thermodynamic properties of real-world fuels. However, their work only modelled a single flame.

Liu et al.'s FlameForge [59] is a volumetric combustion simulator that captures the multi-phase combustion of charring materials, focusing on general wooden structures. It is not directly applicable to our work, since they use a voxel representation of the geometry. Trees have complex branching structures and a high surface-area-to-volume ratio, where voxel representations work better for dense volumetric objects, which makes a voxel representation not ideal for our case. However, the material combustion formulae are applicable.

2.5. Forest fire simulation

Forests add another level of complexity to fire simulation. Early models predicting wildfire worked with Cellular Automata (CA), either working with a square [16] or a hexagonal grid [108]. Several methods were presented, including a method using cellular automata to simulate wildfire propagation [29] [107] [3]. For simulating complex fire behaviour, CA is too simple. Because of the simplified discrete representation, grid distortions can occur.

The next generation of wildfire modelling can be divided into two main branches, empirical and physical

Figure 2.2: Left: the hierarchical module structure presented by [35]. Right: The bounding boxed approach by [119]

models [7]. Empirical models basically try to describe the pattern, while physical models explicitly try to capture the fundamental process. Empirical models are simplified mathematical models, but are able to solve fire propagation fast [89] [111], while physical models often numerically solve equations for the fluid dynamics and chemistry of fires [12] [2].

The new generation of wildfire models used both physical and empirical fire models, and even the interaction of wildfire with the surrounding atmosphere. With the introduction of artificial intelligence, new solutions have been developed. Sharma and Puskar [98] developed machine learning and neural network approaches using factors like meteorology, terrain, vegetation, and infrastructure. Another example is where Neural networks were used in [93]. Here, satellite imagery over large areas is used to predict the occurrence of wildfires. Another work made use of meteorological data in combination with infrared scanners to minimize false alarms in forest fire detection [4]. While these approaches can be effective for forecasting fire occurrence or spread probability, they do not explicitly model the physical combustion process itself. Since the objective of this research is to simulate fire behaviour in a specific reconstructed forest scene, a physics-based approach is preferred.

Apart from methods that use a large set of data, some methods leaned more towards recreating the fire combustion process digitally. Pirk et al. [80] presented a method for calculating the combustion of botanical tree models by representing trees as connected particles. Built on top of this, the aforementioned paper called *Fire in Paradise* [35] scaled the combustion to fire propagation through an entire ecosystem. This work had several notable contributions, such as a hierarchical module structure to represent trees and physics-based combustion dynamics. Section 3 covers this paper in more detail.

In the follow-up work, *Scintilla* [53], the FiP forest fire model was expanded with several factors for increased realism and accuracy. Factors were added, such as fire spread on the ground, in grass, branch litter, and undergrowth vegetation, as well as the incorporation of fuel moisture, firebrands, and turbulence of fire. Mechanisms presented in their work will not be used in our work since we want to maintain simplicity as much as possible, for the results to reflect mainly the performance of the Gaussian combustion.

Another approach by [119] presented a finite state machine approach. The simulation represented trees in two parts, where the leaves were represented by spherical bounding boxes, and the trunk and thick branches were described by columnar bounding boxes.

These models assume the fire simulation starts with a ground truth of the forest scene, represented in the necessary tree representation. Figure 2.2 provides a comparison of tree representations. Realistically, there is no ground truth of the scene readily available for real-world forest fires.

3

Background

3.1. Neural Radiance Fields

Before explaining the method used in this paper, it is important to mention its predecessor, called Neural Radiance Field (NeRF). NeRF represents a scene as a continuous function learned through differentiable volume rendering. This implicit neural scene representation is parametrized by a Multi-Layer Perceptron (MLP), which models the geometry and appearance of a 3D scene. Once it is trained, the representation can be used to generate novel 2D views by integrating predicted colour and density values along camera rays.

Even though NeRF greatly improved the quality in novel view synthesis, it still suffers from low training and rendering speed [115].

3.2. Gaussian Splatting

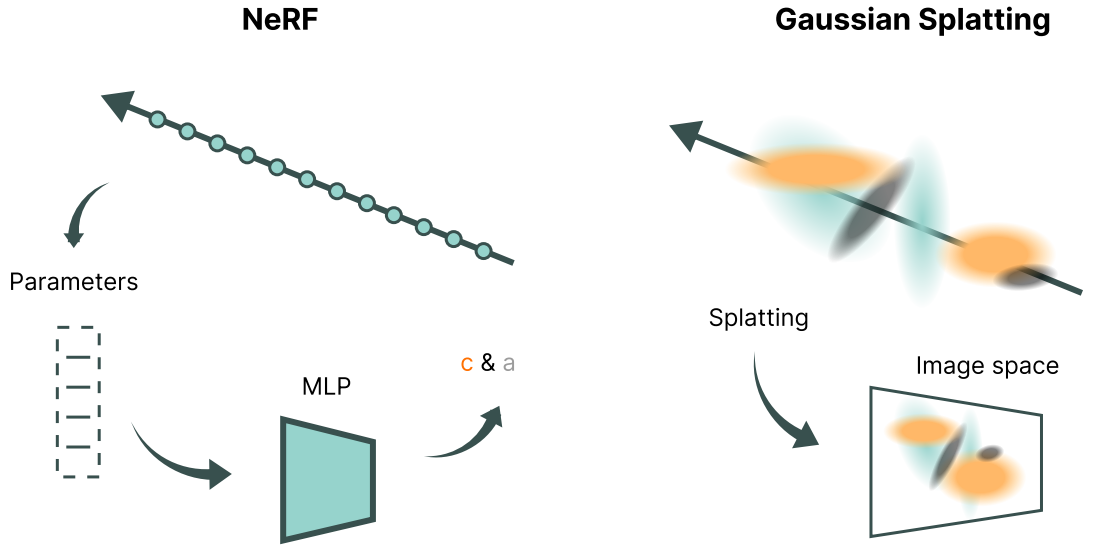


Figure 3.1: The conceptual difference between NeRF and 3DGS. NeRF uses a neural network to create a volumetric scene representation; rendering is done by querying an MLP for each point. 3DGS represents a scene as a collection of 3D Gaussian distributions, and rendering is done by directly rasterizing the explicit primitives.

3DGS is a 3D scene representation method first presented by Kerbl et al. in 2023 [48]. It can be seen as a successor or alternative to NeRF. Unlike NeRF, 3DGS makes use of an explicit scene representation, using Gaussian ellipsoids to model the scene for efficient rendering. Therefore, it has several advantages over NeRF, such as faster training and inference times, being more interpretable, and facilitating editing. This is why, in this study, Gaussian Splatting is chosen over NeRF. Figure 3.1 shows

a simplified schematic of the conceptual difference between the two methods.

3DGS representation

The method starts from a set of posed images, together with camera intrinsics and extrinsics, just like NeRF.

In Gaussian splatting, a 3D scene is represented as a set of 3D Gaussian primitives represented by the following parameters:

- Mean $\mu_k \in \mathbb{R}^3$ (The location of the point in 3D space)
- Covariance $\Sigma_k \in \mathbb{R}^{3\times3}$ (the anisotropic scale and orientation)
- View-dependent colour function $f_k \in \mathbb{R}^d$ (often modelled with low-order spherical harmonics)
- Alpha $\alpha_k \in [0, 1]$ (opacity)

Each Gaussian implicitly defines a spatial density that determines how strongly it contributes to nearby points in space and, after projection, to image pixels. This density can be written as:

$$\mathcal{G}_k(\boldsymbol{x}) \coloneqq \exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_k)^\top \boldsymbol{\Sigma}_k^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_k)\right),$$

where k denotes the index for the k-th Gaussian.

Splatting and rendering

Given a set of 3D Gaussians and a camera, this step produces a rendered image that can be compared to ground truth.

Rendering a view is achieved in two main stages. The first is a preprocessing step that projects the Gaussians into screen space. And the second is a rasterization process with front-to-back alpha compositing.

Firstly, each 3D Gaussian is projected into image space using the camera projection $\pi(\cdot)$. The projected mean is:

$$\mu_k^\pi \coloneqq \pi(\mu_k),$$

and the covariance is approximated using a first-order Taylor expansion:

$$\Sigma_k^\pi \coloneqq J_k^\pi \, \Sigma_k \, (J_k^\pi)^\top,$$

where J_k^π denotes the Jacobian of the projection evaluated at μ_k. This results in a 2D Gaussian $\mathcal{G}_k^\pi$ in screen space.

Secondly, by alpha-compositing each primitive's decoded feature, an image is created. For each pixel, only Gaussians whose projection overlaps the tile the pixel belongs to are considered.

These Gaussians are sorted based on ascending depth of their means. The final pixel colour C_p is obtained via front-to-back alpha compositing:

$$C_p = \sum_{i=0}^{N-1}\left(C_i \, \alpha_i \, \mathcal{G}_i^\pi(\mathbf{u}) \prod_{j=0}^{i-1}\left(1-\alpha_j \, \mathcal{G}_j^\pi(\mathbf{u})\right)\right),$$

where C_i comes from the view dependent colour f_k and α_i denotes the opacity of the ith Gaussian at the pixel from the projected Gaussian $\mathcal{G}_k^\pi$ [20].

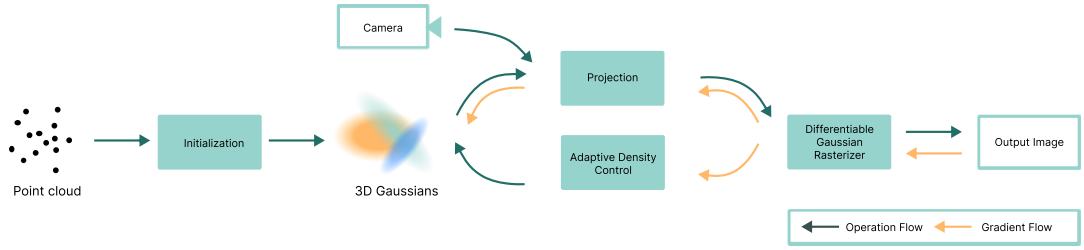


Figure 3.2: Optimization procedure. Point cloud initialized with SfM.

Optimization

The optimization step is visualized in Figure 3.2. The input is a sparse point cloud generated from given images with associated camera paths based on Structure from Motion (SfM). The Gaussians are initialized at the locations of the points in the point cloud. Adaptive density control is applied, which is a process responsible for the pruning and densification of the Gaussians. Next, a camera position and its respective image are sampled, and an image is rendered by the method mentioned above. The resulting image is compared to the original image, and the loss is calculated and backpropagated to optimize the number of Gaussians and their characteristics.

3.3. Feature Splatting

Feature Splatting, presented by Qiu et al. [86], enriches 3DGS with semantics from vision language foundation models. This is achieved by distilling object-centric vision-language features into 3D Gaussians. For this purpose, Feature Splatting appends an additional vector $f_i \in \mathbb{R}^d$ to each individual Gaussian. The Gaussians are rendered in a view-independent manner since the semantics of an object remain the same regardless of viewing directions.

CLIP [87] features are enhanced with DINOv2 [77] and the Segment Anything Model (SAM) [50] to improve the quality of the Gaussian features. CLIP is a neural network developed by OpenAI that connects text and images through a shared embedding space, which enables it to recognise visual concepts semantically. DINOv2 is a self-supervised computer vision-only model released by Meta AI. It excels at structural understanding of the scene, like depth estimation, textures, and geometry. SAM is a segmentation AI model, also developed by Meta AI, that allows for promptable segmentation. Since CLIP features are very coarse, if they are used alone, this results in low-quality 3D features.

SAM is used to generate a set of part-level masks, M. Given M and the coarse CLIP feature map F_C, Masked Average Pooling (MAP) is used to aggregate a single feature vector

$$w = MAP(M, F_C) = \frac{\sum_{i \in F_C} M(i) * \frac{F_C(i)}{\|F_C(i)\|}}{\sum_{i \in F_C} M(i)}, \tag{3.1}$$

where i represents a pixel in the feature map. w is assigned to all pixels within the part segmentation. If a pixel belongs to multiple parts, the average value of the features of all relevant parts is assigned. The result is a SAM-enhanced CLIP feature map

A shallow MLP is introduced to reduce the possibility of over-fitting. The rendered features $\hat{\mathbf{F}}$ is the input, and the output is two branches: the first is the DINO feature $\hat{\mathbf{F}_D}$ for its coherent part-level semantics, and the second is the CLIP features $\hat{\mathbf{F}_C}$

$$\hat{\mathbf{F}_C}, \hat{\mathbf{F}_D} = MLP(\hat{\mathbf{F}}). \tag{3.2}$$

$\hat{\mathbf{F}_C}$ is supervised using the SAM-enhanced CLIP feature map using cosine loss. $\hat{\mathbf{F}_D}$ is supervised using the DINOv2 feature map using cosine loss. The CLIP term is scaled in the joint loss $\mathcal{L}_{CLIP}$ + $\mathcal{L}_{DINO}$ with λ = 0.1. The optimization uses DINO features as a mild smoothing term. The result is Gaussians with regularized CLIP features.

These features can then be matched to CLIP text embeddings for every vocabulary, using the CLIP text encoder. Both positive vocabularies

$$L^+ = \text{'bulldozer'}. \tag{3.3}$$

As well as negative vocabularies, for example:

$$L^- = \text{'objects' and 'things'}, \tag{3.4}$$

after which pairwise cosine similarity is calculated between the rasterized CLIP features of every Gaussian and the text embeddings. Figure 3.3 shows an example of similarity estimations within a scene.

Figure 3.3: Example similarity estimation given the text prompt *'A vase with flowers'*. Example taken from FS [86].

3.4. Forest fire modelling system

Our work takes inspiration from the forest fire simulation by Hädrich et al. [35] titled *Fire In Paradise* (FiP), which introduces a method for realistically simulating wildfires at the scale of forests. The paper presents a tool with which fire can be simulated interactively, and the effects of various factors like terrain, ecosystem composition, and counteractive measures can be tested.

The most relevant innovation for our work is their novel mathematical formulation for plant combustion, as well as their two-phase combustion simulation, where the modules are updated based on heat exposure, after which the heat flow system is updated. Some differences exist between their system and ours. In FiP, the combustion is calculated on truncated-cone-like tree segments called 'modules', whereas our system uses Gaussian splats to represent material. The FiP system models heat flow as a fluid simulation, whereas our simulation takes a particle-based approach.

Tree module structure

In FiP, each branch is represented as a truncated cone. The combustion of wood is calculated on a module-level, which is defined as follows. A module M consists of a set of (adjacent) truncated cones. A tree T is a set of connected modules. Each tree can be decomposed into different numbers of modules depending on the desired level of detail. Finally, a forest F is defined as a set of trees.

Since our work is based on 3DGS scenes, which are not hierarchical by nature, this structure will not be adopted. However, hierarchy may be useful in the future for simulating large scenes.

Fire combustion model

The combustion process in [35] is calculated at module level. For each timestep, for each module, the mass loss is calculated via

$$\frac{dM}{dt} = -k(T_M)\, c\, A, \tag{3.5}$$

where k denotes the reaction rate, dependent on the temperature T_M. c is a dimensionless char insulation parameter, and A is the pyrolysing front area, both depending on the tree geometry.

The reaction rate $k(T_M)$ is defined as

$$k(T_G) = \eta \begin{cases} 0 & \text{if } T_G < T_0, \\ S((T_G - T_0)/(T_1 - T_0)) & \text{if } T_0 \leq T_m \leq T_1, \\ 1 & \text{if } T_G > T_1, \end{cases} \tag{3.6}$$

And is applied with a constant η. $S(x) = 3x^2 - 2x^3$ indicates a sigmoid interpolation function, from zero to one for temperatures between T_0 = $150°C$, and T_1 = $450°C$.

The wind speed u is accounted for in η, where $\eta = 1$ equals no wind, and $\eta = (\eta_{\text{max}} - 1)S(u/u_{\text{ref}}) + 1$. A threshold velocity of u_{ref} is defined, at which a maximum boost is reached.

4

Methodology

The aim of this thesis is to create a practical and reliable forest fire simulation based on physical processes. An overview of the proposed system pipeline is presented in Figure 4.1. As discussed in chapter 2, the capture of forest regions may involve different modalities, such as imagery, spectral data, LiDAR data, etc. Given that forests can be very extensive, for usability, we assume that the data we are working with is photos taken from a drone or aeroplane flying overhead.

The system takes photographs or a video of a scene as input. After preprocessing steps, to estimate camera parameters for the input images and a sparse scene reconstruction, we compute a semantics-improved 3DGS scene representation that allows fire simulation in the scene, including the prediction of the movement of fire through the scene.

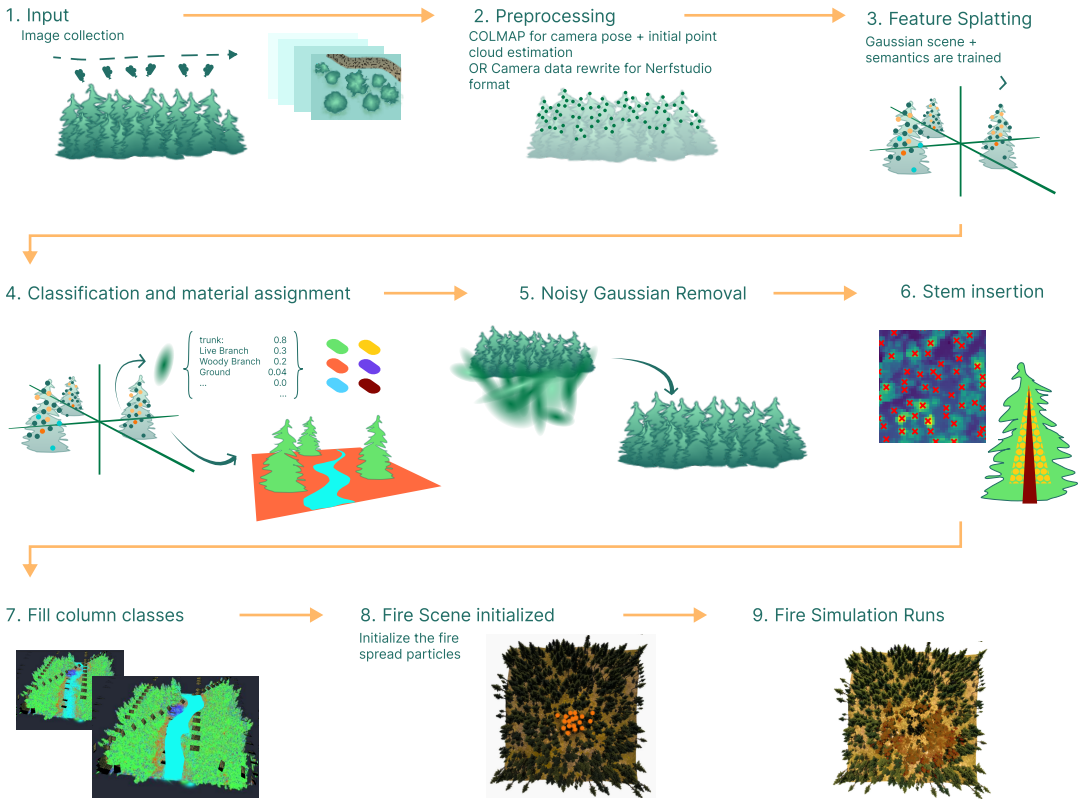


Figure 4.1: The overarching pipeline of the system. From step 2, all is done within the Nerfstudio framework.

In the following sections, we provide more details on the individual steps.

4.1. First stage: Semantic 3DGS Scene creation.

The first step in the pipeline is to convert captured video frames or images into a 3D digital scene as the starting point for the fire simulation.

4.1.1. Data requirements

Even though one of the strengths of the pipeline is that it works with an unstructured set of photos of a scene, it is useful to provide advice on certain guidelines. Following these guidelines will help to ensure that the recreated scene contains as few artefacts as possible.

Diverse camera angles

The generally sparse views of aerial surveillance can create the problem that the recreated 3D model is not realistic [37]. It is therefore recommended that the drones fly in a pattern so that the scene regions are observed from different angles, ideally in a structured formation like a grid. Additionally, the camera should ideally not be facing perpendicular to the ground but at an angle. The Open Forest Observatory (OFO) [24] created a drone data collection protocol, which can be used as a guideline for capturing forest footage. Figure 4.2 shows two examples of forest capture missions in California.

Lightning conditions

Secondly, scene digitization is facilitated if the photos are taken with approximately the same lighting, because different shadow structures can confuse the estimation of the camera positions. According to [24], the best results are obtained when performing flights within 3 hours of solar noon to minimize shadows.

Image overlap

The frames or photos must have a certain level of overlap to ensure that every part of the scene is visible. OFO advises around 80% front and 70% side overlap for low-altitude oblique missions, and 90% front and 90% side overlap for high-altitude nadir missions.

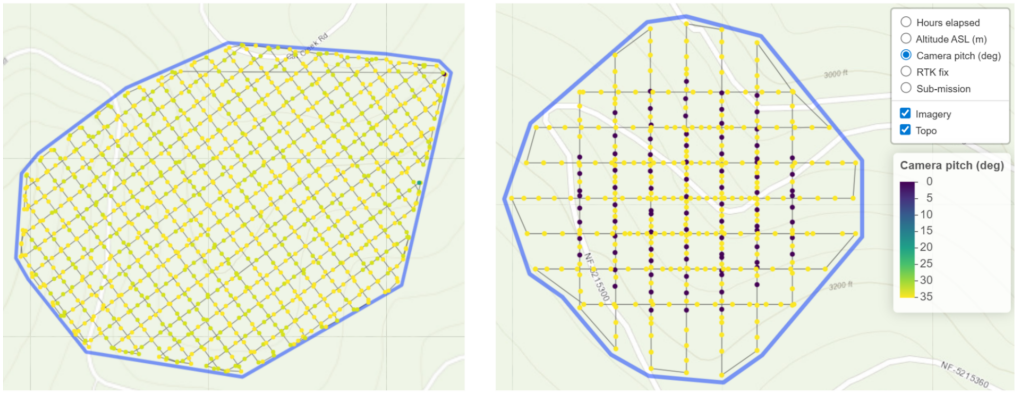


Figure 4.2: Examples of an ideal capturing pattern. One larger and one smaller capture mission from the Open Forest Observatory [76].

The better the data adheres to these points, the better the reconstructed scene is. In turn, a more accurate scene representation will lead to a more reliable fire simulation.

4.1.2. Preprocessing steps

Gaussian splatting requires not only images but also camera intrinsics and extrinsics for input, as well as an initial point cloud to enforce the structure. If the camera positions, angles, and intrinsics are known, it is very useful to provide them to the preprocessing phase, as this can prevent artefacts. This can be done by converting these properties into the input format used for FS. In case the data does not include these elements, a preprocessing step is provided using COLMAP [95] [96]. COLMAP is a standard Structure-from-Motion (SfM) and Multi-View Stereo (MVS) pipeline created by Schönberger et al. [96] in 2016. It estimates camera extrinsics and intrinsics through SfM, which are then used in the MVS pipeline to create a dense representation of the scene.

There are more recent methods for SfM and MVS, like DUST3R [113], which is an AI-driven approach that treats reconstruction as a regression problem using transformer models. It is robust to sparse, unconstrained views. However, we decided to still work with COLMAP as we assume the images do have a lot of overlap, and our image sets have many images. COLMAP has the additional practical advantage that it has been integrated with Nerfstudio.

4.1.3. Forest Representation based onGaussian Splatting

To represent the 3D forest scene, we chose the method of 3DGS, with its implementation within Nerfstudio. Nerfstudio [102] is an open-source tool designed to make NeRF or Gaussian splatting techniques easy to use. It was chosen because it contains many extensions, has a user-friendly UI, and good documentation.

Among the included 3DGS approaches, Nerfstudio contains a fast and memory-efficient implementation of Gaussian splatting called GSplat [118]. In Nerfstudio, the realization of the GSplat method is called SplatFacto.

In particular, we built our work on top of Feature Splatting, a semantics-enriched Gaussian splatting approach by Qui et al that enriches 3DGS with rich semantics from vision language foundation models. The Nerfstudio FS version does not include the material dynamics, but they are not necessary for this paper. Figure 4.3 shows how all the aforementioned methods relate and result in the FS implementation used and adapted by this paper.

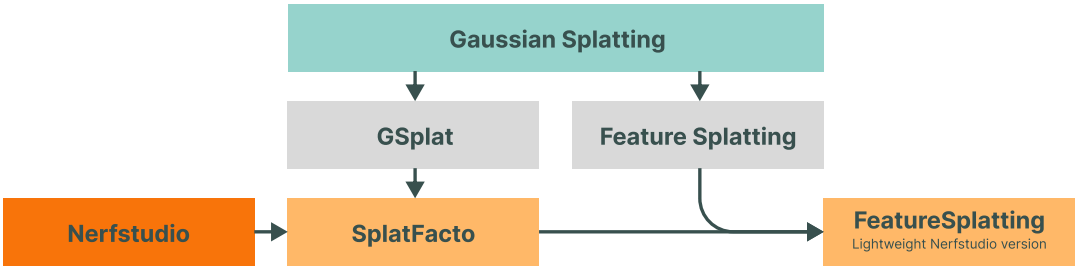


Figure 4.3: The relation between GSplat, SplatFacto, and FS.

Given input images, camera intrinsics and extrinsics, and an initial point cloud, FS optimizes for a unified Gaussian representation as explained in Section 3. Additional to optimizing for geometry and texture, this method also infuses the scene with semantics using features from large-scale 2D vision models. The result is that each Gaussian has the following properties:

- *Mean* $\mu_k \in \mathbb{R}^3$ (the point's location in 3D space)
- *Covariance* $\sum_k$ (the stretch/scale of the point)
- *View-dependent color function* $f_k \in \mathbb{R}^d$ (often modeled with low-order spherical harmonics)
- *Alpha* $\alpha_k \in [0, 1]$ (transparency)
- *Distilled semantic features* as a learned latent feature vector.

4.1.4. Semantic segmentation

The material of an object has a large influence on the combustion process [49]. Since the vision language models used in FS [86] allow us to estimate the similarity to the object or material a Gaussian represents, it is very beneficial to use this in the calculations for fire prediction, not only to steer the combustion behaviour, but also to identify which objects do not burn.

Since most of the scenes are coniferous forests, trees will be divided into different parts, or classes. It must be taken into account that the 3DGS-based scene is a digital surrogate of the real forest scene, so it is not infinitely precise.

The class division chosen for our approach follows the class division defined by the FOR-instance dataset [82], which was created to provide ground truth annotations for tree segmentation algorithms. In their work, the authors divide a tree into three classes:

1. *Stem*: points belonging to the central tree stem axis.
2. *Woody branches*: woody material on the tree without any leaves or needles attached.
3. *Live branches*: branches with leaves or needles attached to it.

Given the relatively large scale of wildfires, this division is sufficiently abstract to be recognized from afar, but distinctive enough to model the main differences in combustion.

As the scenes often include not only trees, but also other types of materials, like rocks, rivers, or man-made objects which are either not flammable or have a certain type of combustive behaviour, a system is defined to add and/or remove classes depending on the specific scene. To make this system user-friendly and adaptive to a situation, we decided to make the classification adaptable by the user. A JSON file can be edited to change the material list. A material is defined by the properties shown in Figure 4.4.

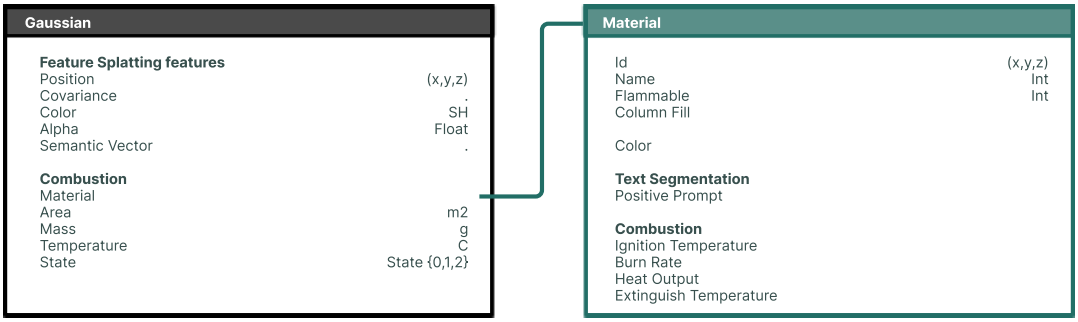


Figure 4.4: Properties stored with Gaussians. Beyond the standard properties like position, covariance, and colour, we store combustion and material properties.

Class assignment

To assign material attributes across the scene, we segment the scene into regions with respective material classes by means of a positive text prompt. This consists of a group of words that are distinctive for that specific class. Each material text is converted into a semantic vector using the CLIP text encoder [87]. The input prompt is tokenized into word pieces. Tokens are passed through a Transformer-based language encoder, and the final hidden representation is projected into the CLIP embedding space [86]. Some examples of class specific text prompts are:

- *Stem*: ”tree stem, bark, trunk, log, tree trunk, tree bark, wood”
- *Live branch*: ”green leaves, leafy branch, foliage, green needles, green branch, leaves,”
- *Rock*: ”rock, stone, boulder, granite, concrete”

In our 3DGS-based scene representation, each Gaussian in the scene contains a learned feature vector. Each vector is passed through a small MLP, which produces a CLIP-aligned feature vector. The semantic similarity between each Gaussian and the text prompt is computed using a dot product between normalized vectors. Similarity scores are computed for each class prompt. The predicted class for a Gaussian is assigned to the highest-scoring prompt. A confidence threshold was added, where Gaussians with a score too low are assigned to the default non-flammable class. Figure 4.5 shows an example of the scene material classification.

Stem insertion

In 3DGS [48], the main goal is to recreate a scene to look like the images it is trained on. This means it shapes the Gaussians to match the visible area. Therefore, pine needles are often more present than the trunk, which is often occluded, as also explained by Puliti et al. [82]. Especially in dense forests, the underlying, unseen forest structure is therefore less, if not unrepresented, in forest scenes. Since tree stems and dry branches make an important contribution to the combustion, we provide the option to reinsert stems within the scene.

Trunks are assigned by an algorithm that identifies local maxima in the canopy. By segmenting the live-branch class assigned Gaussians, a canopy height map (CHM) is generated by discretising the

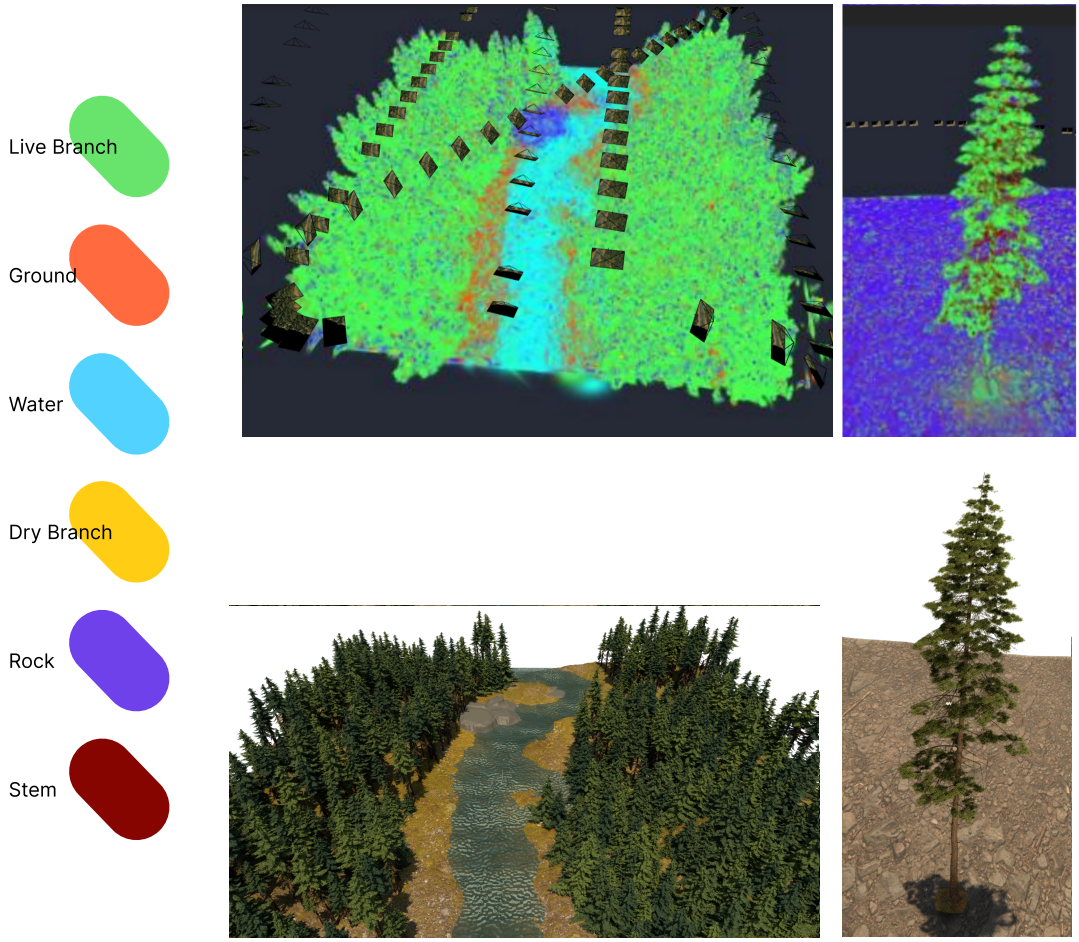


Figure 4.5: Classification of the scene into materials defined by the user. On the left, the colours each material represents. On the right, two example forest scenes.

horizontal plane into a voxel grid and storing the maximum height value of the live-branch typed Gaussians within each cell. Next, the CHM is smoothed using a Gaussian filter with NaN-aware weighting to ensure missing data does not bias neighbouring values.

Local maxima with a minimum prominence are selected as candidate treetops. The candidate treetops are further validated by checking for vertical support. Only if a sufficient number of original 3D points exist within a small horizontal radius beneath the peak location, the treetop is kept. This ensures that noise maxima are not taken into account.

Because neighbouring peaks can belong to the same tree crown, spatial clustering is applied horizontally. All peaks closer than a minimum distance are merged, and the highest point within each cluster is assigned as the final treetop estimate. Figure 5.13 provides an example of the treetop detection on a 20% inclined slope. This shows that the tree recognition works reasonably well on non-flat ground.

Finally, simplified stem geometry is reconstructed for each detected tree. For every treetop, stem points are inserted vertically from approximately 80% of the detected canopy height down to ground level at the same horizontal position. Woody-branch Gaussians are inserted in a cone-like manner, following the simplified model of Mendoza et al. [63], as shown in Figure 4.7.

Ground removal

Another challenge raised by 3DGS scenes is the presence of noisy Gaussians at the edges of the scene. For example, the scene may not be adequately represented, leading to large Gaussians. Since these can incorrectly influence the fire behaviour, the noisy Gaussians need to be removed. Firstly, a ground height map is estimated, using all ground material Gaussians. To reduce noise and remove extreme

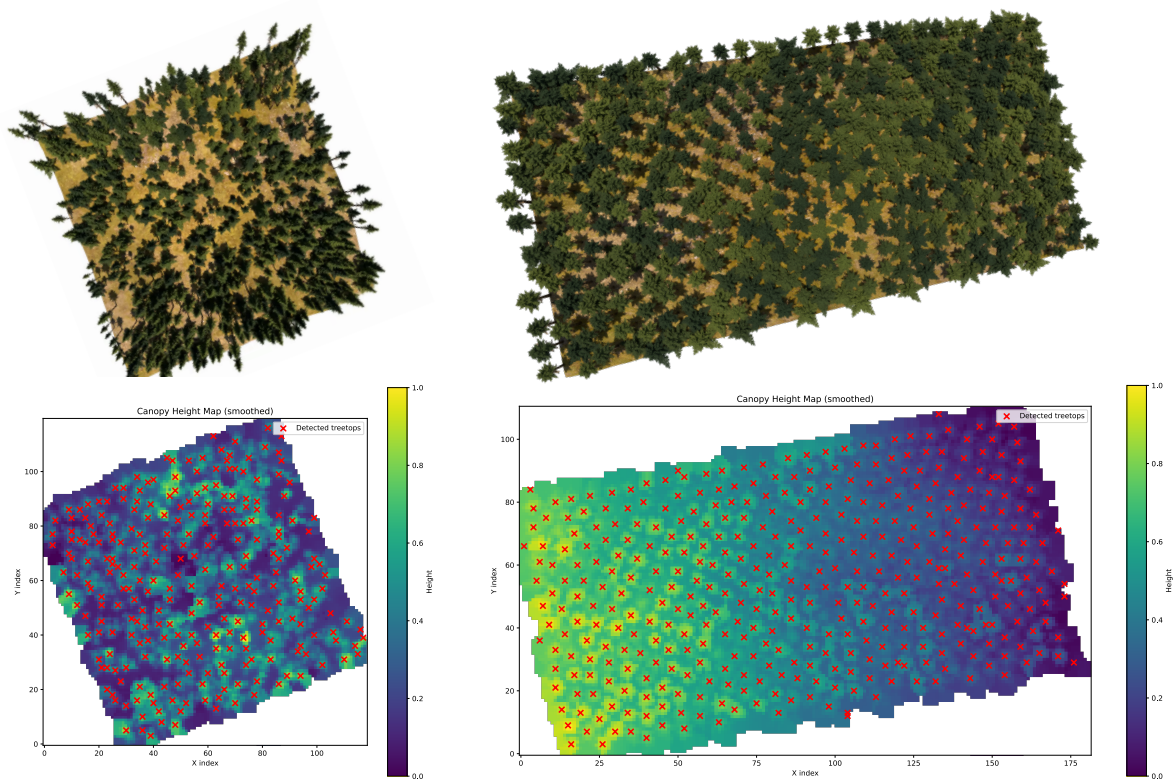


Figure 4.6: Peak detection in a canopy height map. Each cross depicts a detected treetop. The colour indicates the relative height of the scene (purple = lowest canopy point, light green = highest canopy point

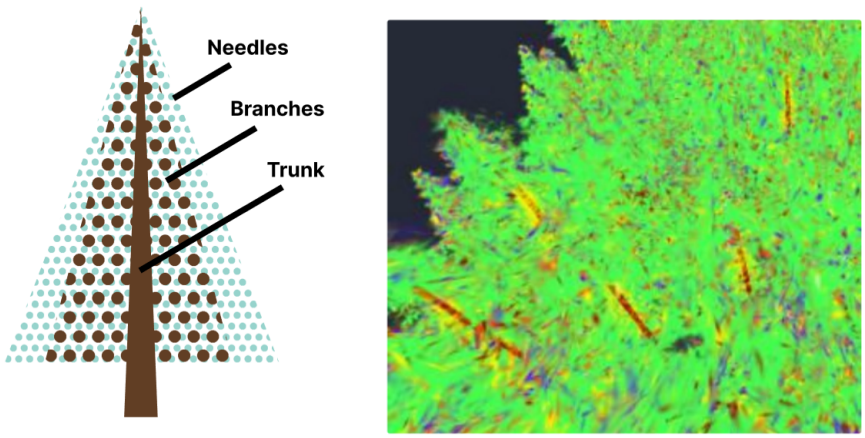


Figure 4.7: Stem and dry branch insertion into the scene. The cone-like dispersion follows the simplified model proposed by [63]

points, the vertical coordinates of the points are clipped based on percentiles. The filtered point cloud is then discretized into a 2D grid with a cell size of approximately 0.5 m. For each grid cell, all points within the cell are averaged to obtain an initial estimate of the ground height. Then, using a k-d tree nearest neighbour search, empty cells are filled so the height map does not contain gaps. Finally, a Gaussian filter is applied for smoothing. The ground map can also be used to set all underground Gaussians to non-flammable, to make sure the fire does not unrealistically propagate through the ground.

Figure 4.8 shows an example of a ground map, and a before-and-after image of ground noise removal.

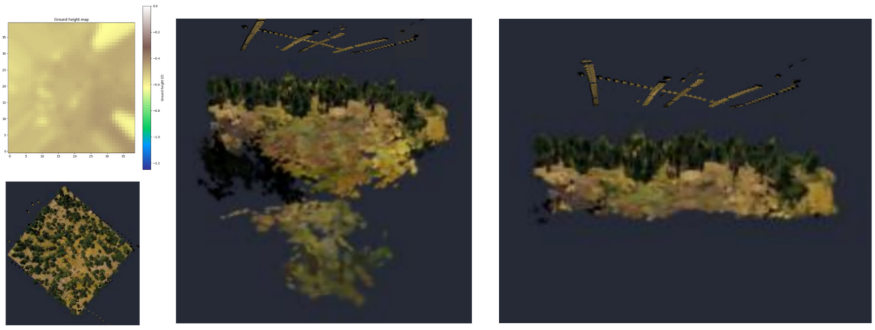


Figure 4.8: Example of ground removal. Left: the scene and the corresponding height map. Middle: Gaussian scene before ground removal. Right: Gaussian scene after ground removal.

Other sources of noise, such as the sky or scene boundaries, can also affect the simulation. These are not explicitly addressed in this work. For the time being, manual editing of the Gaussian scene using an external editor, such as SuperSplat [23], is recommended.

Column filling materials

As the classification is not entirely uniform, many areas, even if they are primarily classified as one type, contain outliers. This is usually not a problem, except for a few types that are strictly non-flammable. If not all Gaussians in that section contain the non-flammable material, the fire may still pass through the material, as if jumping across to the other side via these 'stepping stones'. This can be seen in Figure 4.9, because not all Gaussians within the water are classified as 'water', fire can still spread through. Therefore, we have decided that some classes have the property of filling an entire column with their type, these being the classes 'water' and 'road'. The scene is columnised, and for each column that contains over a certain percentage of the column material, all Gaussians whose centre lies within the column are assigned the class. This is visualized in Figure 4.10. This prevents fire from crossing under the rivers or roads.

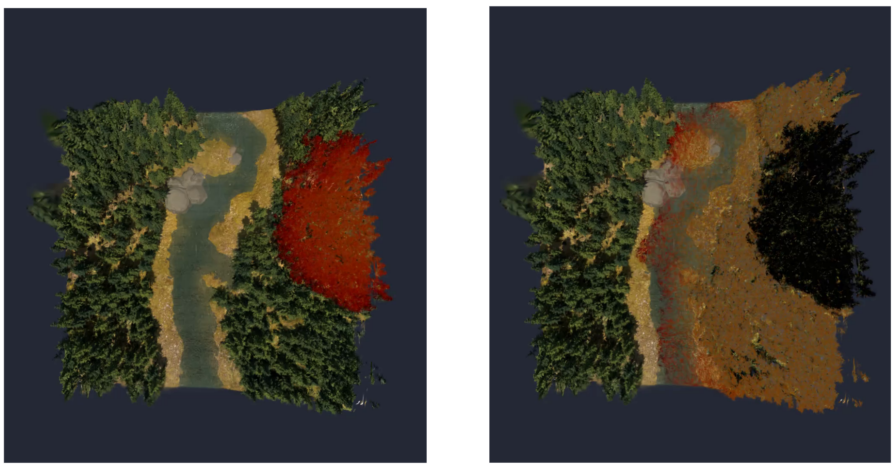


Figure 4.9: The occurrence that not all Gaussians in the river are classified as water, making it possible for fire to cross over to the other side through the water.

4.1.5. Material-specific combustion parameters

To determine the degree of combustion based on the class, class-specific combustion values are used. For values for the dried branches, the research conducted by Chen et al. [15, 28] is used. The values for ignition temperature, maximum mass loss temperature, and burnout temperature depend on the degree of heating. Currently, for simplicity, we use an average value. Each of these temperatures changes with tree type [43] [99], material thickness [83], and environmental factors such as heat gain [15] [91], wind [110] [92], environmental temperature, and air composition [84] [30]. Since these are

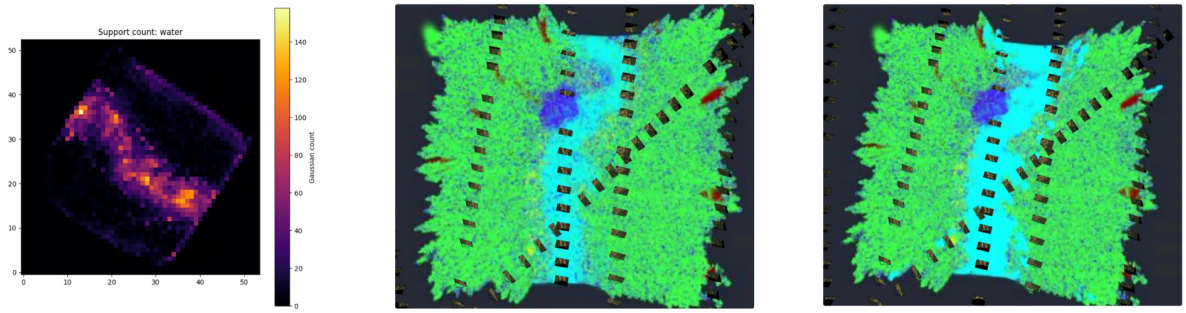


Figure 4.10: Column fill procedure for a river scene. Left: Columnised heat map of Gaussians classified as water. Middle: Class debug-view before column fill. Right: Class debug-view after column fill. The river is now solely filled with water-type Gaussians, and fire can only cross over through the air.

Table 4.1: Combustion parameters used in the wildfire simulation for the flammable scene classes. Burn rate c controls the mass-loss rate in Eq. 4.16, while the heat output parameter scales the energy released during combustion.

Class	Ignition Temp. (°C)	Extinguish Temp. (°C)	Burn Rate c
Stem	300	120	0.02
Woody branch	260	100	0.05
Live branch	220	80	0.15

almost impossible to determine with certainty for a 3DGS estimated scene, we also use average values for this. The values are shown in Table 4.1.

4.2. Second stage: Wildfire Simulation

Because the digital forest scene is a recreation, the exact physical properties of the real forest are not fully known. Therefore, a purely physics-based simulation is not feasible, and certain generalizations must be introduced. Fire spread is therefore modelled using a particle system, while the combustion process itself follows a physics-based approach.

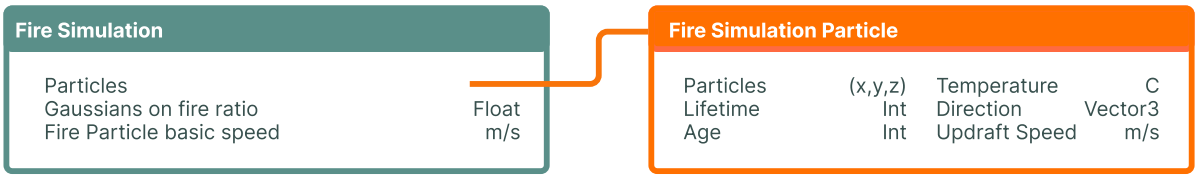


Figure 4.11: The relation between the fire simulation and the fire spread particles.

4.2.1. Fire spread particle motion

In the simulation, fire spread is modelled using a set of particles that represent the fire spread. Figure 4.11 shows the relation of fire simulation particles to the fire simulation. Each particle moves through the scene according to three components: wind advection, thermal updraught, and a stochastic motion term. The particle position is updated each timestep according to

$$\mathbf{p}_{t+1} = \mathbf{p}_t + \mathbf{v}_{wind}\Delta t + \mathbf{v}_{updraught}\Delta t + \mathbf{d}_p\, s_p\, \Delta t \tag{4.1}$$

where:

- $\mathbf{p}_t$ is the particle position at time t
- $\mathbf{v}_{wind}$ is the horizontal wind velocity
- $\mathbf{v}_{updraught}$ is the vertical updraught velocity
- $\mathbf{d}_p$ is the particle's random unit direction vector

- s_p is the particle's individual speed factor
- Δt is the simulation timestep

Wind Advection

Wind moves particles horizontally across the scene. The wind direction θ and speed w are converted into a velocity vector in the ground plane:

$$\mathbf{v}_{wind} = w \begin{bmatrix} \cos\theta \\ \sin\theta \\ 0 \end{bmatrix} \tag{4.2}$$

The wind speed can be set within the UI, and is based on the Beaufort scale [51], shown in Table 4.2

Force	Description	wind speed [m/s]
0	Calm	0–0.2
1	Light air	0.3–1.5
2	Light breeze	1.6–3.3
3	Gentle breeze	3.4–5.4
4	Moderate breeze	5.5–7.9
5	Fresh breeze	8.0–10.7
6	Strong breeze	10.8–13.8
7	Near gale	13.9–17.1
8	Gale	17.2–20.7
9	Strong gale	20.8–24.4
10	Storm	24.5–28.4
11	Violent storm	28.5–32.6
12	Hurricane	≥ 32.7

Table 4.2: Beaufort scale of wind force [51]

Thermal updraught

Heat moves upwards primarily due to convection, where heated liquids or gases expand, become less dense than their surroundings, and are forced upward by gravity, since more updraught invites more oxygen to the fire, which increases the temperature even more, etc. This is a self-reinforcing cycle [101], which is why the vertical updraught velocity is modeled as a nonlinear function of temperature:

$$v_{updraught}(T) = v_{min} + \left(\frac{\text{clamp}(T, T_{min}, T_{max}) - T_{min}}{T_{max} - T_{min}} \right)^{\gamma} (v_{max} - v_{min}) \tag{4.3}$$

where

- T is the particle temperature
- T_{min} and T_{max} define the temperature range
- v_{min} and v_{max} are the minimum and maximum updraught velocities
- γ controls the nonlinearity of the curve

The exponent γ induces the effect that the stronger the fire, the higher the temperature, which produces significantly stronger updraughts.

The resulting vertical velocity is applied as

$$\mathbf{v}_{updraught} = \begin{bmatrix} 0 \\ 0 \\ v_{updraught}(T) \end{bmatrix} \tag{4.4}$$

Random Particle Direction

Like every particle system, particles are assigned a random direction, which is a vector sampled uniformly on the unit sphere. Sampling directly from a sphere avoids directional bias that would occur if each axis were sampled independently.

The direction vector is defined as

$$\mathbf{d}_p = \begin{bmatrix} \sin\theta\cos\phi \\ \sin\theta\sin\phi \\ \cos\theta \end{bmatrix} \tag{4.5}$$

where $\theta \in [0, \pi]$ and $\phi \in [0, 2\pi]$ are random spherical angles.

Temperature-Dependent Particle Speed

Particles also receive an individual speed factor, within a range, depending on the temperature of the particle

$$s_{base}(T) = s_{min} + \left(\frac{\text{clamp}(T, T_{min}, T_{max}) - T_{min}}{T_{max} - T_{min}} \right)^{\gamma} (s_{max} - s_{min}) \tag{4.6}$$

To introduce variability in fire spread particle trajectories, Gaussian noise is added:

$$s_p = \text{clamp}(s_{base}(T) + \epsilon,\ 0.7,\ 1.3) \tag{4.7}$$

where

$$\epsilon \sim \mathcal{N}(0, \sigma^2) \tag{4.8}$$

This component is included to prevent particles from following identical trajectories.

Particle Lifetime

Each particle exists for a finite number of timesteps L. At every update, the particle age increases by one, and the particle is removed when

$$\text{age} \geq L \tag{4.9}$$

Fire particle cap

To ensure that the fire spreads at a consistent rate regardless of the overall fire size, the maximum number of active fire particles is dynamically adjusted based on the number of Gaussians burning at that moment. Lower and upper bounds are defined for the total number of particles in the simulation. The maximum allowed number of particles is then interpolated between these bounds according to the fraction of Gaussians that are on fire relative to the total number of Gaussians in the scene. Currently, for a scene of about 70 m^2, the maximum ranges between 800 and 3000 Gaussians, but this can be adapted for larger scenes. As a result, small fires generate fewer particles while large fires are allowed to produce more particles.

Since particles are spawned from burning Gaussians, regions containing many Gaussians have more potential spawning locations. Therefore, very detailed parts of the reconstructed scene burn faster. This mirrors the effect that scene regions with more geometric detail, like branches and leaves, tend to ignite and burn faster due to increased surface area and greater exposure to oxygen.

4.2.2. Combustion: Heat Transfer

In this work, the thermal energy gain of a scene element is modelled as the sum of radiative and convective heat transfer mechanisms. Conductive heat transfer is intentionally omitted since the scene is represented using Gaussian splatting primitives, which are abstract volumetric elements that lack explicit topological connectivity. Since conduction fundamentally depends on physical contact and material continuity, it cannot be reliably defined between these independent Gaussian primitives. There are several types of fire, as shown in Figure 4.12. For this work, we only focus on crown fire, as our scenes solely include (conifer) trees. Inserting hedges and other low vegetation in the scene would enable the surface fire to be modelled as well, as the two types are similar in behaviour. Fire particles could be extended to multiple types, to account for ground fire and surface fire in the future as well.

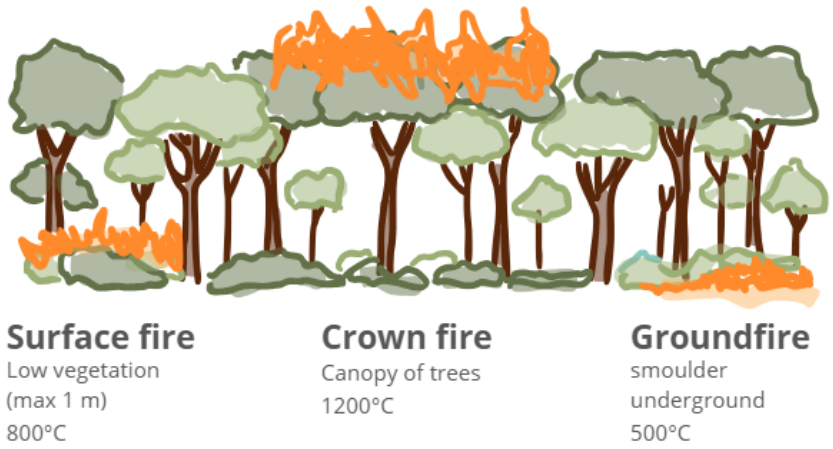


Figure 4.12: The three types of fire in a forest fire [18]. Ground fire typically progresses much more slowly and can stay burning for days or weeks. Crown fire burns the upper canopy and spreads rapidly. Surface fire burns low vegetation like bushes, and its speed is between crown and ground fire.

Firstly, the net heat flux is computed taking the two heat transfer mechanisms into account. Heat flux is the rate of heat transfer per unit area. the total heat flux q_{total} is expressed as.

$$q_{total} = q_{radiation} + q_{convection}, \tag{4.10}$$

where q is the heat flux in W/m^2.

Radiation is the thermal energy exchange via electromagnetic radiation, which does not require a medium or physical contact. It is modelled using the Stefan–Boltzmann law according to

$$q_{radiation} = \epsilon\sigma(T_{fire}^4 - T_{surface}^4), \tag{4.11}$$

where ϵ is the emissivity and σ the Stefan–Boltzmann constant.

Convection is the transfer of heat through liquids and gases, such as air, given as

$$q_{convection} = h_c(T_{fire} - T_{surface}), \tag{4.12}$$

where h_c is the convective heat transfer coefficient.

The total heat gain Q in W is calculated, for a Gaussian with effective surface area A:

$$Q = q_{total}A, \tag{4.13}$$

where q_{total} is the total heat flux. Identical effective areas for convection and radiation are assumed because Gaussian primitives lack well-defined geometric connectivity or visibility relations. The surface area A corresponds to the effective exposed area of a Gaussian primitive. This area is approximated

using the surface area of an ellipsoid. This is approximated using Knud Thomsen's formula [66] with the expression

$$S \approx 4\pi \left(\frac{a^p b^p + a^p c^p + b^p c^p}{3} \right)^{\frac{1}{p}}$$

with the least relative error (±1.061% worst case) when $p \approx 1.6075$.

Since Gaussian splats are an abstract representation of scene elements, it was decided to assign to each Gaussian the average area of the surface area of all Gaussians. Since assigning the ellipsoid surface to the Gaussians resulted in most Gaussians either burning out instantly or not setting on fire at all, it was decided that the surface area would be kept abstract.

For each Gaussian, the temperature difference ΔT is calculated via the formula:

$$\Delta T = Q \Delta t / (m \, c_{material}), \tag{4.14}$$

where Δt is the timestep, m is the mass of the Gaussian, and $c_{material}$ is the specific heat capacity of the assigned material in $Jkg^{-1}K^{-1}$.

As a result, we get the final temperature according to:

$$\Delta T = Q \Delta t / (m \, c_{material}). \tag{4.15}$$

4.2.3. Combustion: Mass Loss

Mass loss represents the pyrolytic decomposition of the material as a result of a temperature-dependent reaction. The mass-loss formulation follows the formulae from *Fire in Paradise* [35]. Mass reduction of a Gaussian element is described by

$$\frac{dM}{dt} = -k(T) \, c \, A, \tag{4.16}$$

where M is the remaining mass, $k(T)$ is the temperature-dependent reaction rate [s^{-1}], and c denotes the charring mass per unit surface area $[kg \, m^{-2}]$. We follow the previous work [35] for the specification of the charring coefficient c. Because explicit char-layer dynamics are not modelled in the Gaussian representation, this coefficient is treated as a spatially constant material parameter. A is the effective pyrolysing surface area of the Gaussian [m^2]. The reaction rate k is defined by Pirk et al. [80] as

$$k(T_G) = \eta \begin{cases} 0 & \text{if } T_G < T_0, \\ S((T_G - T_0)/(T_1 - T_0)) & \text{if } T_0 \leq T_m \leq T_1, \\ 1 & \text{if } T_G > T_1, \end{cases} \tag{4.17}$$

where T_0 = $150°C$, and T_1 = $450°C$, which denote the onset and full-activation temperatures of pyrolysis, respectively. $S(x) = 3x^2 - 2x^3$ indicates a sigmoid interpolation function. The factor η represents wind-induced cooling effects. Because wind cooling is modelled separately in the present work, η is set to 1, which equals to no wind.

4.2.4. State dependent behaviour

After the heating calculations, the temperature of the Gaussians near the fire particles has increased. Gaussians outside the influence radius remain thermally unchanged. The updated temperatures determine the subsequent combustion state transitions (ignition, burning, or cooling) as shown in Figure 4.13, which are handled in later stages of the combustion update.

Within the scene, burning Gaussians are depicted in colour depending on their temperature, which is based on the Temperature of Fire scale by the UK City Fire Protection services [54]. On top of this, a small stochastic bias is added toward hotter temperature colours to make the fire more visually interesting.

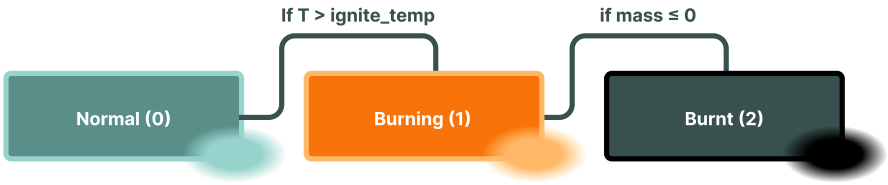


Figure 4.13: States of a Gaussian.

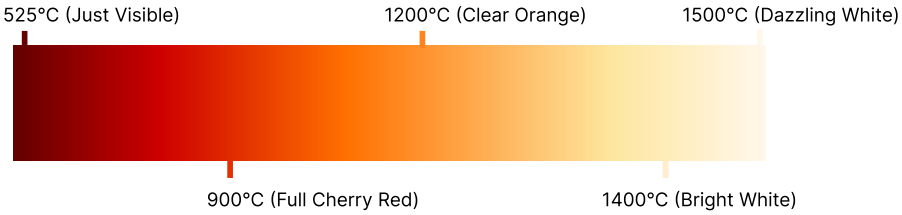


Figure 4.14: colour of Gaussians depending on temperature.

4.2.5. Scene scaling

Reconstructed Gaussian scenes do not inherently correspond to real-world spatial dimensions. For example, a real forest scene may cover an area of approximately 100×100 meters, while the reconstructed Gaussian representation occupies an abstract range such as $[-1, 1]$ in the x- and y-directions. As a result, distances in the 3DGS scene do not directly correspond to real-world measurements.

To account for this, a scaling transformation is introduced between the Gaussian scene coordinate system and real-world units. All quantities such as distances, velocities, and areas are scaled to match the real world. Then all processes that depend on spatial dimensions are calculated in real-world units. After, the results are converted back into the Gaussian scene coordinate system used for rendering and simulation.

4.2.6. Voxel hash speed-up

Within the simulation, 3DGS scenes include a lot of Gaussian Splats. The number of fire spread particles can reach high levels as well. For the update step, a naive approach is the following:

for each Gaussian: check distance to all fire particles

This results in a computational complexity of $O(n \cdot m)$, where n is the number of fire particles and m is the number of Gaussians. As both quantities can become large, this approach quickly becomes computationally expensive. Therefore, a voxel hash data structure is used to accelerate neighbour queries. The scene is discretized into a 3D voxel grid, and each Gaussian is assigned to the voxel in which its centre lies. Instead of checking all Gaussians in the scene, each fire particle only needs to query Gaussians located in its own voxel and the neighbouring voxels. With this, the complexity of neighbour queries is approximately $O(n)$, assuming a roughly uniform spatial distribution of Gaussians. Figure 4.15 shows an example of fire spread using the voxel hash.

4.3. Implementation details

The proposed system is implemented in Python and built on top of the Nerfstudio framework [102], which provides the underlying infrastructure for Gaussian Splatting [48], and Feature Splatting [86]. Figure 4.16 shows the Nerfstudio viewer, and Figure **??** shows in close up the UI adaptations for the fire simulation presented in this thesis. Lastly, PyTorch is used for numerical computations and GPU acceleration.

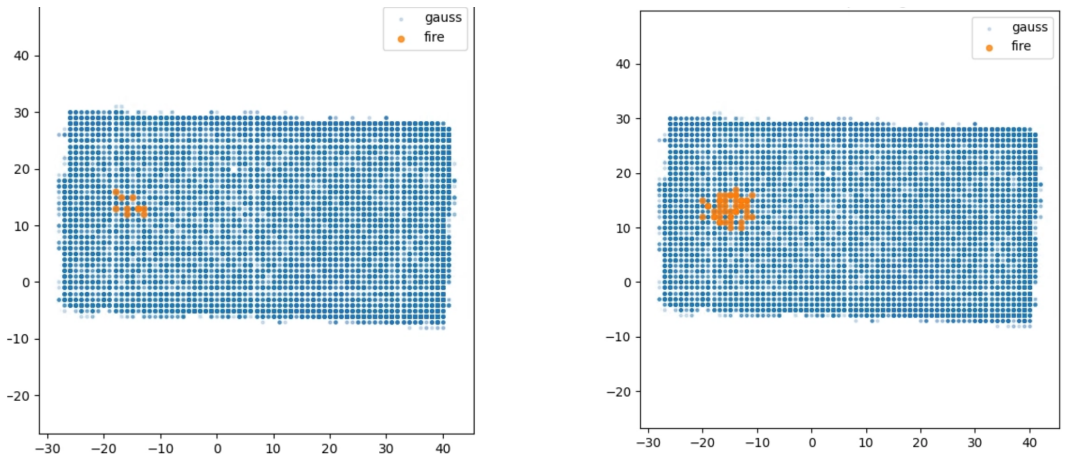


Figure 4.15: A debug top-view of the fire spread using voxel neighbours. The orange circles indicate voxels that currently contain fire particles.

Figure 4.16: The interface within Nerfstudio. On the bottom right, the UI additions for fire simulation of this thesis can be seen.

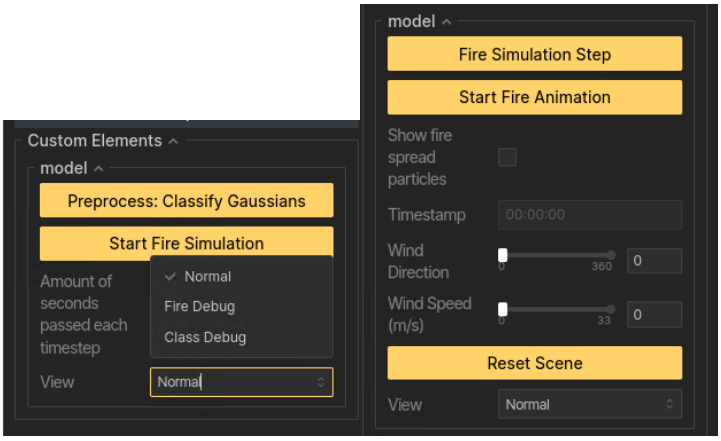


Figure 4.17: UI panel within the Nerfstudio interface, close-up. Left: the first options before the fire simulation is initialized. Right: the fire simulation controls, which appear after the fire simulation has started.

5 Evaluation

5.1. Data

For the evaluation, this work initially employs synthetic data to enable controlled development of the fire-simulation framework without being constrained by sensor noise and limited resolution. Using synthetic environments allows the combustion model to be validated under fully known conditions. After establishing correct physical behaviour in these controlled settings, the approach is later evaluated on a captured scene from the Open Forest Observatory dataset [76].

All synthetic scenes used in this report were created in *Blender* [27]. Virtual camera trajectories were designed to emulate drone footage. The renderings produce image sequences comparable to aerial surveying footage. This way, the testing scenario is comparable to the real-life scene capture pipeline, and full control over geometry, materials, and environmental parameters is maintained.

As illustrated in Figure 5.1, three types of low-polygon tree models were constructed by procedurally scattering leaves with hand-drawn textures along a central axis. Taller tree variants additionally include dry branches.

The terrain surfaces were generated using the *A.N.T. Landscape* Blender extension [26], which is a tool for creating controllable large-scale elevation structures. Trees were then distributed across the terrain using a Poisson disk distribution to prevent point intersection. A Fractal Brownian Motion [44] texture is used to determine the spawning probability of each tree type. This approach introduces natural clustering effects: nearby groups of trees grow taller as they compete for light, while sparsely distributed trees are generally smaller [56]. The spawn chance is also affected by the steepness of the terrain. Individual trees vary slightly in scale, rotation, and colour.

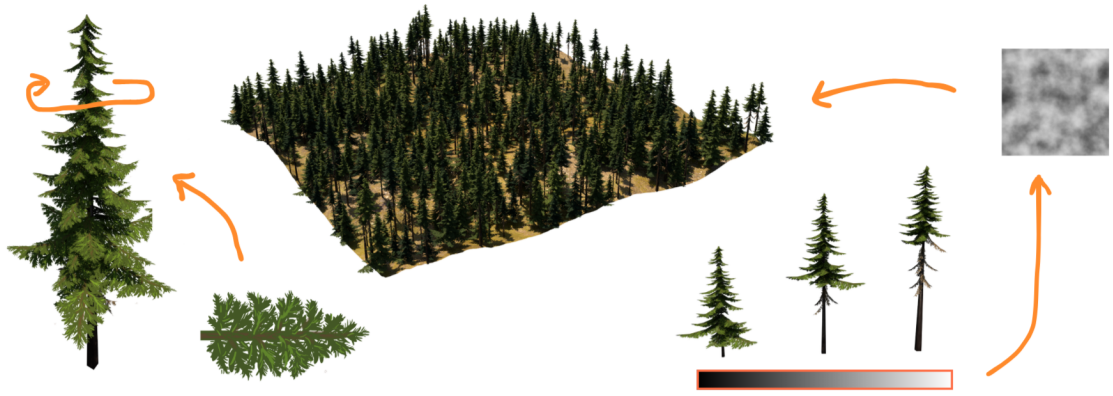


Figure 5.1: Tree and forest creation process in Blender.

5.2. Hardware and system specifications

All experiments and simulations were conducted on a Linux workstation. The wildfire simulation was executed using GPU-accelerated computations through PyTorch.The used hardware has the following specifications:

- **CPU:** 11th Gen Intel(R) Core(TM) i9-11900KF @ 3.50 GHz (16 cores, x86_64 architecture)
- **GPU:** NVIDIA GeForce RTX 3090 with 24 GB VRAM
- **System memory:** 64 GB RAM

Furthermore, our implementation has been developed under the following environment:

- **Operating system:** Ubuntu 20.04.4 LTS
- **Python:** Version 3.10.18
- **CUDA:** Version 11.4
- **PyTorch:** Version 2.1.2 (CUDA-enabled)
- **NeRFStudio:** Version 1.1.5

5.3. Evaluation protocol

Qualitative analysis

The qualitative analysis strategy is based on prior work on wildfire prediction and spread modelling, such as [119] and [35]. Direct quantitative comparison with existing approaches remains challenging, as many published systems do not include publicly available code and frequently differ in input data types, modelling assumptions, and intended outputs. Therefore, this section assesses whether the simulated fire behaviour reproduces physical and ecological patterns reported in wildfire research. Such patterns include slope-dependent spread, wind-driven elongation of the fire front, fuel-density effects, and obstruction-limited propagation.

Forest Cover

This experiment is inspired by the forest cover tests presented by FiP [35]. Fire spread is compared across three scenes with different vegetation densities of 25%, 45%, and 90%. All other environmental conditions were kept constant. In Figure 5.2, the timestamped frames show that higher fuel density leads to faster, more continuous fire propagation, where the fire in a 90% density forest progresses faster through the scene. The sparser the forest distribution, the less even the fire propagation. This behaviour aligns with real-world wildfire dynamics, where fuel availability strongly governs spread rate. To quantify and illustrate the fire behaviour, two graphs were created. The first, shown in 5.3 is a graph showing the percentage of burning Gaussians over time. Several effects can be deduced from this. A denser forest burns faster, but also burns out more quickly, whereas a sparser forest burns much more gradually. The smoothness of the curve also shows that a sparser forest burns less consistently, where the curve is ragged compared to a smoother curve for denser forests. The second graph in Figure 5.4 shows the burnt mass over time. It shows how denser forests burn more quickly.

As rendering is performed in a separate thread from the fire simulation, the same render time can result in a different time in simulation, depending on the intensity of the fire. Therefore, a log file was kept to keep track of the timestamps each frame belongs to, along with simulation data for plots. For fires covering large areas, more calculations are needed. Hence, the differences in lengths in the top graph in Figure 5.3.

Slope influence on fire spread

The next test mirrors experiments from previous fire simulation evaluations [119] [35] in verifying whether the simulation adheres to the effect that fire travels faster uphill versus downhill. Figure 5.5 shows the test fire movement at different time steps, for different angled slopes.

Two main observations can be made. Firstly, the results confirm that fire spreads more rapidly in the uphill direction and more slowly downhill, which is consistent with findings reported in [119] [35]. Secondly, the graph shows that the rate of increase in fire spread speed grows with slope steepness

in the uphill direction. In contrast, the downhill case exhibits a more linear relationship between slope angle and spread rate. This behaviour is visualized in Figure 5.6.

These observations are consistent with existing literature. To this end, Rothermel's Rate of Spread (ROS) must first be explained. This is an empirical formula developed by Richard C. Rothermel in 1972 to predict the forward speed of a wildland surface fire. Subsequent research has further analysed the contribution of each of these factors. For example, Geng et al. [31] investigated the spread of surface fires in Pinus koraiensis needle beds under no-wind conditions across varying slopes. Their results show that as the slope increased from 10° to 20°, the ROS increased by a factor of 0.51, while an increase from 20° to 30° resulted in a factor of 1.18. This indicates a non-linear, accelerating increase in spread rate with slope, which supports the observation from our simulation. Nevertheless, further evaluation is required to draw definitive conclusions.

Wind effect

To test the wind effect, two tests were conducted. Both on a flat forest scene of 90%, as the high tree coverage removes the effect that tree placement has on fire spread, as the scene is practically evenly scattered with fuel. Firstly, the scene was ignited with different wind speeds to show the effect on fire spread. Figure 5.7 illustrates the effect of different wind speeds, for the same ignition point, and the same wind direction. This shows that fire speed increases with wind velocity, the fire front becomes more elongated in the wind direction, and the fire burns out faster. This confirms that the implemented wind model produces the expected scaling of spread rate with wind strength. Figure 5.8 shows the fire simulation with four different wind directions.

Forest Obstructions

The following section covers the effect of natural or man-made gaps in a forest. These evaluation methods have not been adapted from other papers but are novel. In nature, natural breaks in a forest, like rivers, can stop fire from spreading. To first test this, a synthetic forest scene including a river has been created. Figure 5.9 shows the effect. In cases of severe wind, fire can still spread over these gaps.

There are also man-made forest gaps, made to stop fire from spreading in case of a forest fire. There are many firefighting organisations that have formalized a protocol for the size of these gaps, to make sure fire is prevented from spreading. These gaps are called firebreaks and fuel breaks.

Firebreaks and fuel breaks are sections in a forest where vegetation is removed or reduced to interrupt the continuity of combustible material.

A firebreak is typically a strip of land in which vegetation is removed down to bare mineral soil. Operational guidelines indicate widths on the order of 6 to 10 meters or greater, depending on local conditions. Additionally, the guidelines recommend a minimum width proportional to nearby fuel height, commonly two to three times the height of the tallest adjacent vegetation [22, 1]

Fuelbreaks (or shaded fuelbreaks) operate at a larger scale, and reduce vegetation density. The required width depends strongly on the slope of the terrain, wind-driven spotting due to embers, and fuel characteristics. Recommended widths may reach several hundred feet; for example, approximately 90 meters on level terrain has been suggested in prior forestry guidance [22]. In addition to width, horizontal crown separation is also critical. A minimum spacing of roughly 2.5 meters between tree crowns is recommended on flat terrain, and increases with slope [1].

To test whether our simulation holds true to these guidelines, simplified analytical rules are adopted to calculate the widths for fuel and fire breaks for our scenes.

Firebreak width is modelled as a bounded function of local fuel height:

$$W_{\text{fire}} = \text{clamp}(3H_{\text{fuel}},\ 0.6\ \text{m},\ 2.6\ \text{m})\,. \tag{5.1}$$

Fuelbreak width is governed by slope-driven flame extension, wind-driven spotting, as well as a minimum size depending on vegetation height. To ensure containment under worst-case conditions, the required width is defined as the maximum of these contributing factors:

$$W_{\text{fuel}} = \max(W_{\text{slope}},\ W_{\text{wind}},\ 3H_{\text{fuel}})\,. \tag{5.2}$$

Figure 5.10 shows the schematic design of a firebreak for a situation without wind. The road in the centre does not influence fire behaviour; it is just for realism, as roads are often used as fire break locations for accessibility reasons, and it shows the centre of the scene in our case.

First, the results of the firebreak test without wind are analysed. This can be seen in Figure 5.11. It is clear that, without the wind, the fire does not spread to the other side. However, it is evident that as soon as there is some wind, the fire does spread. In this scene, all Gaussians beneath the ground level were classified as ground Gaussians, as grassy Gaussians are often still classified as leaves. This decision was made so the simulation is not influenced by the classification prompt setting and solely predicts forest burning.

Figure 5.12 shows that up to a wind speed of 15 m/s, fire does not spread.

The conclusion that can be drawn from these tests is that the wind has slightly too much effect on the movement of particles, as the fire does spread at wind speeds that are not particularly strong. This can be taken into account in further developments of the model, as it is easy to adjust the extent to which the wind influences the fire particles. Currently, the effect was 0.1 times the wind speed [19], but this appears to be too strong for our simulation and must be reduced.

Quantitative analysis

Burnt fuel

It is interesting to investigate whether this model can predict the total amount of burnt mass in a scene, as this relates to the amount of harmful gases released into the atmosphere.

To verify whether this is correct, it can be predicted from the synthetic scene that the scene has about 300 trees, each tree around 8 m tall. Each tree has, on average, 800kg of dry biomass [60]. In typical forest fires, large trees rarely burn completely. While small branches and foliage (up to 10mm diameter) are often 100% consumed, the trunk may only lose 5% or less of its mass due to its density and thermal resistance. If we assume 30% of the tree is fully burnt, 72.000 kg is predicted to be burnt. Currently, without stem insertion, the simulation predicts only 1500 kg of mass to be burnt. As Gaussians mostly model the surface, it makes sense that the mass is far below the predicted value. The effect of stem insertion on the predicted mass should repair this a bit. After stem insertion, the system predicted that about xxx kg of mass was burnt. This is still not enough, though it is more realistic. This can be further investigated. More stem Gaussians can be added per tree; the mass per stem Gaussian should be recalculated.

Consideration has been given to fine-tuning the mass, starting with a single tree. However, since a single tree splat scene contains far more Gaussians than the trees in the forests of the test scenarios, this has not been done for now, as it would involve fine-tuning on a different scale. However, this would certainly be a useful solution in the future.

Figure 5.13 shows the amount of mass burnt over a simulation.

Risk prediction on a real forest scene

A crucial test is to determine whether and how the fire simulation works with real forest images. For this purpose, mission 001210 from the Open Forest Observatory drone dataset was used [75]. The LiDaR point cloud was taken as the point cloud input, as well as the camera intrinsics and extrinsics from the drone measurements. Some preprocessing was required. Firstly, the scene was scaled down because most 3DGS scenes in Nerfstudio have a local size of around 2 abstract units. The camera coordinates also had to be converted to local coordinates.

Figure 5.14 shows that a realistic 3DGS scene can be created from this input. Next, a fire simulation was performed on the scene, as shown in Figure 5.15. Currently, the scene was very tilted, which made the fire propagate upwards.

5.4. Ablation study

It is important to test the consistency of the forest fire model, as it is a stochastic simulation because it depends on the movement of particles with a randomized direction. Therefore, 5 consecutive runs were performed on the forest scene with 45% canopy cover. Figure 5.16 shows that the values for burnt mass and percentage of burning Gaussians are very similar across each run, with a small standard deviation.

5.5. discussion

Overall, the fire simulation shows realistic baseline fire behaviour. The results show promising fire spread dynamics influenced by forest density and composition, The fire behaviour is realistic regarding the influence of wind speed and wind direction. Additionally, the model captures the accelerated burnout of fire particles under stronger wind.

The forest obstruction tests, which are introduced in this thesis, provide valuable additional evaluation. While the wind experiments suggest that the wind behaviour is plausible, the obstruction tests verify more strictly whether the wind influence is over- or underestimated. The river scene shows realistic behaviour, but for the formalized forest- and firebreak scenes, the fire can cross with a much lower wind speed. This can be explained by the presence of flammable Gaussians near the ground. Although this may initially seem undesirable, it reflects the fact that real-world firebreaks are rarely completely devoid of combustible material. Even though the firefighting instructions state to remove all flammable materials, for large-scale operations it is realistic that some flammable material is still left at the scene. These tests showed that fire particles are influenced too strongly by the wind parameters. In the future, these tests offer a useful mechanism for tuning wind influence to better match expected firebreak behaviour.

It was also observed that lower wind speeds were sufficient for the fire to cross firebreaks compared to crossing a river of similar width. This difference can be attributed to the presence of these intermediate flammable Gaussians in the firebreak scenario. The river scene has a larger strip of completely non-flammable material due to the column-fill procedure, which shows the worth of that preprocessing step.

In addition to the column filling step, the entire preprocessing pipeline significantly improves the robustness of the system. The treetop detection method proves to be robust to both scene scale and ground slope. This enables stems and branches to burn more slowly than the live branches. This distinction also improves the estimation of burnt mass.

The burnt mass estimation test is another novel contribution of this work. Similar to the wind tests, while the visual fire appears realistic, a more quantitative analysis is useful to more accurately test the fire behaviour. The current results indicate that the predicted mass loss is not yet fully accurate. Further calibration is required, particularly in balancing Gaussian mass, surface area, and material-specific burn rates. This evaluation step provides an opportunity for tuning these parameters in future work.

Finally, the application to real drone data shows insight into the practical applicability of the system. The proposed system is able to run a fire simulation on a real forest section, which is promising. Even though the preprocessing pipeline is designed to be capable of processing real-world scenes, the results remain affected by noise and inconsistencies. The performance of classification is reasonable, though class-specific prompts might require further refinement. It was noticed that the larger-scale scenes also introduce additional challenges. A hierarchical structure may improve simulation stability and speed.

Overall, the results demonstrate a promising and robust fire simulation pipeline. Given that this pipeline has the additional challenge of not running on simulation-ready scenes, but starting from scratch, from an image collection of a forest, the performance is robust and relatively good. All tests inspired by other fire simulation papers performed well. The novel evaluation methods show some fine-tuning is needed in certain areas. The wildfire simulation is also applicable to real-world data, although there is still room for improvement in this regard.

5.5.1. Limitations

Influence of scene reconstruction quality

One bottleneck of the accuracy of the wildfire simulation is the quality of the reconstructed Gaussian scene. If the reconstructed geometry or camera poses are inaccurate, the resulting Gaussian representation may be spatially incorrect. Since the combustion simulation operates directly on this representation, reconstruction errors propagate into the fire behaviour prediction.

In particular, the preprocessing stage is crucial, since camera poses and the initial point cloud need to be accurate to generate a consistent Gaussian scene. This can be circumvented by following the data acquisition protocol, but the risk remains.

In this work, camera poses are estimated using COLMAP [95]. COLMAP does not explicitly exploit temporal information or image ordering. Since in drone footage the image order is sequential, incorporating this information can improve pose estimation. Furthermore, recent approaches such as DUSt3R [113] or other learning-based reconstruction methods may provide more robust reconstructions.

Incorporating LiDAR data could also significantly improve the initial point cloud used during reconstruction.

Semantic classification

The accuracy of the wildfire simulation also depends on the quality of the semantic classification of Gaussian primitives. If an object cannot be correctly classified, it may be incorrectly treated as flammable or non-flammable, which can significantly alter the predicted fire behaviour.

In the current implementation, classification is performed using CLIP-based text prompts [86]. It was noted that the choice of prompts has a large influence on the classification of the scene. Experimentation with prompt engineering could improve the reliability of the semantic segmentation.

Another possible improvement is the incorporation of annotated training data. Datasets such as the FOR-instance dataset provide labelled examples of forest structures and could be used to fine-tune the classification model for specific classes such as tree stems, branches, rocks, or vegetation. However, training on a limited dataset introduces the risk of overfitting to specific forest environments. One advantage of the current approach is its reliance on vision–language models, which generalize well to unseen scenes. A hybrid approach may therefore be beneficial, where key classes are trained using annotated data while the remaining classes are identified through text prompts. It must be noted that most of the classification was tested on synthetic data, which most probably does not match the real images that the vision language models were trained on. This should perform better on real images.

Lastly, most vegetative Gaussians are either classified as 'live branch' or 'stem'. Apart from the stem insertion, most of the time, no 'dry branch' Gaussians exist. Therefore, it might indicate that the vision language models cannot distinguish at that level of detail, and the class of dry branch can potentially be discarded.

Fire simulation

The fire simulation depends on a lot of parameters, such as the spawning chance of a Gaussian on fire, the range of the lifespan of a particle, etc., which influence the fire behaviour very much. Without a real forest fire example, it is difficult to find an optimal combination of these.

5.6. Unsuccessful attempts

In the classification, an attempt was made to use negative prompts in addition to positive text prompts, and then to use a softmax to determine which class a Gaussian belongs to. This only resulted in poorer performance, probably due to the fact that there are far too many words to describe the inverse of a class. Especially since many classes are very similar. If, for example, a negative prompt of a woody branch was "tree trunk", this also pushed the semantic meaning away from woody branch. For this reason, negative text prompts have not been used for the time being.

Another interesting find was made in the slope scenes. For visual purposes, similar to FiP, a block of ground was added below the slope to present it as a slice of terrain as shown in Figure 5.17 A.

Where in FiP this did not influence the fire simulation, as only the modules of trees burn, it did affect fire propagation in this model since all elements within a scene are classified as a material. Within the ground, there are some Gaussians still classified as flammable materials, so part of the fire propagation capacity was effectively “consumed” by the ground. Although the number of fire particles scales with the amount of fire present in the scene, the slowdown effect remained noticeable. This resulted in the fire progressing a bit slower through either steep slopes compared to the flat slope which had limited to no ground Gaussians. Since real-world forest scans do not include such artificial ground blocks, they were removed from the scene and tests were performed on scenes like shown in Figure 5.17 B.

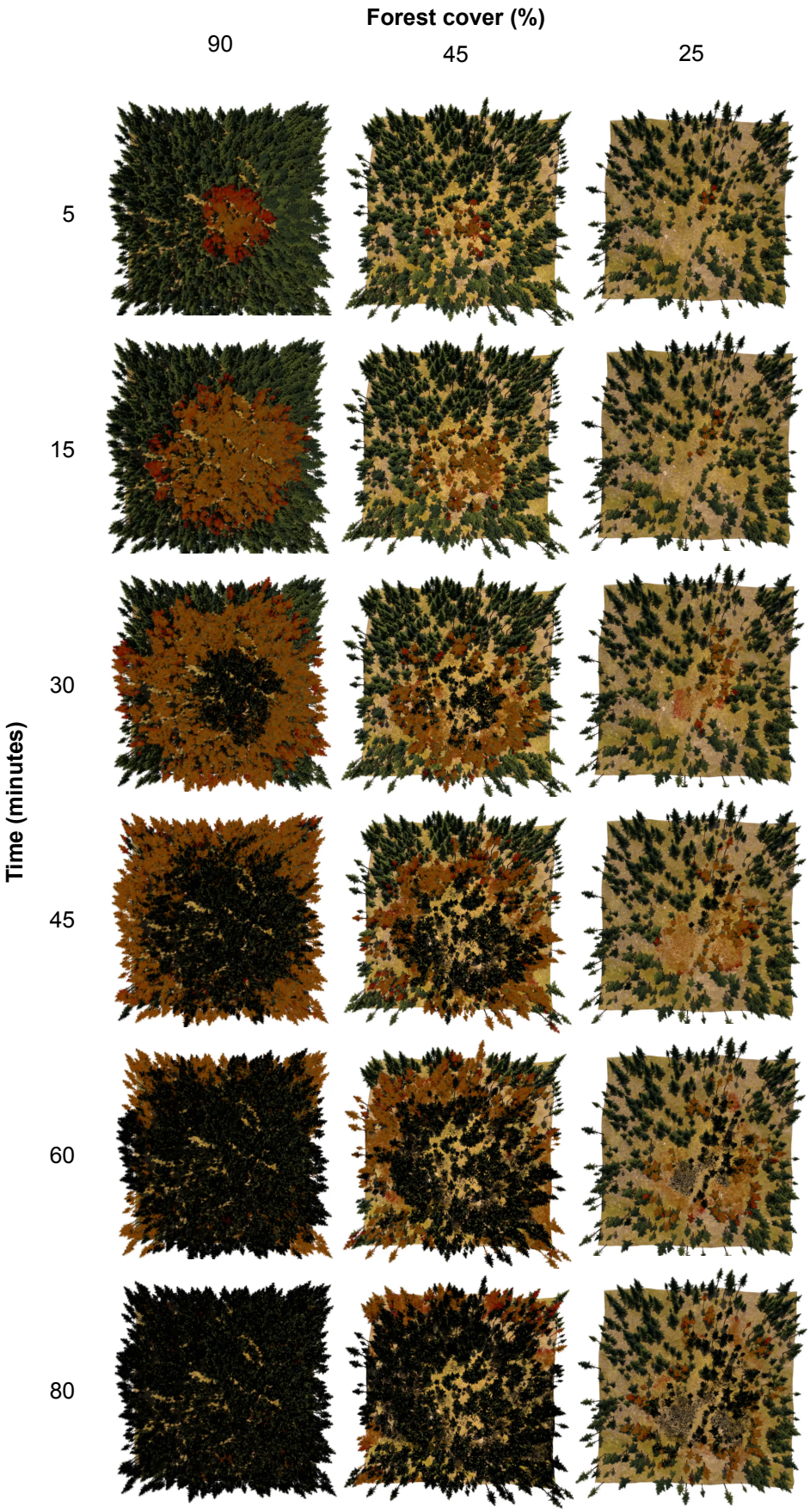


Figure 5.2: Fire spread over time for different forest cover densities. Each scene spans 60x60m. This complies with the expectation that the denser the forest, the faster and more evenly fire spreads, whereas for denser forests, fire progresses less uniformly.

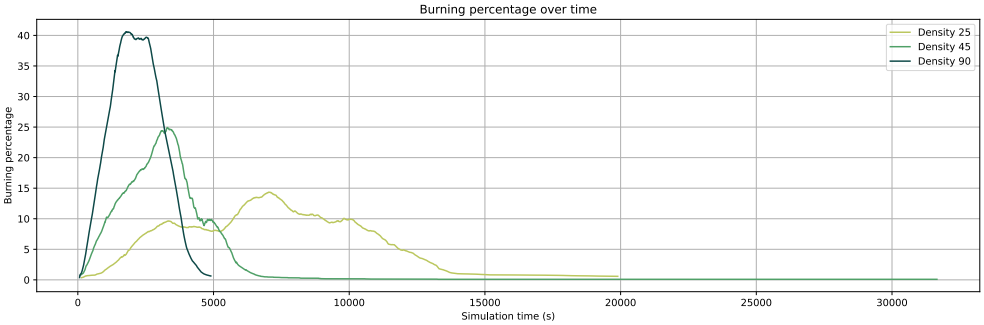


Figure 5.3: The percentage of Gaussians burning at a certain timestamp for scenes with respective forest coverage of 25%, 45%, and 90%. The plot is normalized. As the scene is a square, the percentage decreases after Gaussians start to burn out, until no burning Gaussians are left. The graph confirms that for dense forests, fire spreads faster and burns out faster. The sparser the forest, it can also be seen that the graph is less smooth, as the spread is less uniform.

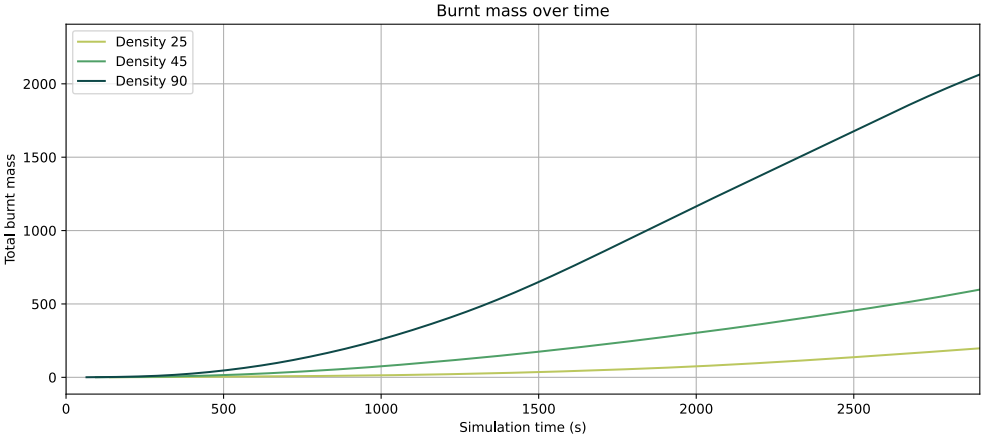


Figure 5.4: The total amount of burnt mass in kg over time for scenes with respective forest coverage of 25%, 45%, and 90%. This shows how denser forests burn more quickly.

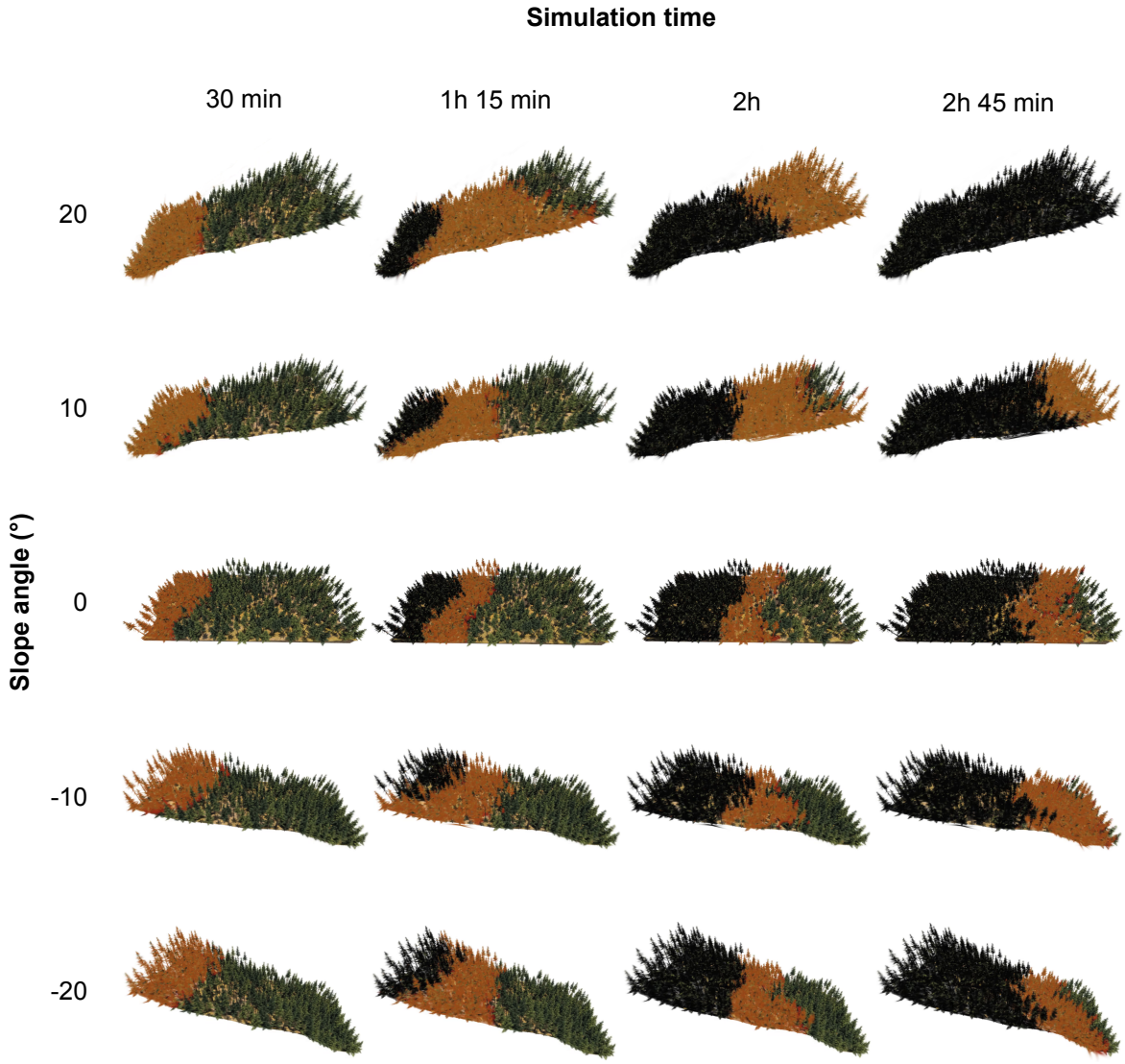


Figure 5.5: The forest-fire simulation was evaluated on five terrains with different slope angles of $-20°, -10°, 0°, 10°, 20°$ degrees, respectively. Each scene spans 50x100m. In all cases, ignition occurred at the same (x,y) location, and no wind was applied. Each row corresponds to one slope configuration, while columns show vertical snapshots at timestamps after ignition. This shows the effect of the slope on the spreading speed of the fire, where fire moves faster upwards, and slower down a slope, which matches literature.

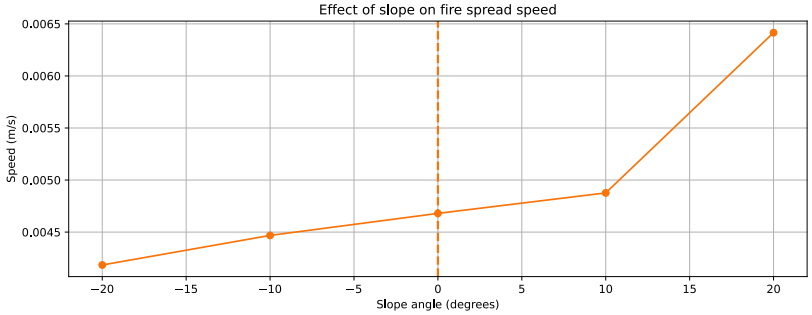


Figure 5.6: The speed of fire spread for the different slope angles of $-20°, -10°, 0°, 10°, 20°$ degrees. This shows an accelerating increase in speed for upwards slopes, while downwards, the speed reduces approximately linearly. This result is promising as it matches findings by Geng et al. [31] on the ROS.

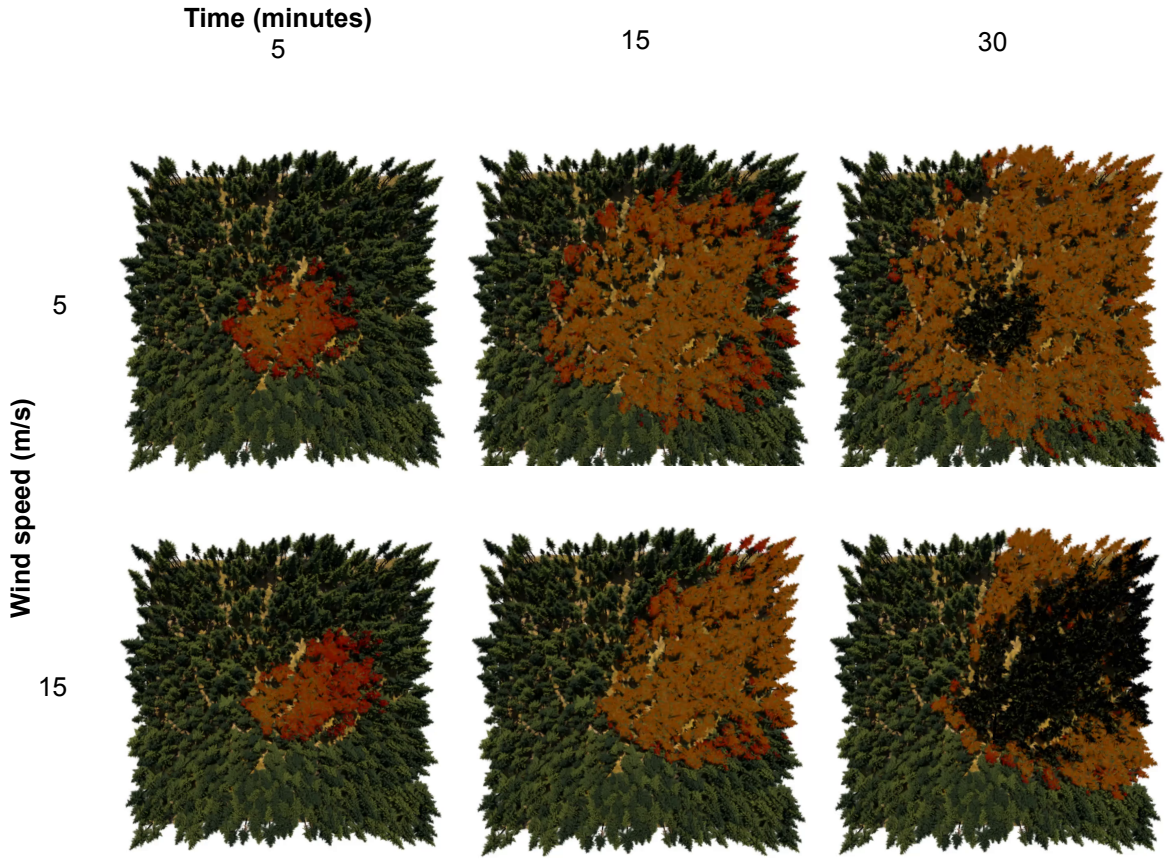


Figure 5.7: The same scene was simulated under different wind speeds (5 and 15 m/s, both in the same wind direction), with frames taken after a fixed duration. The scene scale is 60x60m. As wind speed increases, the fire front becomes more elongated in the wind direction, and the fire burns out faster.

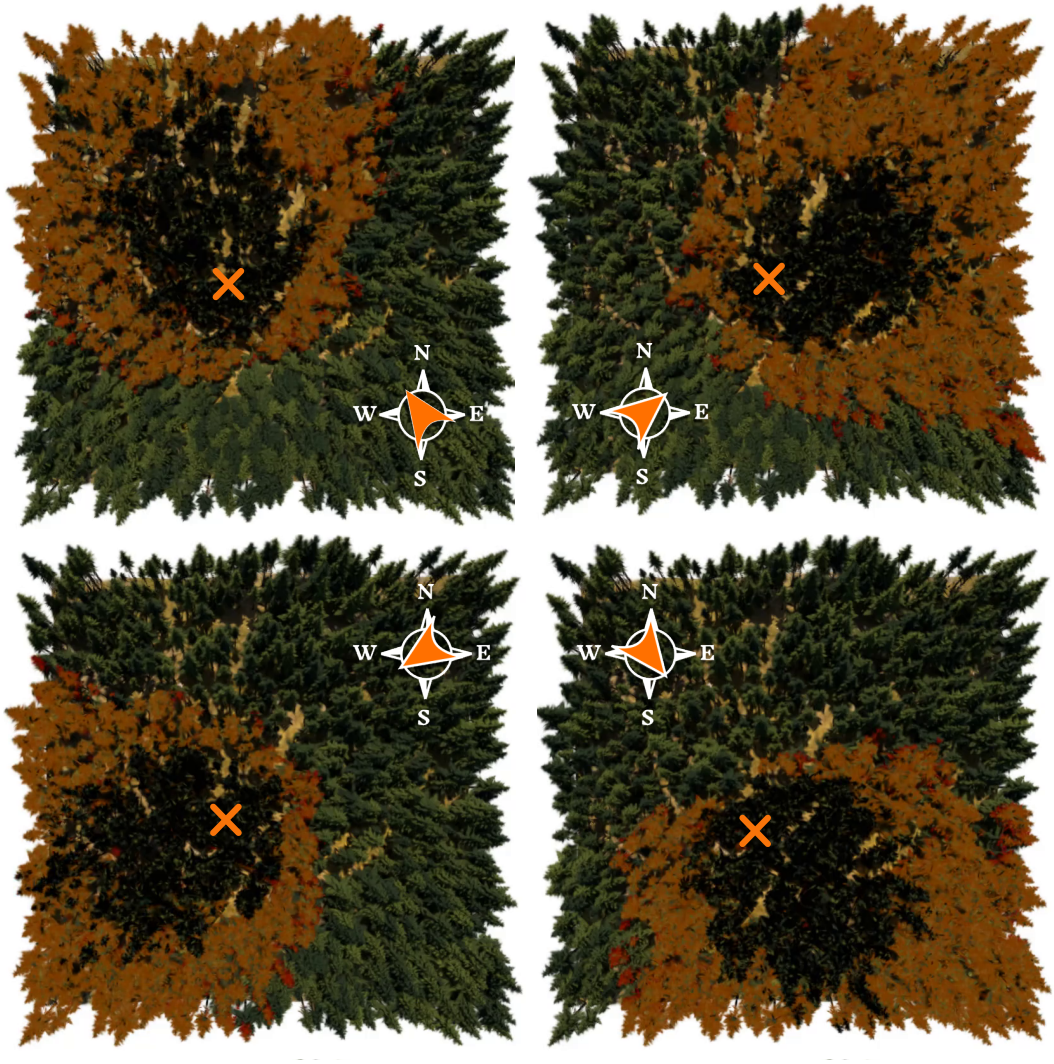


Figure 5.8: Fire spread under four different wind directions. Wind speed is 10m/s or 5 on the Beaufort scale: ”Fairly strong wind / A brisk breeze”. The overlay indicates the fire starting positions with an x, and the wind direction with an arrow, for each simulation. The scene scale is 60x60m.

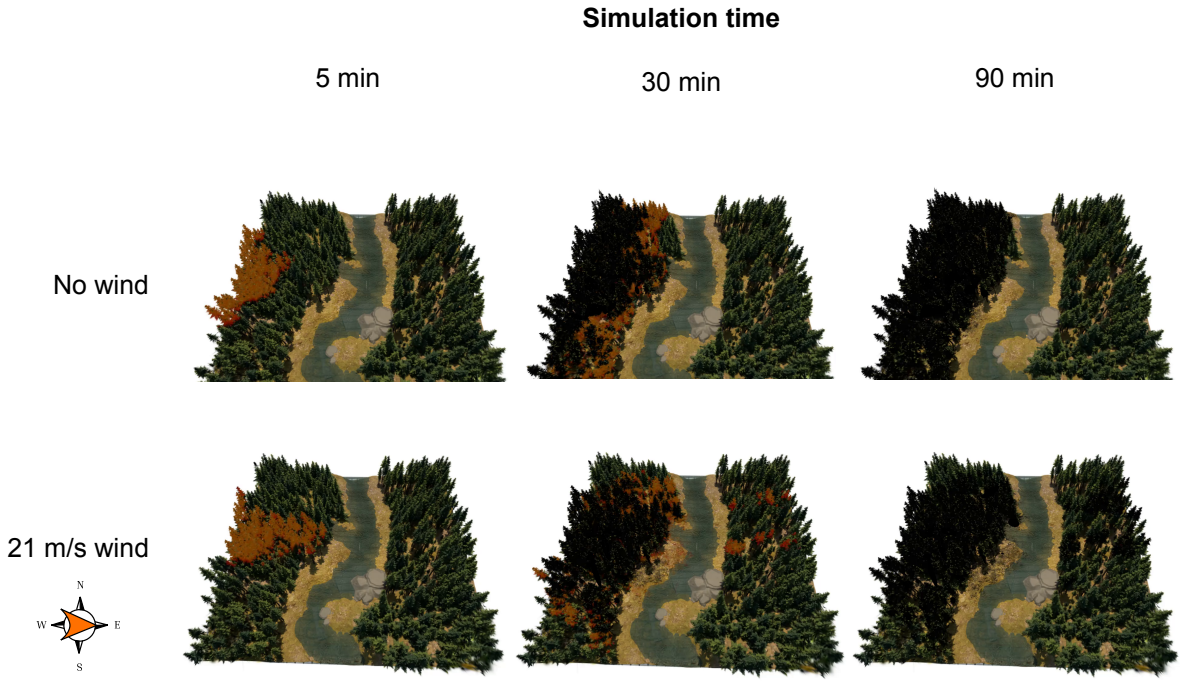


Figure 5.9: Example of a natural forest gap, namely a river, stopping fire spread. The simulation acts as expected. In cases of strong wind, such as 21 m/s or 'strong gale', fire can still cross. The wind direction is perpendicular to the river, as indicated by the arrow. This scene spans 170x170m.

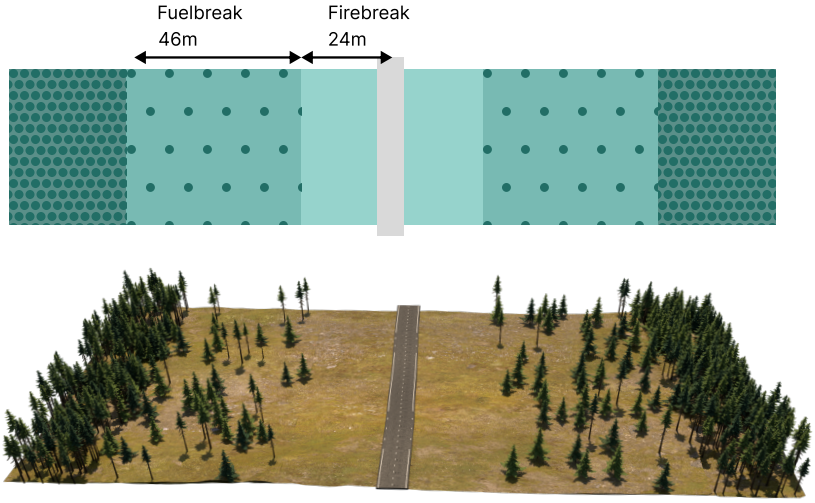


Figure 5.10: A schematic overview of the structure of a fire/fuelbreak. Starting from the centre, an area strip is completely cleared; this is the firebreak. Next comes a strip known as the fuelbreak, in which all low vegetation is removed, and the density of trees is reduced. There is a minimum distance between trees. The bottom figure shows how this translates into a test scene.

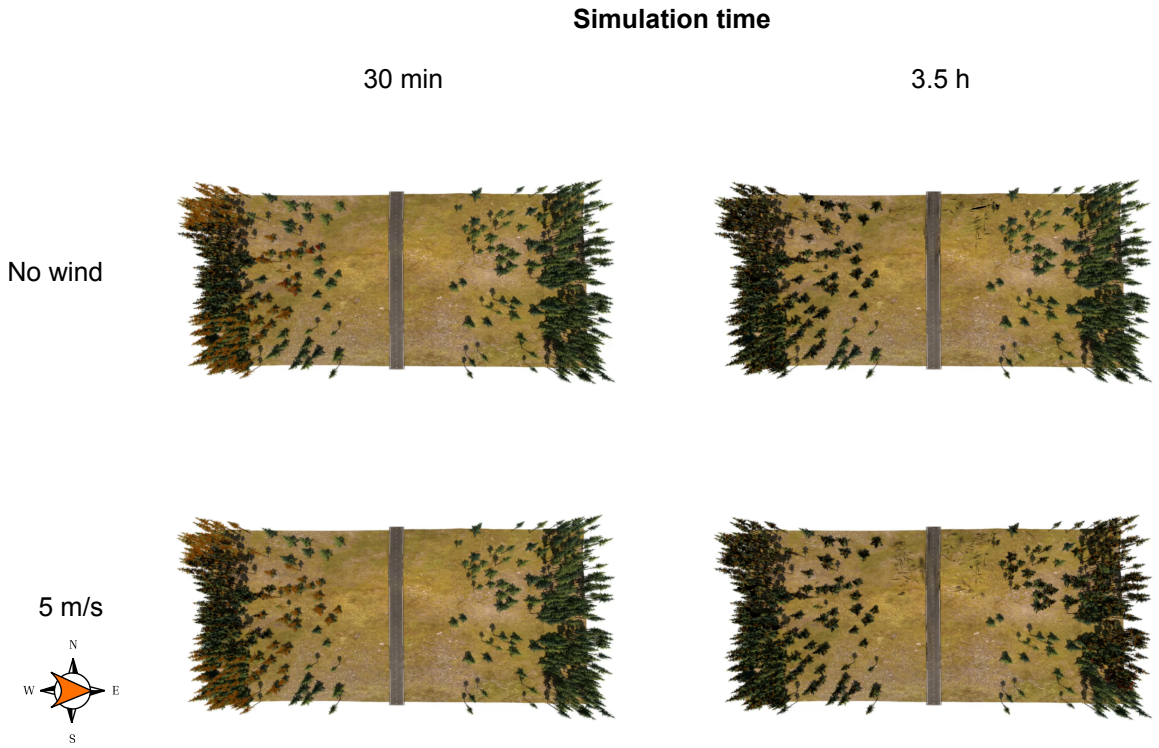


Figure 5.11: Fire spread comparison over a firebreak calculated for a no-wind situation. As expected, with no wind, the fire does not cross, but with the addition of wind in the direction indicated by the arrow, the fire can still cross to the other side.

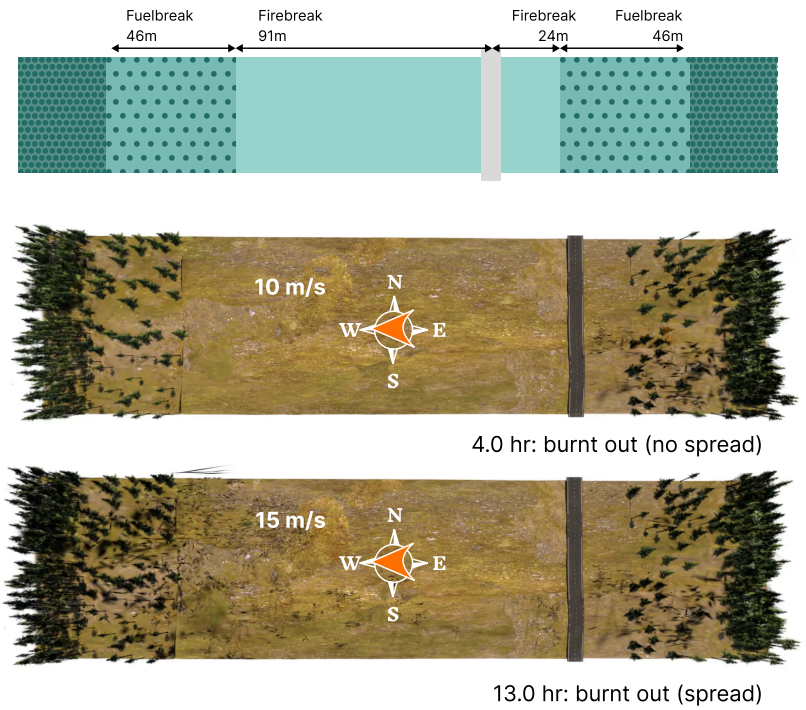


Figure 5.12: Firebreak designed for perpendicular wind. Around 15 m/s wind speed in the direction of the arrow, the fire crosses over to the other side.

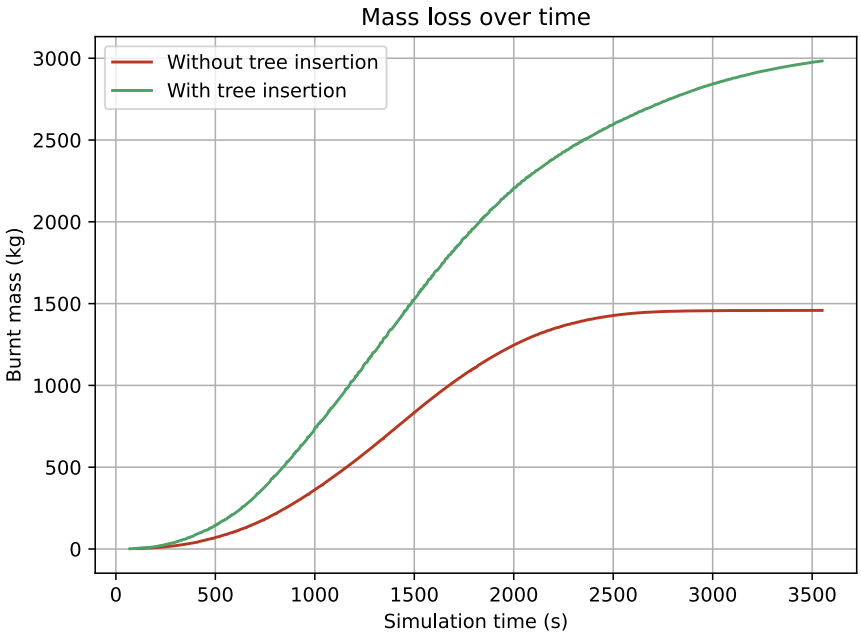


Figure 5.13: The mass burnt over two simulations, one without stem insertion, and one with stems and dry branches inserted at detected treetops.

Figure 5.14: Renders from different angles of the reconstructed splatted scene based on drone images from the Open Forest Observatory [76].

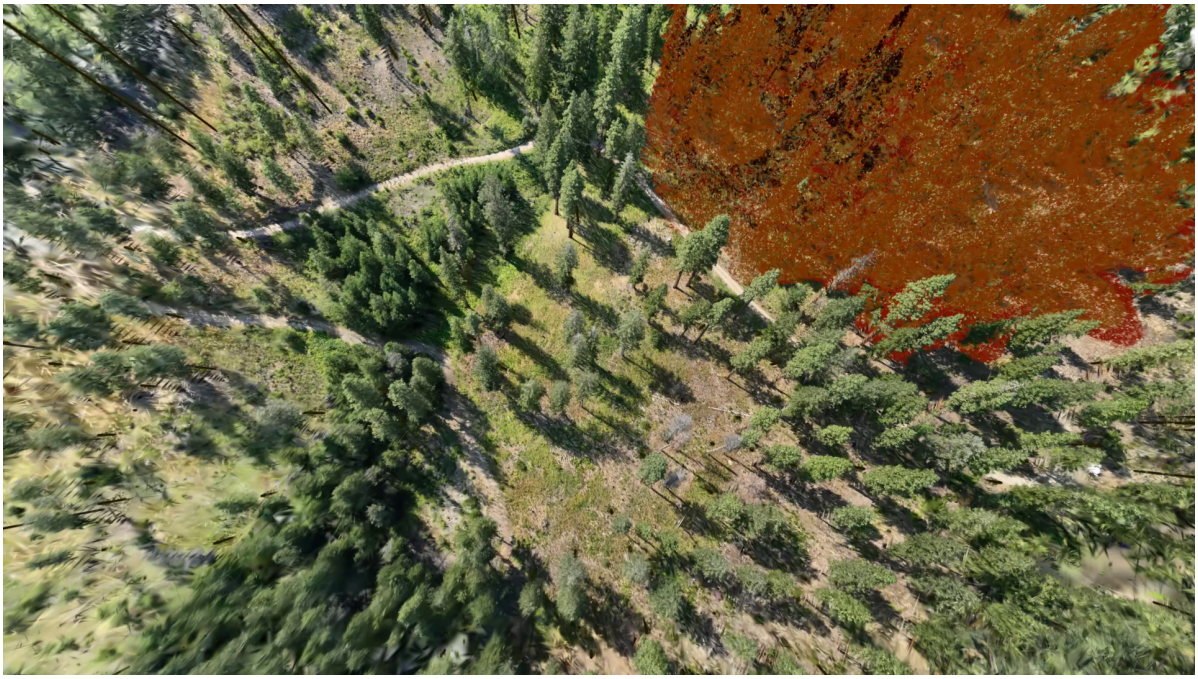


Figure 5.15: Fire simulation running on the real forest dataset.

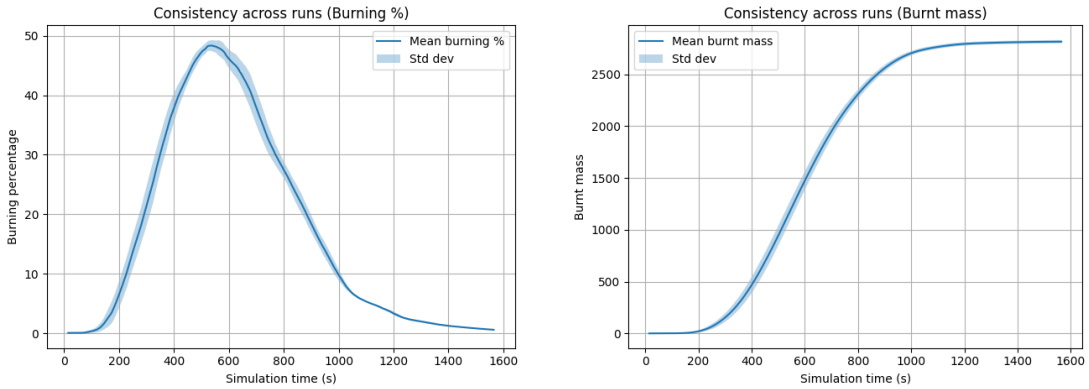
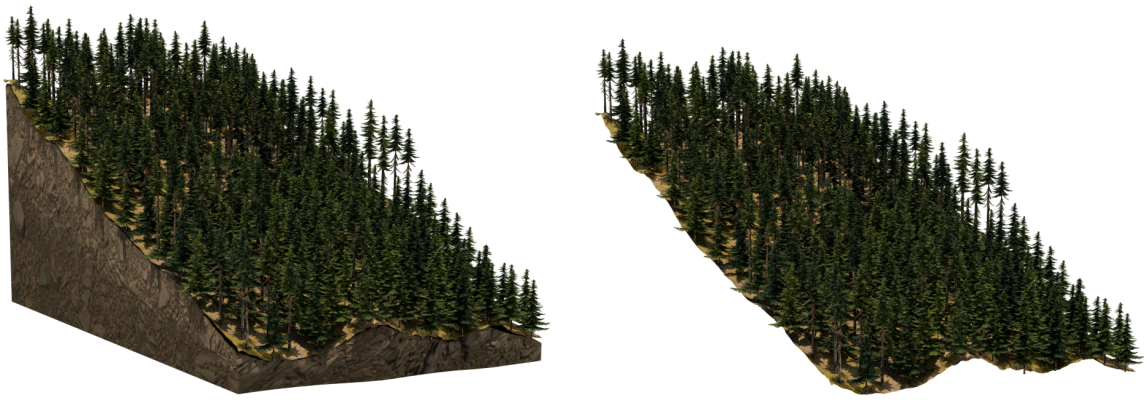


Figure 5.16: Simulation consistency across 5 separate burns. The standard deviation is plotted against the mean. The plots show the fire simulation burns consistently across multpiple runs.



(a) With ground block (b) Without ground block

Figure 5.17: Effect of adding an artificial ground block to slope scenes. The presence of a ground block leads to unintended fire propagation through the ground, which slows down fire spread compared to scenes which do not have it.

6

Conclusion

Overall, the evaluation shows that the proposed wildfire simulation reproduces several characteristic behaviours reported in wildfire literature. The synthetic scene experiments show that the model responds realistically to variations in environmental factors. Firstly, higher vegetation density leads to faster and more continuous fire propagation, and the slope experiments confirm that fire spreads more rapidly in the uphill direction. The wind experiments show that increasing wind velocity increases the spread distance. From the novel forest obstruction tests, including the fuel- and firebreak scenes, it can be deduced that the particles are affected slightly too much by the wind speed. The tests are useful for evaluating more quantitatively whether the extent of wind effect is realistic, and can be used for further finetuning. The slope scene tests provide very useful insights. They not only prove the fire progresses faster up a slope compared to down a slope, but also show an accelerated speed increase that appears only upwards, which is supported by literature. The evaluation also shows that the proposed simulation performs consistently and stably across repeated runs. The experiments on real-world data also demonstrate that the approach can be applied to reconstructed forest scenes obtained from drone imagery. However, practical challenges still exist, such as noise and the larger scale. Lastly, the quantitative estimation of burnt biomass reveals that the current model underestimates the total burned mass compared to a rough analytical prediction.

Overall, the results suggest that the proposed framework can reproduce realistic qualitative wildfire dynamics and operate on both synthetic and real-world scenes.

Furthermore, this wildfire simulation framework offers a lot of opportunities and directions for potential improvement.

One essential direction of improvement is the integration of real wildfire observations. If the temporal data of real fire spread becomes available, the simulation parameters could be calibrated or trained using these observations. This would enable the creation of digital twin scenarios in which the simulated fire behaviour is aligned with real-world fire dynamics.

Another potential extension is the introduction of hierarchical semantic classification. Instead of assigning a single flat label to each Gaussian, objects could be classified at multiple levels of abstraction. For example, primitives could first be classified as forest or non-forest elements, followed by more detailed categories such as tree stem, branch, leaf, or ground vegetation. Such a hierarchical structure could enable more localized operations within the simulation, for example, improved ground removal, targeted combustion modelling, or region-specific processing.

In the same direction, classification could move from hard assignment to a soft assignment of classes. So instead of one label, a Gaussian may be assigned to different material classes, hence, a mixed prediction score, since the Gaussians represent not exactly one segment of the scene, but can also be a combination of materials. Therefore, the material classes should also not be restricted to one material.

The system could also be extended with interactive annotation capabilities. Users could manually

label specific objects within the input images, such as buildings or different wood types, and assign custom combustion properties to these elements. These annotations could then be incorporated into the feature distillation process. Even a region in which fire has already spread could be annotated.

Beyond prediction, the framework could also support planning and prevention strategies. For example, the system could be extended to identify optimal locations for fuel breaks or tree cutting lanes designed to slow or redirect wildfire spread.

Finally, the current model focuses primarily on canopy fire behaviour. In reality, wildfire dynamics often involve multiple fire types. Examples are surface fires and underground fires. Underground fires, for instance, burn slowly within organic soil layers and can lead to delayed reignition of surface vegetation. Another direction is the extension of particles to also facilitate water-, or cooling-particles.

References

[1] United States Department of Agriculture. *Firebreak design procedures*. Tech. rep. United States Department of Agriculture, Nov. 2018, pp. 1–7. URL: https://efotg.sc.egov.usda.gov/references/public/IN/Technical_Note_6_Forestry_Firebreaks.pdf.

[2] Frank A Albini. *Estimating wildfire behavior and effects*. Vol. 30. Department of Agriculture, Forest Service, Intermountain Forest and Range …, 1976.

[3] A. Alexandridis et al. "A cellular automata model for forest fire spread prediction: The case of the wildfire that swept through Spetses Island in 1990". In: *Applied Mathematics and Computation* 204.1 (2008), pp. 191–201. ISSN: 0096-3003. DOI: https://doi.org/10.1016/j.amc.2008.06.046. URL: https://www.sciencedirect.com/science/article/pii/S0096300308004943.

[4] Begoña C Arrue, Aníbal Ollero, and JR Matinez De Dios. "An intelligent system for false alarm reduction in infrared forest-fire detection". In: *IEEE Intelligent Systems and their Applications* 15.3 (2000), pp. 64–73.

[5] N. AYACHE. "fast and reliable passive trinocular stereo vision". In: *Proc. of First Int'l Conf. on Computer Vision* 422 (1987). URL: https://cir.nii.ac.jp/crid/1571698601330494592.

[6] Brian N Bailey. "Helios: A scalable 3D plant and environmental biophysical modeling framework". In: *Frontiers in Plant Science* 10 (2019), p. 1185.

[7] A. Bakhshaii and E.A. Johnson. "A review of a new generation of wildfire–atmosphere modeling". In: *Canadian Journal of Forest Research* 49.6 (2019), pp. 565–574. DOI: 10.1139/cjfr-2018-0138. eprint: https://doi.org/10.1139/cjfr-2018-0138. URL: https://doi.org/10.1139/cjfr-2018-0138.

[8] P. A. Beardsley, A. Zisserman, and D. W. Murray. "Navigation using affine structure from motion". In: *Computer Vision — ECCV '94*. Ed. by Jan-Olof Eklundh. Berlin, Heidelberg: Springer Berlin Heidelberg, 1994, pp. 85–96. ISBN: 978-3-540-48400-4.

[9] PA Beardsley, A Zisserman, and DW Murray. *Sequential update of projective and affine structure from motion. Report OUEL 2012/94*. 1994.

[10] Paul Beardsley, Phil Torr, and Andrew Zisserman. "3D model acquisition from extended image sequences". In: *European conference on computer vision*. Springer. 1996, pp. 683–695.

[11] Brian W. Brassard and Han Y. H. Chen. "Stand structural dynamics of North American boreal forests". In: *Critical Reviews in Plant Sciences* 25.2 (Mar. 2006), pp. 115–137. DOI: 10.1080/07352680500348857. URL: https://doi.org/10.1080/07352680500348857.

[12] James K Brown. "Handbook for inventorying downed woody material". In: *Gen. Tech. Rep. INT-16. Ogden, UT: US Department of Agriculture, Forest Service, Intermountain Forest and Range Experiment Station. 24 p.* 16 (1974).

[13] Chris Buehler et al. "Unstructured lumigraph rendering". In: *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*. 2001, pp. 425–432.

[14] Christina Campen and Kalina Velev. *Wildfires and climate Change - NASA Science*. May 2025. URL: https://science.nasa.gov/earth/explore/wildfires-and-climate-change/.

[15] Ruiyu Chen et al. "Combustion characteristics, kinetics and thermodynamics of Pinus Sylvestris pine needle via non-isothermal thermogravimetry coupled with model-free and model-fitting methods". In: *Case Studies in Thermal Engineering* 22 (2020), p. 100756. ISSN: 2214-157X. DOI: https://doi.org/10.1016/j.csite.2020.100756. URL: https://www.sciencedirect.com/science/article/pii/S2214157X20304986.

[16] Keith C Clarke, James A Brass, and Phillip J Riggan. "A cellular automaton model of wildfire propagation and extinction". In: *Photogrammetric Engineering and Remote Sensing. 60 (11): 1355-1367* 60.11 (1994), pp. 1355–1367.

[17] Janice L. Coen, National Science Foundation, and National Center for Atmospheric Research. *Modeling Wildland Fires: A description of the Coupled Atmosphere-Wildland Fire Environment Model (CAWFE)*. Tech. rep. NCAR Earth System Laboratory Mesoscale and Microscale Meteorology Division, Feb. 2013. URL: https://opensky.ucar.edu/system/files/2024-08/technotes_511.pdf.

[18] Northwest Fire Science Consortium. *TYPES OF FIRE*. 2017. URL: https://www.nwfirescience.org/sites/default/files/publications/Types%20of%20Fire.pdf.

[19] Miguel G. Cruz and Martin E. Alexander. "The 10 percent wind speed rule of thumb for estimating a wildfire's forward rate of spread in forests and shrublands". In: *Annals of Forest Science* 76.2 (Apr. 2019). DOI: 10.1007/s13595-019-0829-8. URL: https://doi.org/10.1007/s13595-019-0829-8.

[20] François Darmon et al. "Robust Gaussian Splatting". In: *CoRR* abs/2404.04211 (2024). DOI: 10.48550/ARXIV.2404.04211. arXiv: 2404.04211. URL: https://doi.org/10.48550/arXiv.2404.04211.

[21] John L. De Ris, Peter K. Wu, and G. Heskestad. "Radiation fire modeling". In: *Proceedings of the Combustion Institute* 28.2 (2000), pp. 2751–2759. ISSN: 1540-7489. DOI: https://doi.org/10.1016/S0082-0784(00)80696-7. URL: https://www.sciencedirect.com/science/article/pii/S0082078400806967.

[22] Frank C. Dennis and Knowledge to Go Places. *Fuelbreak Guidelines for Forested Subdivisions communities*. Tech. rep. Colorado State University, 2005. URL: https://static.colostate.edu/client-files/csfs/pdfs/fuelbreak_guidellines.pdf.

[23] PlayCanvas Developer. *SuperSplat Editor*. URL: https://superspl.at/editor.

[24] *Drone data collection workflow | Open Forest Observatory*. 2024. URL: https://openforestobservatory.org/workflows/data-collection/drone/.

[25] A Hernández Encinas et al. "Simulation of forest fire fronts using cellular automata". In: *Advances in Engineering Software* 38.6 (2007), pp. 372–378.

[26] Blender Foundation. *A.N.T.Landscape*. 2024. URL: https://extensions.blender.org/add-ons/antlandscape/.

[27] Blender Foundation. *Blender*. Mar. 2024. URL: https://www.blender.org/.

[28] Lelis Gonzaga Fraga et al. "Influence of Operating Conditions on the Thermal Behavior and Kinetics of Pine Wood Particles Using Thermogravimetric Analysis". In: *Energies* 13.11 (2020). ISSN: 1996-1073. DOI: 10.3390/en13112756. URL: https://www.mdpi.com/1996-1073/13/11/2756.

[29] Joana Gouveia Freire and Carlos Castro DaCamara. "Using cellular automata to simulate wildfire propagation and to assist in fire management". In: *Natural hazards and earth system sciences* 19.1 (Jan. 2019), pp. 169–179. DOI: 10.5194/nhess-19-169-2019. URL: https://nhess.copernicus.org/articles/19/169/2019/nhess-19-169-2019.html.

[30] Priya Garg et al. "Limiting conditions of smoldering-to-flaming transition of cellulose powder". In: *Fire Safety Journal* 141 (2023), p. 103936. ISSN: 0379-7112. DOI: https://doi.org/10.1016/j.firesaf.2023.103936. URL: https://www.sciencedirect.com/science/article/pii/S0379711223002047.

[31] Daotong Geng et al. "Modification of the Rothermel model parameters – the rate of surface fire spread of Pinus koraiensis needles under no-wind and various slope conditions". In: *International Journal of Wildland Fire* 33.4 (Apr. 2024), WF23118. ISSN: 1049-8001. DOI: 10.1071/WF23118. eprint: https://connectsci.au/wf/article-pdf/doi/10.1071/WF23118/154055/wf23118.pdf. URL: https://doi.org/10.1071/WF23118.

[32] Rohit Ghosh, Jishnu Adhikary, and Rezki Chemlal. "Fire Spread Modeling Using Probabilistic Cellular Automata". In: *Cellular Automata Technology - Third Asian Symposium, ASCAT 2024, Durgapur, India, February 29 - March 2, 2024, Revised Selected Papers*. Ed. by Mamata Dalui, Sukanta Das, and Enrico Formenti. Vol. 2021. Communications in Computer and Information Science. Springer, 2024, pp. 45–55. DOI: 10.1007/978-3-031-56943-2_4. URL: https://doi.org/10.1007/978-3-031-56943-2%5C_4.

[33] Christophe Godin. “Representing and encoding plant architecture: a review”. In: *Annals of forest science* 57.5 (2000), pp. 413–438.

[34] Steven J. Gortler et al. “The Lumigraph”. In: *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1996, New Orleans, LA, USA, August 4-9, 1996*. Ed. by John Fujii. ACM, 1996, pp. 43–54. DOI: 10.1145/237170.237200. URL: https://doi.org/10.1145/237170.237200.

[35] Torsten Hädrich et al. “Fire in paradise: mesoscale simulation of wildfires”. In: *ACM Trans. Graph.* 40.4 (2021), 163:1–163:15. DOI: 10.1145/3450626.3459954. URL: https://doi.org/10.1145/3450626.3459954.

[36] Dennis Haitz et al. “The Potential of Neural Radiance Fields and 3D Gaussian Splatting for 3D Reconstruction from Aerial Imagery”. In: *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences* 10 (2024), pp. 97–104.

[37] Dennis Haitz et al. “The Potential of Neural Radiance Fields and 3D Gaussian Splatting for 3D Reconstruction from Aerial Imagery”. In: *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences* X-2-2024 (June 2024), pp. 97–104. DOI: 10.5194/isprs-annals-X-2-2024-97-2024.

[38] Yujin Ham, Mateusz Michalkiewicz, and Guha Balakrishnan. “DRAGON: Drone and Ground Gaussian Splatting for 3D Building Reconstruction”. In: *2024 IEEE International Conference on Computational Photography (ICCP)*. 2024, pp. 1–12. DOI: 10.1109/ICCP61108.2024.10644903.

[39] C.G. Harris and J.M. Pike. “3D positional integration from image sequences”. In: *Image and Vision Computing* 6.2 (1988). 3rd Alvey Vision Meeting, pp. 87–90. ISSN: 0262-8856. DOI: https://doi.org/10.1016/0262-8856(88)90003-0. URL: https://www.sciencedirect.com/science/article/pii/0262885688900030.

[40] Richard I Hartley. “Euclidean reconstruction from uncalibrated views”. In: *Joint European-US workshop on applications of invariance in computer vision*. Springer. 1993, pp. 235–256.

[41] Charles Hernandez et al. “Statistical modelling of wildfire size and intensity: a step toward meteorological forecasting of summer extreme fire risk”. In: *Annales Geophysicae*. Vol. 33. 12. Copernicus GmbH Göttingen, Germany. 2015, pp. 1495–1506.

[42] Christopher Horvath and Willi Geiger. “Directable, high-resolution simulation of fire on the GPU”. In: *ACM Transactions on Graphics* 28.3 (July 2009), pp. 1–8. DOI: 10.1145/1531326.1531347. URL: https://doi.org/10.1145/1531326.1531347.

[43] Elaine T Howard. “Heat of combustion of various southern pine materials”. In: *Wood Science 5 (3): 194-197* (1973).

[44] Oliver C. Ibe. *Markov processes for stochastic modeling*. Elsevier Inc., Jan. 2013. Chap. 9. DOI: 10.1016/c2012-0-06106-6. URL: https://doi.org/10.1016/c2012-0-06106-6.

[45] R. Kallada Janardhan and S. Hostikka. “Predictive computational fluid dynamics simulation of fire spread on wood cribs”. In: *Fire Technology* 55.6 (Apr. 2019), pp. 2245–2268. DOI: 10.1007/s10694-019-00855-3. URL: https://doi.org/10.1007/s10694-019-00855-3.

[46] Matthew W. Jones et al. “Global rise in forest fire emissions linked to climate change in the extratropics”. In: *Science* 386.6719 (Oct. 2024). DOI: 10.1126/science.adl5889. URL: https://doi.org/10.1126/science.adl5889.

[47] Ioannis Karafyllidis and Adonios Thanailakis. “A model for predicting forest fire spreading using cellular automata”. In: *Ecological Modelling* 99.1 (1997), pp. 87–97.

[48] Bernhard Kerbl et al. “3D Gaussian Splatting for Real-Time Radiance Field Rendering”. In: *ACM Trans. Graph.* 42.4 (2023), 139:1–139:14. DOI: 10.1145/3592433. URL: https://doi.org/10.1145/3592433.

[49] Mohammed M. Khan, Archibald Tewarson, and Marcos Chaos. “Combustion Characteristics of Materials and Generation of Fire Products”. In: *SFPE Handbook of Fire Protection Engineering*. Ed. by Morgan J. Hurley et al. New York, NY: Springer New York, 2016, pp. 1143–1232. ISBN: 978-1-4939-2565-0. DOI: 10.1007/978-1-4939-2565-0_36. URL: https://doi.org/10.1007/978-1-4939-2565-0_36.

[50] Alexander Kirillov et al. “Segment Anything”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. Oct. 2023, pp. 4015–4026.

[51] KNMI. *KNMI - Windschaal van Beaufort*. URL: https://www.knmi.nl/kennis-en-datacentrum/uitleg/windschaal-van-beaufort.

[52] Nobuo Kochi et al. “All-around 3D plant modeling system using multiple images and its composition”. In: *Breeding Science* 72.1 (2022), pp. 75–84. DOI: 10.1270/jsbbs.21068.

[53] Andrzej Kokosza et al. “Scintilla: Simulating Combustible Vegetation for Wildfires”. In: *ACM Trans. Graph.* 43.4 (2024), 70:1–70:21. DOI: 10.1145/3658192. URL: https://doi.org/10.1145/3658192.

[54] Kyle and Kyle. *The temperature of fire*. Nov. 2024. URL: https://www.cityfire.co.uk/news/the-temperature-of-fire/.

[55] Stéphane Laveau and Olivier D. Faugeras. “3-D scene representation as a collection of images”. In: *12th IAPR International Conference on Pattern Recognition, Conference A: Computer Vision & Image Processing, ICPR 1994, Jerusalem, Israel, 9-13 October, 1994, Volume 1*. IEEE, 1994, pp. 689–691. DOI: 10.1109/ICPR.1994.576404. URL: https://doi.org/10.1109/ICPR.1994.576404.

[56] Rik Leemans and I. Colin Prentice. “Description and simulation of tree-layer composition and size distributions in a primaeval Picea-Pinus forest”. In: *Plant Ecology* 69.1-3 (Apr. 1987), pp. 147–156. DOI: 10.1007/bf00038696. URL: https://doi.org/10.1007/bf00038696.

[57] Jiaqi Li et al. “A Method for the 3D Reconstruction of Landscape Trees in the Leafless Stage”. In: *Remote Sensing* 17.8 (2025). ISSN: 2072-4292. DOI: 10.3390/rs17081473. URL: https://www.mdpi.com/2072-4292/17/8/1473.

[58] Jie Liang et al. “3D Plant Modelling via Hyperspectral Imaging”. In: *2013 IEEE International Conference on Computer Vision Workshops, ICCV Workshops 2013, Sydney, Australia, December 1-8, 2013*. IEEE Computer Society, 2013, pp. 172–177. DOI: 10.1109/ICCVW.2013.29. URL: https://doi.org/10.1109/ICCVW.2013.29.

[59] Daoming Liu et al. “FLAMEFORGE: Combustion Simulation of Wooden Structures”. In: *Proc. ACM Comput. Graph. Interact. Tech.* 8.4 (Aug. 2025). DOI: 10.1145/3747855. URL: https://doi.org/10.1145/3747855.

[60] Zachary Lowry. *How much does a tree weigh? Use this chart to find out*. Feb. 2026. URL: https://thetimberlandinvestor.com/how-much-does-a-tree-weigh/#:~:text=Most%20%E2%80%9Caverage%E2%80%9D%20sized%20full%2D,to%20give%20an%20exact%20answer..

[61] Milosz Makowski et al. “Synthetic silviculture: multi-scale modeling of plant ecosystems”. In: *ACM Trans. Graph.* 38.4 (2019), 131:1–131:14. DOI: 10.1145/3306346.3323039. URL: https://doi.org/10.1145/3306346.3323039.

[62] Z. Melek and J. Keyser. “Interactive simulation of fire”. In: *10th Pacific Conference on Computer Graphics and Applications, 2002. Proceedings.* 2002, pp. 431–432. DOI: 10.1109/PCCGA.2002.1167889.

[63] Hector Mendoza, Alexander Brown, and Allen Joseph Ricks. “Modeling High Heat Flux Combustion of Coniferous Trees Using Chemically Reacting Lagrangian Particles.” In: *Modeling High Heat Flux Combustion of Coniferous Trees Using Chemically Reacting Lagrangian Particles.* Sandia National Lab. (SNL-NM), Albuquerque, NM (United States). Oct. 2019. URL: https://www.osti.gov/biblio/1642826.

[64] Lars M. Mescheder et al. “Occupancy Networks: Learning 3D Reconstruction in Function Space”. In: *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*. Computer Vision Foundation / IEEE, 2019, pp. 4460–4470. DOI: 10.1109/CVPR.2019.00459. URL: http://openaccess.thecvf.com/content%5CCVPR%5C2019/html/Mescheder%5COccupancy%5CNetworks%5CLearning%5C3D%5CReconstruction%5Cin%5CFunction%5CSpace%5CCVPR%5C2019%5Cpaper.html.

[65] Alexandre Meyer and Fabrice Neyret. “Multiscale Shaders for the Efficient Realistic Rendering of Pine-Trees”. In: *Proceedings of the Graphics Interface 2000 Conference, May 15-17, 2000, Montréal, Québec, Canada*. Ed. by Sidney S. Fels and Pierre Poulin. Canadian Human-Computer Communications Society, 2000, pp. 137–144. URL: http://www.graphicsinterface.org/proceedings/2000/184/.

[66] Gerard Michon. *Surface area of an ellipsoid - scalene ellipsoid - numericana*. 2005. URL: http://nbarth.net/notes/src/notes-calc-raw/others/X-numericana/ellipsoid.htm.

[67] Ben Mildenhall et al. “NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis”. In: *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part I*. Ed. by Andrea Vedaldi et al. Vol. 12346. Lecture Notes in Computer Science. Springer, 2020, pp. 405–421. DOI: 10.1007/978-3-030-58452-8_24. URL: https://doi.org/10.1007/978-3-030-58452-8%5C_24.

[68] Christos Mitsanis, William Hurst, and Bedir Tekinerdogan. “A 3D functional plant modelling framework for agricultural digital twins”. In: *Comput. Electron. Agric.* 218 (2024), p. 108733. DOI: 10.1016/J.COMPAG.2024.108733. URL: https://doi.org/10.1016/j.compag.2024.108733.

[69] R. Mohr, L. Quan, and F. Veillon. “Relative 3D Reconstruction Using Multiple Uncalibrated Images”. In: *The International Journal of Robotics Research* 14.6 (1995), pp. 619–632. DOI: 10.1177/027836499501400607. eprint: https://doi.org/10.1177/027836499501400607. URL: https://doi.org/10.1177/027836499501400607.

[70] Roger Mohr, Long Quan, and Françoise Veillon. “Relative 3D reconstruction using multiple uncalibrated images”. In: *The International Journal of Robotics Research* 14.6 (1995), pp. 619–632.

[71] Martien Molenaar and Tao Cheng. “Fuzzy spatial objects and their dynamics”. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 55.3 (2000), pp. 164–175. ISSN: 0924-2716. DOI: https://doi.org/10.1016/S0924-2716(00)00017-4. URL: https://www.sciencedirect.com/science/article/pii/S0924271600000174.

[72] Arnadi Murtiyoso et al. “Virtual forests: a review on emerging questions in the use and application of 3D data in forestry”. In: *International Journal of Forest Engineering* 35.1 (2024), pp. 29–42.

[73] Michael B. Nielsen et al. “Physics-Based combustion Simulation”. In: *ACM Transactions on Graphics* 41.5 (Mar. 2022), pp. 1–21. DOI: 10.1145/3526213. URL: https://doi.org/10.1145/3526213.

[74] Michael Bang Nielsen et al. “Physics-Based Combustion Simulation”. In: *ACM Trans. Graph.* 41.5 (2022), 176:1–176:21. DOI: 10.1145/3526213. URL: https://doi.org/10.1145/3526213.

[75] Open Forest Observatory. *Mission 001210 | Open Forest Observatory*. Aug. 2025. URL: https://openforestobservatory.org/data/drone/mission-details/001210/.

[76] openforestobservatory. *Forest Data | Open Forest Observatory*. 2025. URL: https://openforestobservatory.org/data/.

[77] Maxime Oquab et al. *DINOv2: Learning Robust Visual Features without Supervision*. 2024. arXiv: 2304.07193 [cs.CV]. URL: https://arxiv.org/abs/2304.07193.

[78] Wojtek Palubicki et al. “Ecoclimates: climate-response modeling of vegetation”. In: *ACM Trans. Graph.* 41.4 (2022), 155:1–155:19. DOI: 10.1145/3528223.3530146. URL: https://doi.org/10.1145/3528223.3530146.

[79] Jeong Joon Park et al. “DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation”. In: *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*. Computer Vision Foundation / IEEE, 2019, pp. 165–174. DOI: 10.1109/CVPR.2019.00025. URL: http://openaccess.thecvf.com/content%5C_CVPR%5C_2019/html/Park%5C_DeepSDF%5C_Learning%5C_Continuous%5C_Signed%5C_Distance%5C_Functions%5C_for%5C_Shape%5C_Representation%5C_CVPR%5C_2019%5C_paper.html.

[80] Sören Pirk et al. “Interactive wood combustion for botanical tree models”. In: *ACM Trans. Graph.* 36.6 (2017), 197:1–197:12. DOI: 10.1145/3130800.3130814. URL: https://doi.org/10.1145/3130800.3130814.

[81] Stefano Puliti et al. “Benchmarking tree species classification from proximally-sensed laser scanning data: introducing the FOR-species20K dataset”. In: *CoRR* abs/2408.06507 (2024). DOI: 10.48550/ARXIV.2408.06507. arXiv: 2408.06507. URL: https://doi.org/10.48550/arXiv.2408.06507.

[82] Stefano Puliti et al. “FOR-instance: a UAV laser scanning benchmark dataset for semantic and instance segmentation of individual trees”. In: *CoRR* abs/2309.01279 (2023). DOI: 10.48550/ARXIV.2309.01279. arXiv: 2309.01279. URL: https://doi.org/10.48550/arXiv.2309.01279.

[83] Puuinfo. *Fire properties of wood - Puuinfo*. Nov. 2020. URL: https://puuinfo.fi/puutieto/wood-as-a-material/fire-properties-of-wood/?lang=en#:~:text=When%20the%20temperature%20of%20wood,individual%20piece%20of%20wood%20decreases..

[84] Yiqian Qiao et al. “Transition from smouldering to flaming combustion of pine needle fuel beds under natural convection”. In: *Proceedings of the Combustion Institute* 40.1 (2024), p. 105343. ISSN: 1540-7489. DOI: https://doi.org/10.1016/j.proci.2024.105343. URL: https://www.sciencedirect.com/science/article/pii/S1540748924001536.

[85] Hanqing Qiu et al. “Forest digital twin: A new tool for forest management practices based on Spatio-Temporal Data, 3D simulation Engine, and intelligent interactive environment”. In: *Computers and Electronics in Agriculture* 215 (2023), p. 108416.

[86] Ri-Zhao Qiu et al. “Language-Driven Physics-Based Scene Synthesis and Editing via Feature Splatting”. In: *Computer Vision - ECCV 2024 - 18th European Conference, Milan, Italy, September 29-October 4, 2024, Proceedings, Part XLI*. Ed. by Ales Leonardis et al. Vol. 15099. Lecture Notes in Computer Science. Springer, 2024, pp. 368–383. DOI: 10.1007/978-3-031-72940-9_21. URL: https://doi.org/10.1007/978-3-031-72940-9%5C_21.

[87] Alec Radford et al. “Learning Transferable Visual Models From Natural Language Supervision”. In: *Proceedings of the 38th International Conference on Machine Learning*. Ed. by Marina Meila and Tong Zhang. Vol. 139. Proceedings of Machine Learning Research. PMLR, 18–24 Jul 2021, pp. 8748–8763. URL: https://proceedings.mlr.press/v139/radford21a.html.

[88] William T. Reeves. “Particle systems—a technique for modeling a class of fuzzy objects”. In: *Proceedings of the 10th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH ’83. Detroit, Michigan, USA: Association for Computing Machinery, 1983, pp. 359–375. ISBN: 0897911091. DOI: 10.1145/800059.801167. URL: https://doi.org/10.1145/800059.801167.

[89] Richard C Rothermel. *A mathematical model for predicting fire spread in wildland fuels*. Vol. 115. Intermountain Forest & Range Experiment Station, Forest Service, US …, 1972.

[90] Peter Rousseeuw and Annick Leroy. *Robust regression and outlier detection*. John wiley & sons, 1987. URL: https://books.google.nl/books?hl=nl&lr=&id=cZEzEQAAQBAJ&oi=fnd&pg=PA1&dq=Robust+Regression+and+Outlier+Detection&ots=tcUeUmVBMr&sig=V1urqAj5g1HjgafZw57jrL7QSmo&redir_esc=y#v=onepage&q=Robust%20Regression%20and%20Outlier%20Detection&f=false.

[91] Muhammad A Santoso et al. “Review of the transition from smouldering to flaming combustion in wildfires”. In: *Frontiers in Mechanical Engineering* 5 (2019), p. 49.

[92] Muhammad A Santoso et al. “The effects of pulsating wind on the transition from smouldering to flaming combustion”. In: *Fire Safety Journal* 141 (2023), p. 103993. ISSN: 0379-7112. DOI: https://doi.org/10.1016/j.firesaf.2023.103993. URL: https://www.sciencedirect.com/science/article/pii/S0379711223002618.

[93] Younes Oulad Sayad, Hajar Mousannif, and Hassan Al Moatassime. “Predictive modeling of wildfires: A new dataset and machine learning approach”. In: *Fire Safety Journal* 104 (2019), pp. 130–146. ISSN: 0379-7112. DOI: https://doi.org/10.1016/j.firesaf.2019.01.006. URL: https://www.sciencedirect.com/science/article/pii/S0379711218303941.

[94] Johannes L. Schonberger and Jan-Michael Frahm. “Structure-From-Motion Revisited”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2016.

[95] Johannes Lutz Schönberger and Jan-Michael Frahm. “Structure-from-Motion Revisited”. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.

[96] Johannes Lutz Schönberger et al. “Pixelwise View Selection for Unstructured Multi-View Stereo”. In: *European Conference on Computer Vision (ECCV)*. 2016.

[97] Belal Shaheen et al. “ForestSplat: Proof-of-Concept for a Scalable and High-Fidelity Forestry Mapping Tool Using 3D Gaussian Splatting”. In: *Remote Sensing* 17.6 (2025). ISSN: 2072-4292. DOI: 10.3390/rs17060993. URL: https://www.mdpi.com/2072-4292/17/6/993.

[98] Sanjeev Sharma and Puskar Khanal. “Forest Fire Prediction: A Spatial Machine Learning and Neural Network Approach”. In: *Fire* 7.6 (2024). ISSN: 2571-6255. DOI: 10.3390/fire7060205. URL: https://www.mdpi.com/2571-6255/7/6/205.

[99] Long Shi and Michael Chew. “A review of thermal properties of timber and char at elevated temperatures”. In: *Indoor and Built Environment* 32 (Aug. 2021), p. 1420326X2110355. DOI: 10.1177/1420326X211035557.

[100] Chiranjibi Sitaula et al. “Recent advances in scene image representation and classification”. In: *Multim. Tools Appl.* 83.3 (2024), pp. 9251–9278. DOI: 10.1007/S11042-023-15005-9. URL: https://doi.org/10.1007/s11042-023-15005-9.

[101] E. J. Strobach et al. “Isolating and investigating updrafts induced by wildland fires using an airborne doppler Lidar during FIREX□AQ”. In: *Journal of Geophysical Research Atmospheres* 128.14 (July 2023). DOI: 10.1029/2023jd038809. URL: https://doi.org/10.1029/2023jd038809.

[102] Matthew Tancik et al. “Nerfstudio: A Modular Framework for Neural Radiance Field Development”. In: *ACM SIGGRAPH 2023 Conference Proceedings*. SIGGRAPH ’23. 2023.

[103] Jiadong Tang et al. “DroneSplat: 3D Gaussian Splatting for Robust 3D Reconstruction from In-the-Wild Drone Imagery”. In: *CoRR* abs/2503.16964 (2025). DOI: 10.48550/ARXIV.2503.16964. arXiv: 2503.16964. URL: https://doi.org/10.48550/arXiv.2503.16964.

[104] Steve W Taylor et al. “Wildfire prediction to inform fire management: statistical science challenges”. In: (2013).

[105] Philip Hilaire Sean Torr. “Motion segmentation and outlier detection”. PhD thesis. University of Oxford UK, 1995.

[106] Jan Trochta et al. “3D Forest: An application for descriptions of three-dimensional forest structures using terrestrial LiDAR”. In: *PloS one* 12.5 (2017), e0176871.

[107] Andrea Trucchia et al. “PROPAGATOR: An Operational Cellular-Automata Based Wildfire Simulator”. In: *Fire* 3.3 (2020). ISSN: 2571-6255. DOI: 10.3390/fire3030026. URL: https://www.mdpi.com/2571-6255/3/3/26.

[108] Giuseppe A. Trunfio et al. “A New Algorithm for Simulating Wildfire Spread through Cellular Automata”. In: *ACM Trans. Model. Comput. Simul.* 22.1 (2011), 6:1–6:26. DOI: 10.1145/2043635.2043641. URL: https://doi.org/10.1145/2043635.2043641.

[109] Shimon Ullman. “The interpretation of structure from motion”. In: *Proceedings of the Royal Society of London. Series B. Biological Sciences* 203.1153 (1979), pp. 405–426.

[110] Juan Pablo Valdivieso and Juan de Dios Rivera. “Effect of wind on smoldering combustion limits of moist pine needle beds”. In: *Fire Technology* 50.6 (2014), pp. 1589–1605.

[111] CE Van Wagner. “Forest Fire Research–Hindsight and Foresight[1]”. In: *General Technical Report PSW.* 101-109 (1978), p. 115.

[112] Marcos Vanella et al. “A Multi-Fidelity Framework for Wildland Fire Behavior Simulations over Complex Terrain”. In: *Atmosphere* 12.2 (Feb. 2021), p. 273. DOI: 10.3390/atmos12020273. URL: https://www.mdpi.com/2073-4433/12/2/273.

[113] Shuzhe Wang et al. *DUSt3R: Geometric 3D Vision Made Easy*. 2023. arXiv: 2312.14132 [cs.CV].

[114] Shuzhe Wang et al. “Dust3r: Geometric 3d vision made easy”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2024, pp. 20697–20709.

[115] Tong Wu et al. “Recent advances in 3D Gaussian splatting”. In: *Comput. Vis. Media* 10.4 (2024), pp. 613–642. DOI: 10.1007/S41095-024-0436-Y. URL: https://doi.org/10.1007/s41095-024-0436-y.

[116] Zhaohui Wu, Zhong Zhou, and Wei Wu. “Realistic Fire Simulation: A Survey”. In: *International Conference on Computer-Aided Design and Computer Graphics* 8 (Sept. 2011), pp. 333–340. DOI: 10.1109/cad/graphics.2011.26. URL: https://doi.org/10.1109/cad/graphics.2011.26.

[117] Yiqing Xu et al. “Modeling Forest Fire Spread Using Machine Learning-Based Cellular Automata in a GIS Environment”. In: *Forests* 13.12 (2022). ISSN: 1999-4907. DOI: 10.3390/f13121974. URL: https://www.mdpi.com/1999-4907/13/12/1974.

[118] Vickie Ye et al. “gsplat: An open-source library for Gaussian splatting”. In: *Journal of Machine Learning Research* 26.34 (2025), pp. 1–17.

[119] Jiawei You et al. “Real-time 3D visualization of forest fire spread based on tree morphology and finite state machine”. In: *Computers Graphics* 103 (2022), pp. 109–120. ISSN: 0097-8493. DOI: https://doi.org/10.1016/j.cag.2022.01.009. URL: https://www.sciencedirect.com/science/article/pii/S0097849322000097.

[120] Anthony Chun Yin Yuen et al. “Fire scene investigation of an arson fire incident using computational fluid dynamics based fire simulation”. In: *Building Simulation* 7.5 (Jan. 2014), pp. 477–487. DOI: 10.1007/s12273-014-0164-9. URL: https://doi.org/10.1007/s12273-014-0164-9.

[121] Xiaopeng Zhang and Frédéric Blaise. “Progressive Polygon Foliage Simplification”. In: *Tsinghua University Press* (2007). URL: https://inria.hal.science/inria-00107686/file/A03-R-231.pdf.

[122] Zhengyou Zhang and Olivier Faugeras. “A Comparative Study of 3D Motion Estimation”. In: *3D Dynamic Scene Analysis: A Stereo Based Approach*. Springer, 1992, pp. 55–80.

[123] Zhong Zheng et al. “Forest fire spread simulating model using cellular automaton with extreme learning machine”. In: *Ecological Modelling* 348 (2017), pp. 33–43.

[124] Xiaojing Zhou et al. “A model for physics-based fire simulation and analysis”. In: *Virtual Reality* 25.2 (Aug. 2021), pp. 421–432. DOI: 10.1007/s10055-020-00465-3. URL: https://doi.org/10.1007/s10055-020-00465-3.