remote sensing

MDPI

Article

Integration of 3D Gaussian Splatting and Neural Radiance Fields in Virtual Reality Fire Fighting

Haojie Lian [1], Kangle Liu [1], Ruochen Cao [2,*], Ziheng Fei [3], Xin Wen [2] and Leilei Chen [4]

[1] Key Laboratory of In-Situ Property-Improving Mining of Ministry of Education, Taiyuan University of Technology, Taiyuan 030024, China; lianhaojie@tyut.edu.cn (H.L.); liukangle1238@link.tyut.edu.cn (K.L.)
[2] School of Software, Taiyuan University of Technology, Jinzhong 030600, China; xwen@tyut.edu.cn
[3] Department of Architectural Engineering, Hanyang University, Seoul 04763, Republic of Korea; fei000623@hanyang.ac.kr
[4] Henan International Joint Laboratory of Structural Mechanics and Computational Simulation, College of Architectural and Civil Engineering, Huanghuai University, Zhumadian 463000, China; chenllei@mail.ustc.edu.cn
* Correspondence: caoruochen@tyut.edu.cn

Abstract: Neural radiance fields (NeRFs) and 3D Gaussian splatting have emerged as promising 3D reconstruction techniques recently. However, their application in virtual reality (VR), particularly in firefighting training, remains underexplored. We present an innovative VR firefighting simulation system based on 3D Gaussian Splatting technology. Leveraging these techniques, we successfully reconstruct realistic physical environments. By integrating the Unity3D game engine with head-mounted displays (HMDs), we created and presented immersive virtual fire scenes. Our system incorporates NeRF technology to generate highly realistic models of firefighting equipment. Users can freely navigate and interact with fire within the virtual fire scenarios, enhancing immersion and engagement. Moreover, by utilizing the Photon PUN2 networking framework, our system enables multi-user collaboration on firefighting tasks, improving training effectiveness and fostering teamwork and communication skills. Through experiments and surveys, it is demonstrated that the proposed VR framework enhances user experience and holds promises for improving the effectiveness of firefighting training.

Keywords: neural radiance fields; 3D Gaussian splatting; fire; immersive virtual reality

Citation: Lian, H.; Liu, K.; Cao, R.; Fei, Z.; Wen, X.; Chen, L. Integration of 3D Gaussian Splatting and Neural Radiance Fields in Virtual Reality Fire Fighting. *Remote Sens.* **2024**, *16*, 2448. https://doi.org/10.3390/rs16132448

Academic Editors: Mila Koeva, Christine Pohl and Dessislava Petrova-Antonova

Received: 19 May 2024
Revised: 24 June 2024
Accepted: 1 July 2024
Published: 3 July 2024

1. Introduction

Frequent occurrence of fires poses significant threats to both human lives and property. The causes of fires frequently stem from factors such as a lack of sufficient safety awareness, negligent fire prevention practices, and inadequate familiarity with the proper use of fire extinguishing equipment [1]. Therefore, it is imperative to prioritize enhancing public awareness and implementing comprehensive fire safety training programs. Although fire drills are a prevalent and efficacious approach, they can impose significant demands on resources, necessitating substantial allocations of manpower, materials, and financial assets [2]. Moreover, the creation of realistic fire scenarios for drills can potentially engender safety hazards, while logistical support complexities often impede broader accessibility to such training programs for diverse populations. Consequently, conventional fire drills may inadequately address comprehensive societal needs. To meet these challenges, there is an increasing need for pioneering fire safety education methods that effectively guide individuals to overcome the limitations of traditional drills.

Utilizing virtual reality (VR) technology allows for the recreation of necessary scenarios in a virtual environment, providing users with interactive platforms to engage with characters or objects via interactive devices, facilitating task completion. In industrial production, VR technology is progressively employed for workforce training, process simulation, and quality assurance [3]. However, a significant challenge in these applications

is the inclination for virtual scenarios developed using modeling tools to possess visual characteristics that align more closely with video game environments rather than accurately representing real-world scenarios [4].

To enhance the fidelity of the virtual fire protection environment, it is imperative to employ advanced three-dimensional modeling techniques. These technologies are essential for the precise visualization and representation of the geometric and material attributes of fire extinguishing equipment, ensuring an accurate replication of the textures and details inherent in real-world objects. While mesh-based 3D modeling techniques are effective in representing simple objects, mesh modeling can become very complex and unmanageable for objects with complex geometries and details [5]. The demanding level of modeling precision required to accurately capture these intricate features presents a substantial challenge for non-expert users. When working with 3D modeling tools, non-expert users often face challenges in accurately depicting complex features, which ultimately impacts the user experience and reduces immersion. As a result, the inherent limitations of existing 3D modeling tools limit the utility and effectiveness of virtual environments in industrial safety applications, particularly for individual users or non-professionals who require increased precision and detail. This situation highlights the necessity for more streamlined and user-friendly modeling techniques.

In addressing these challenges, we have developed an innovative fire drill system. This system is designed to facilitate efficient fire drill exercises while optimizing resource utilization, thereby reducing the probability of fires caused by human error. Notably, the system features robust immersion capabilities and a wide array of functionalities, offering transformative potential for the field of firefighting.

This study tests the following hypotheses:

1. Hypothesis 1: The use of 3D Gaussian splatting [6] in VR fire training will provide a more immersive and realistic fire scene compared to traditional 3D modeling techniques.
2. Hypothesis 2: Models of fire extinguishing equipment created using Neural Radiance Fields (NeRFs) [7,8] technology will offer more realistic textures and details, enhancing user interaction and learning.
3. Hypothesis 3: Integrating multi-user virtual fire scenarios using the Photon PUN2 [9] framework will improve collaboration and effectiveness in fire suppression tasks compared to single-user VR scenarios.

Preliminary results from our system testing indicate that the virtual reality system designed and developed in this study meets the expected design requirements in terms of realism, operability, and immersion. In addition, multi-user functionality was found to enhance collaboration skills, which are critical for effective fire response. In conclusion, our research contributions are as follows:

- Immersive fire scene: import 3D Gaussian Splatting scenes into Unity3D, and users can wear VR head-mounted displays (HMDs) to roam around the scene and realize interaction with the scene.
- Differences from traditional modeling: Unlike the previous use of 3ds Max [10] and other modeling tools to create models of fire extinguishing equipment, neural radiance fields (NeRFs) technology is used to create models of fire extinguishing equipment with more realistic texture.
- Full System Integration: Design virtual fire scenarios with a multi-user experience using the Photon PUN2 framework, allowing users to collaborate on fire suppression tasks.

2. Materials and Methods

As highlighted above, we designed our immersive system with a purpose of providing a more intuitive way to support fire drill practices. 3D reconstruction is fundamental. In this section, we will first introduce two state-of-the-art 3D reconstruction methods that were applied in our system. Following this, we will then present the design and implementation of our system.

2.1. Neural Radiance Fields

Neural radiance fields (NeRFs) uses a fully connected neural network to learn a radiance field that can render a synthetic view of a 3D scene (Figure 1).

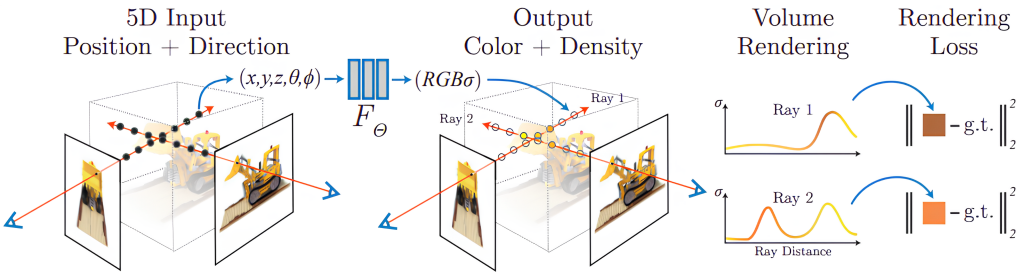


Figure 1. The process of Neural Radiance Fields (NeRFs) [7].

The NeRF maps spatial positions and viewing direction expressed in 5D coordinates (x, y, z, θ, ϕ) to a volume density σ and view-related colors (r, g, b) (see the process in Figure 1). Representation of the three-dimensional world with a volumetric scene function F_Θ with learnable parameters:

$$F_\Theta : (x, y, z, \theta, \phi) \mapsto (r, g, b, \sigma). \tag{1}$$

In order to render a new 2D projection, NeRF adopts the traditional volume rendering method [11]. A ray can be represented as:

$$\mathbf{r}(t) = o + td, \tag{2}$$

and the proximal and distal boundaries of t are t_n and t_f respectively, by marking the origin of a ray as o and the direction of the ray as d. The final color C($\mathbf{r}$) of this ray projection can be expressed as an integral:

$$C(\mathbf{r}) = \int_{t_n}^{t_f} T(t) \cdot \sigma(\mathbf{r}(t)) \cdot c(\mathbf{r}(t), d) dt. \tag{3}$$

The function $T(t)$ represents the cumulative transparency of the ray from t_n to t, that is, the probability that the ray has not been stopped by hitting any particles from t_n to t, specifically written:

$$T(t) = exp(-\int_{t_n}^{t} \sigma(\mathbf{r}(s)) ds). \tag{4}$$

Because the neural network is difficult to learn high-frequency information, the spatial positions and viewing direction are directly used as the input of the network, and the resolution of the rendering effect is low [12]. Using the position information coding method to map the input to high-frequency can effectively solve this problem. γ is used to map the input into a high-dimensional space:

$$\gamma(p) = \left(\sin\left(2^0 \pi p\right), \cos\left(2^0 \pi p\right), \cdots, \sin\left(2^{L-1} \pi p\right), \cos\left(2^{L-1} \pi p\right)\right). \tag{5}$$

The NeRF process is end-to-end microscopic, so these network models can be trained using gradient descent methods. The training loss is directly defined as:

$$\mathcal{L} = \sum_{\mathbf{r} \in \mathbf{R}} \| \hat{\mathbf{C}}(\mathbf{r}) - \mathbf{C}(\mathbf{r}) \|_2^2 \tag{6}$$

where $\mathbf{R}$ is the set of rays in each batch, and $\hat{\mathrm{C}}(\mathbf{r})$, C($\mathbf{r}$) are the predicted color and ground truth for ray $\mathbf{r}$.

2.1.1. NeRF for Fast Training

NeRF is effective in generating high-quality 3D scene views, but training and rendering times are slow. Müller et al. [8] recently proposed instant neural graphics primitives (instant-ngp) with a multi-resolution hash encoding to enhance the training speed of NeRF. Instant-ngp is primarily used to address the efficiency of NeRF when parameterizing fully connected neural networks. The method proposes an encoding approach that makes it possible to implement NeRF using a smaller scale network without incurring a loss of accuracy. The network is augmented by a multi-resolution hash table [13] implementation of feature vectors, performing optimizations based on stochastic gradient descent. The multiresolution structure facilitates GPUs parallelism and is able to reduce computation by eliminating hash collisions. Instant-ngp opens up new possibilities in the field of computer graphics, especially in the areas of virtual reality (VR), augmented reality (AR), and computer-aided design (CAD), through efficient data representation and rendering techniques [14–16].

2.1.2. VR-NeRF

NeRF's ability to quickly generate photo realistic 3D image visions and its low cost of hardware adaptation is a strong attraction for extended reality (XR) hardware, which is still in the early stages of development. NeRF is still in the early stages of development in the XR space, but it is inevitable that it will be used in the future to build certain VR/AR game content or meta-universe worlds. Li et al. [17,18] proposed a NeRF framework that enables the viewing of immersive reconstructed scenes in VR and also implements interactive features with NeRF models. However, this implementation simply projects the reconstructed scene onto a 2D plane, and the model itself still lacks geometric collision data, thus only enabling some simple appearance editing. Although the Magic NeRF Lens [19] was later proposed to trim down the NeRF model, it still needs to be fused with the computer-aided design (CAD) models.

2.2. 3D Gaussian Splatting

Recently, Kerbl et al. [6] proposed 3D Gaussian splatting as a 3D representation of a display that compresses 3D scene information using a set of 3D anisotropic Gaussian kernels, each with a learnable mean, opacity, and covariance matrix Σ.

In contrast to NeRF with implicit scene representation, Gaussian splatting provides an explicit representation that can be seamlessly integrated with post-processing operations (e.g., animation and editing). In addition, the efficient rasterization and superior 3D Gaussian image rendering quality facilitate its integration with virtual reality (VR) technologies. By using 3D Gaussian splatting technology, it is possible to achieve more realistic and detailed image rendering effects in virtual reality (VR) environments. This includes finer processing of lighting, shadows, and reflections in the scene [20], which improves the overall image quality and realism and further promotes the development of virtual reality (VR) technology [21].

There are four main parts to 3D Gaussian Splatting: (1) Get an initialized sparse point cloud via structure-from-motion (SfM) [22]; (2) Generating 3D Gaussian ellipsoid sets using an initialized point cloud; (3) Rendering a new 2D view; (4) Optimized scene representation. We discuss these four steps in detail below. The rendering flow of 3D Gaussian splatting is shown in Figure 2.

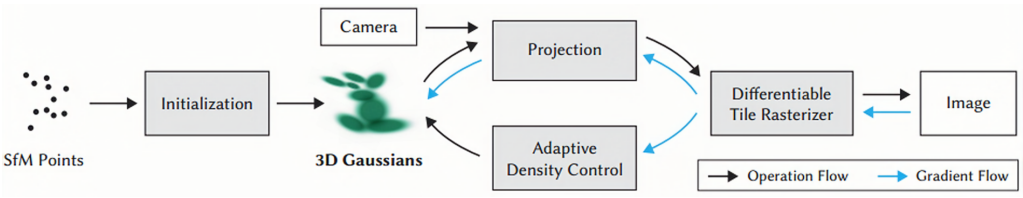


Figure 2. Rendering Process for 3D Gaussian Splatting [6].

SfM initializes the sparse point cloud. Beginning with a sparse SfM point cloud, a series of 3D Gaussian distributions are established. Initially, an initial point cloud is generated using COLMAP [22,23], wherein the COLMAP loop leverages provided data to estimate camera parameters and image position information, which can be inferred from a collection of photographs. This method preserves the favorable attributes of the continuous volume radiation field for scenario optimization while circumventing superfluous computations in vacant space. It furnishes foundational data for subsequent scene reconstruction, thereby enhancing the precision and dependability of the structural recovery process.

Creating 3D Gaussian ellipsoid sets. The 3D Gaussian splatting uses the SfM point cloud to initialize a set of mean values of 3D Gaussian random variables, which are the centroid positions of the Gaussian ellipsoid. Each Gaussian random variable can be defined in terms of its mean $\mu \in \mathrm{R}^3$ and covariance matrix $\Sigma \in \mathrm{R}^{3\times3}$ with probability density of

$$f(\mathrm{x}|\mu,\Sigma) = \frac{1}{\sqrt{(2\pi)^3|\Sigma|}} \exp\left(-\frac{1}{2}(\mathrm{x}-\mu)^T\Sigma^{-1}(\mathrm{x}-\mu)\right), \tag{7}$$

where the covariance matrix Σ contains the rotation matrix R of the Gaussian ellipsoid and the scaling matrix S in each axis.

$$\Sigma = RSS^TR^T \tag{8}$$

In addition, each Gaussian has a multiplier $\alpha \in [0, 1]$ indicating the opacity of the captured radiated light. In 3D Gaussian, the point cloud colors are represented using the spherical harmonics function, which makes the point cloud present different colors at different angles and facilitates iterative efficiency. The spherical harmonics function weakens the high-frequency information to some extent, which is essentially a lossy compression, and turns discrete information into continuous information, allowing gradients to be computed and iterated. The spherical harmonics function and its mean, covariance, and opacity are optimized so that a complete representation of the 3D scene can be learned using only a discrete set of points.

Rendering the new view. To render the view, the 3D space is projected onto the 2D screen space, and pixel colors are computed from near to far in order of their z-depth [24]. 3D Gaussian splatting is also rendered by a rendering method similar to point-based rendering [25], where the color C of each pixel is computed from the N ordered points in the point cloud within a certain radius that can affect the pixel. The rendering equation is:

$$C = \sum_{i\in N} c_i\alpha_i \prod_{j=1}^{i-1}(1-\alpha_j), \tag{9}$$

where c_i is the color of each point, α_i represents the opacity value of the current point i, and α_j represents the opacity value of the point before i. This α_i can be rewritten as:

$$\alpha_i = (1 - exp(-\sigma_i\delta_i)) \tag{10}$$

where δ_i is the distance between adjacent samples. This can be rewritten as:

$$\delta_i = t_{i+1} - t_i \tag{11}$$

Optimizing 3D scene representation. The entire 3D Gaussian splatting process, both the training and the 3D rendering, is end-to-end. This enables back propagation and gradient-based optimization. Sparse point-based rendering faces some additional challenges in terms of the number of points used, so in this step the parameters of the 3D Gaussian (covariance, position, opacity, spherical harmonic function) are optimized for a given image sequence and viewing angle. This step can be implemented using gradient descent or other optimization algorithms with the goal of minimizing the reconstruction error. Additionally, adaptive density control can be used to adjust the number and distribu-

tion of 3D Gaussians for optimal rendering. The loss function used to optimize the scene representation and rendering is as follows:

$$\mathcal{L} = (1 - \lambda)\mathcal{L}_1 + \lambda\mathcal{L}_{\text{D-SSIM}} \tag{12}$$

Minimizing the first item minimizes the pixel difference between the 2D rendering and the real image at the same viewing angle. The purpose of minimizing the second term is to increase the SSIM [26] between images.

2.3. System Design and Implementation

This section provides a detailed overview of our system architecture and the development process. Figure 3 illustrates the architectural diagram of our system.

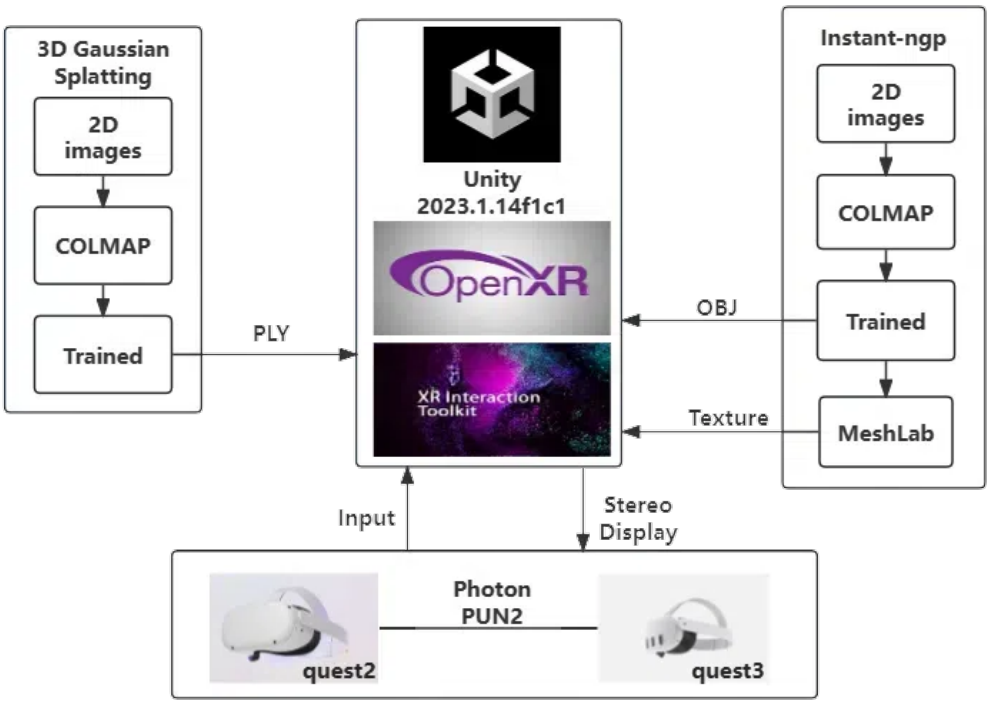


Figure 3. System Architecture Diagram. The primary focus is on the integration of 3D Gaussian splatting and instant-ngp algorithms within Unity3D, as well as the utilization of VR devices for immersive interaction within the virtual scene.

2.3.1. 3D Gaussian Splatting in VR

This section delineates the procedures for inspecting a three-dimensional Gaussian scene through virtual reality (VR) visualization techniques. It encompasses various aspects, including data preparation, algorithmic implementation, and the seamless integration of these processes within a VR environment.

View a 3D Gaussian splatting scene in Unity3D. Initially, the entire scene must be rendered using the 3D Gaussian splatting technique. In this study, we selected the Bicycle dataset from Mip-NeRF 360 [27] as the scene of interest. The Mip-NeRF 360 dataset offers a 360-degree panoramic image, enabling the creation of virtual environments with comprehensive viewpoints. This feature is particularly beneficial for virtual reality (VR) and augmented reality (AR) applications, as it allows users to freely observe and navigate the scene from any direction.

In our development efforts, we have chosen the Unity3D game engine [28] due to its numerous advantages for virtual reality (VR) development. These advantages include cross-platform compatibility, a robust graphics engine and animation system, an extensive resource library, and significant community support. The Unity3D platform offers a user-

friendly visual editor and a streamlined development process, enabling developers to efficiently create immersive virtual experiences and deploy them across a wide range of devices. The following sections detail the process of importing a 3D Gaussian scene into Unity3D, focusing on the two main components: importing the PLY file and editing the imported scene.

Importing PLY File. Aras-p has developed a renderer for Unity3D, known as UnityGaussianSplatting [29], which facilitates the rendering of Gaussian splatting within Unity3D without relying on OpenXR. Utilizing this renderer, we imported the Bicycle scene, which was trained 30,000 times, into Unity3D in the form of a PLY file. The imported scene will serve as the foundation for the entire fire drill system. Figure 4 displays the result of the import process.

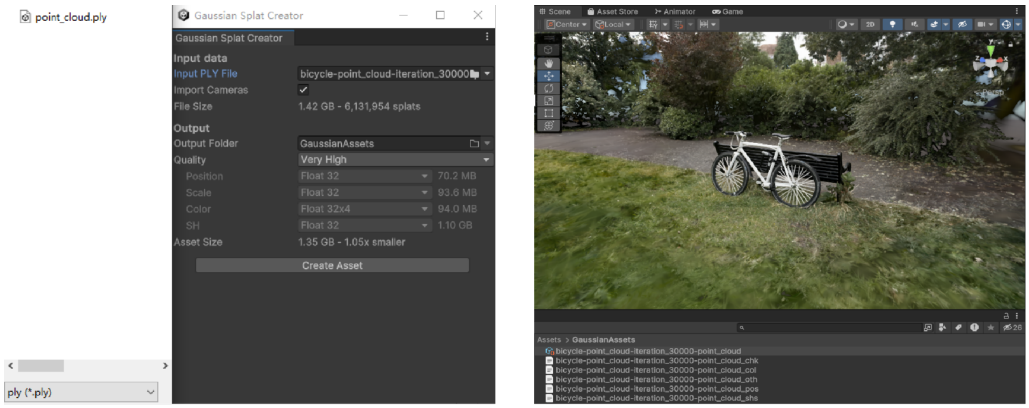


Figure 4. Importing a trained PLY file into Unity3D. (**a**) Creation of 3D Gaussian Assets. (**b**) Scene post-import.

Scene Cleaning. The imported scene is refined using the GaussianCutout script within the renderer. This script employs Gizmos with varying transparency and shapes, based on the current and selected states of objects, to facilitate visualization and manipulation within the editor. We utilized the ellipse pruner and cube pruner tools to post-process the scene, thereby removing any unwanted elements. The pruning process is illustrated in Figure 5.

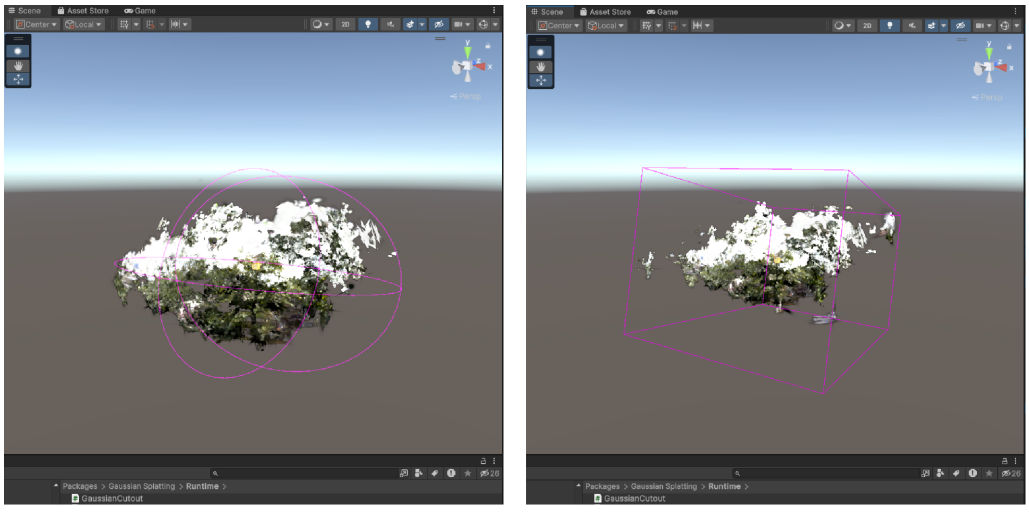


Figure 5. Cleaning the imported scene. (**a**) Ellipse pruner. (**b**) Cube pruner.

3D Gaussian splatting scene import virtual reality (VR). The primary VR application utilizes the OpenXR SDK [30] for virtual reality (VR) development, with a specific emphasis on Oculus Quest 3 and Oculus Quest 2 VR hardware equipped with hand tracking

capabilities. This setup enables users to navigate through realistic scenes and interact with objects within the virtual environment. The VR experience is based on a customized scene that necessitates the addition of colliders, as it lacks inherent collision data. To facilitate smooth navigation, a box collider named "Ground" is incorporated to simulate the ground surface (see Figure 6). Furthermore, teleportation functionality for players and objects is implemented using a "Teleportation Area" script from the XR Interaction Toolkit plugin, ensuring seamless movement within the VR environment [31].

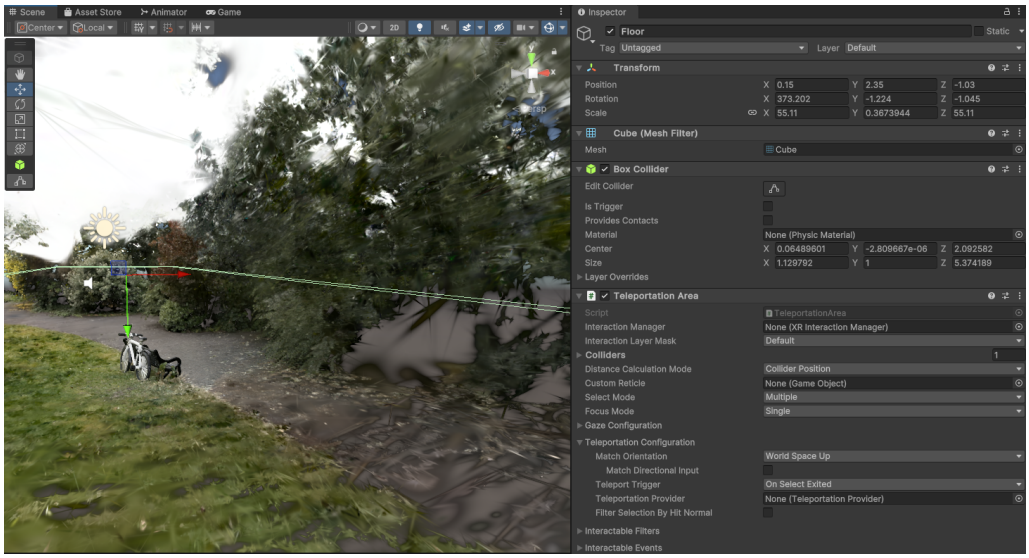


Figure 6. Adding Box Collider and "Teleportation Area" script.

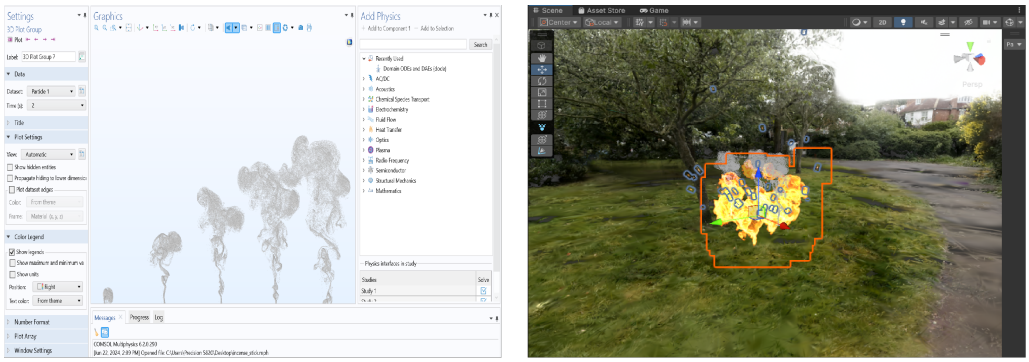
The XR Interaction Toolkit [32] offers a robust and flexible set of development tools that facilitates the creation of interactive and portable VR and AR applications. It provides developers with a wide range of commonly used interaction modes and components, along with support for customization and extensions. This significantly enhances the efficiency and flexibility of the development process. Consequently, this study will utilize the XR Interaction Toolkit for further development.

2.3.2. Creating Fire and Smoke Effects

The development of fire and smoke effects stands as a crucial component within this research endeavor. Concurrently, to enable the subsequent implementation of fire extinguishing functionalities, the design of scripts pertaining to fire control becomes imperative. Importing fire and smoke effects. The Unity3D particle system [33] is utilized to model entities characterized by fluid-like behavior, including liquid flow, smoke, clouds, flames, and similar dynamic elements. Each individual particle serves as a constituent of the overall flowing entity, with their collective behavior contributing to the manifestation of the complete entity's visual effect. For instance, in the context of a flame explosion effect, each particle represents a minuscule component of the explosion; when numerous particles aggregate to form larger clusters, the resultant performance conveys the comprehensive explosion effect. In this scene, the flame and smoke effects are generated using the particle system. By incorporating appropriate flame and smoke materials and textures onto the original particles within the particle system, fine-tuning parameters such as particle lifespan, color variations (e.g., color over lifetime), and other attribute adjustments enable the realistic simulation of flame and smoke effects.

Given the diverse array of parameters inherent in particle system components, we acquire project-specific resources from Unity3D asset store [34] and standard assets library to align with the project's needs. Following necessary adjustments, these resources are seamlessly integrated into the particle system to ensure precise placement of fire and smoke within the scene.

To ensure that our smoke conforms to the physical laws of real-world environments, we conducted a numerical simulation based on computational fluid dynamics (CFD) by taking the following steps: First, we used COMSOL Multiphysics, a numerical analysis software based on Finite Element Methods, to perform a three-dimensional physical simulation of smoke, leveraging its powerful multiphysics simulation capabilities to accurately model the dynamics of smoke in fire scenarios. For this purpose, we selected a smoke simulation case [35] provided by COMSOL (see Figure 7a). After completing the simulation, we generated VTU files to store the simulation data. These files effectively store complex three-dimensional fluid dynamics data, including velocity fields, pressure fields, and smoke concentration. Next, we used Paraview software to convert the VTU files into X3D files. Paraview is an open-source, multi-platform data analysis and visualization application that allows for detailed visualization of simulation data [36]. Finally, we used Blender software to convert the X3D files into FBX files that can be imported into Unity3D (see Figure 7b). This process allowed the physically accurate smoke simulation data to be seamlessly integrated into the Unity3D VR environment. It is also noted that the aforementioned approach is our preliminary attempt at incorporating physical simulation into VR. In the future, we will delve deeper into this issue by introducing real-time physical simulation techniques, such as physics-informed neural networks [37,38] and model order reduction [39,40].



(a) (b)

Figure 7. Importing smoke with physical laws into Unity3D. (**a**) COMSOL simulation results. (**b**) Import smoke.

Design of fire control script. We designed a script called "MyExtinguishableFire" script to control the fire in the scene. The size of the fire depends on the fire's own transform.localScale property, which is a three-dimensional vector; each dimension in the vector determines the size ratio in the x, y, and z directions; the fire is largest when the size ratio is 1 and is extinguished when the size ratio is zero.

Considering that the fire in this project is not extinguished naturally but gradually extinguished through fire extinguishing measures, a m_Maxlife value and a m_Life value indicating the size of the fire should be designed. In addition, there needs to be a m_ExtinguishRate and a m_RecoveryRate. In each frame of the program operation, a new m_Life value is calculated based on these two factors, and the fire situation is calculated based on the m_Life value and assigned to the teansform.localScale property. Algorithm 1 Indicates update fire size.

Attach the box collider component to facilitate detection of collisions with the fire extinguisher jet particles. In order to effectively extinguish the fire, the root of the fire extinguisher jet flames must be employed. Consequently, the thickness of the box collider component of the Fire object is suitably adjusted to encompass the base of the fire. Add the modified fire object to a reasonable position in the scene (see Figure 8).

Algorithm 1 Update fire size

Function Update()
if not $m\_isExtinguished$ and ($m\_Life < m\_Maxlife$ or $m\_ExtinguishRate \neq 0$) **then**
  $m\_Life \leftarrow m\_Life + (m\_RecoveryRate - m\_ExtinguishRate) \times$ Time.deltaTime
  if $m\_Life \leq 0$ **then**
    $m\_Life \leftarrow 0$
    $m\_isExtinguished \leftarrow$ true
    start Extinguishing Coroutine
  end if
  if $m\_Life \geq m\_Maxlife$ **then**
    $m\_Life \leftarrow m\_Maxlife$
  end if
  update fireParticleSystem.transform.localScale **to** $Vector3.one \times \frac{m\_Life}{m\_Maxlife}$
end if

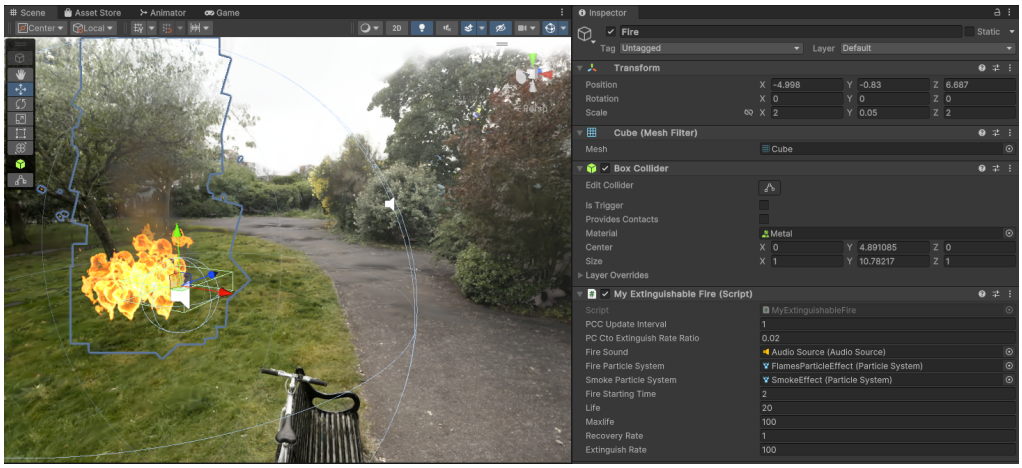


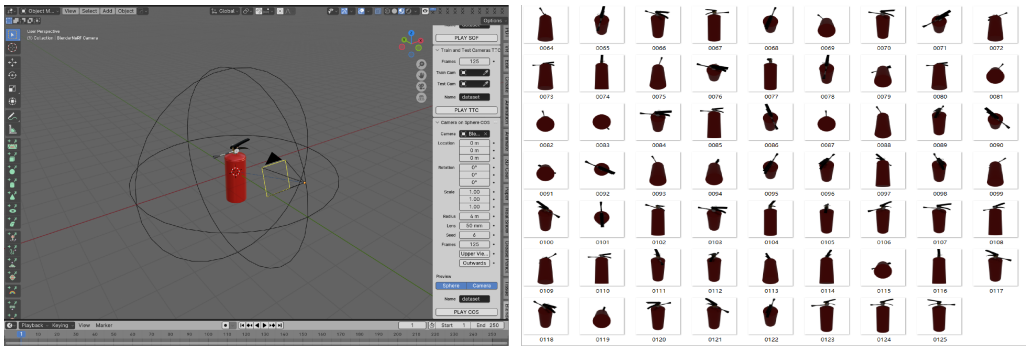
Figure 8. Incorporating Fire into the scene involves adding a collider and control script to the fire object. The image illustrates this process.

2.3.3. Generate Fire Equipment Model

In this fire drill system, the fire extinguisher is the tool to extinguish the fire extinguisher. We need to add a fire extinguisher model. We use instant-ngp to generate the fire extinguisher mesh model. In the initial stage, we need to first have a mock-up of the fire extinguisher equipment. We searched the official blenderco [41] website for a model that suits us and opened the model in Blender. To train the model using instant-ngp, two sets of data are required: one set of data is the photos taken by the camera along a distance without angle, and the other set of data is the camera pose information of these photos.

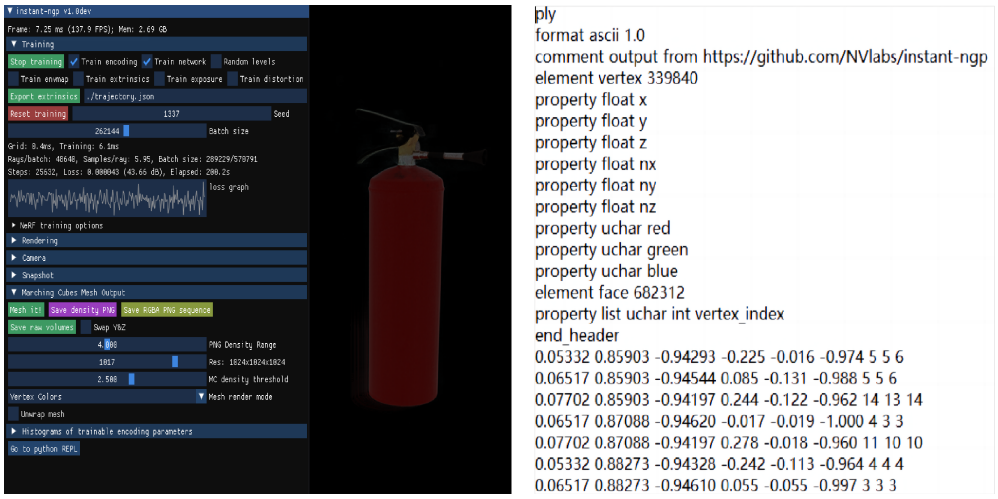
BlenderNeRF [42] is the easiest and fastest way to create synthetic instant-ngp datasets within Blender. Obtain renders and camera parameters with a single click, while having full user control over the 3D scene and camera. Place the camera in a reasonable position from the fire extinguisher, make it face the center of the fire extinguisher model, and set the camera's shooting trajectory. Here we set the trajectory as a sphere in order to generate a wider range of photos (see Figure 9a). After running, 125 png photos are generated, and a json file containing camera parameter information is generated (see Figure 9b).

Generate mesh and texture. Using instant-ngp to train the fire extinguisher model from the generated photos and json file shown in Figure 10a. Adjust the training parameters and save the model in PLY file format (see Figure 10b). Since the fire extinguisher model generated by instant-ngp cannot directly generate texture information, if it is directly imported into Unity3D, the material will be lost, so we first import the model into MeshLab [43], and then use this tool to export the texture map and export the model as an obj file and texture file, and import the generated file into Unity3D.



(a) (b)

Figure 9. Colmap generates pre-trained images. (**a**) The camera captures images from various angles around the model. (**b**) A set of 125 png images is generated.

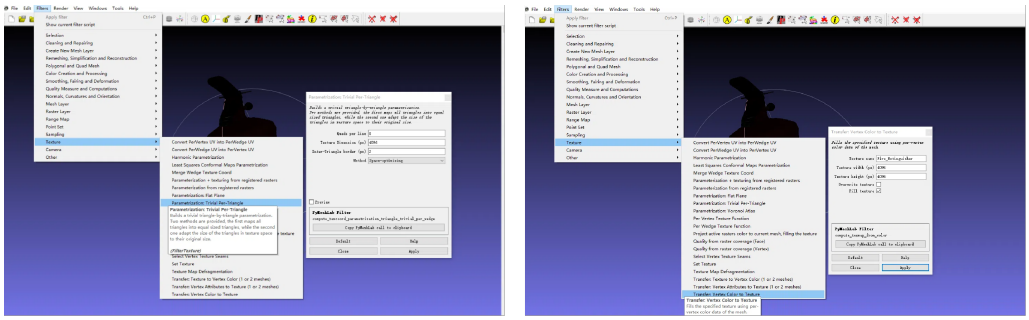


(a) (b)

Figure 10. NeRF generated using instant-ngp. (**a**) Creation of the fire extinguisher model. (**b**) The PLY file format.

The generated PLY file includes vertex coordinates, vertex colors, vertex normal vectors, and face sheets (see Figure 10b). First, we generate a UV map corresponding to the vertex coordinates of the object surface. Here we adjust the texture dimension to 4096 (see Figure 11a). Then we convert the vertex colors to the generated UV texture map (see Figure 11b). When converting, we need to set the length and width of the UV texture map; here it is set to the same size as the texture dimension. Mapping file is saved in png format.

Importing fire extinguishing equipment into Unity3D. Import the generated texture file and obj file into Unity3D; you can see that the fire extinguisher model can be clearly displayed in Unity3D (see Figure 12). The user ended up using the handle controller to implement the fire extinguisher function, so we added the appropriate components and scripts to the imported fire extinguisher model.



(a) (b)

Figure 11. Generate texture maps.

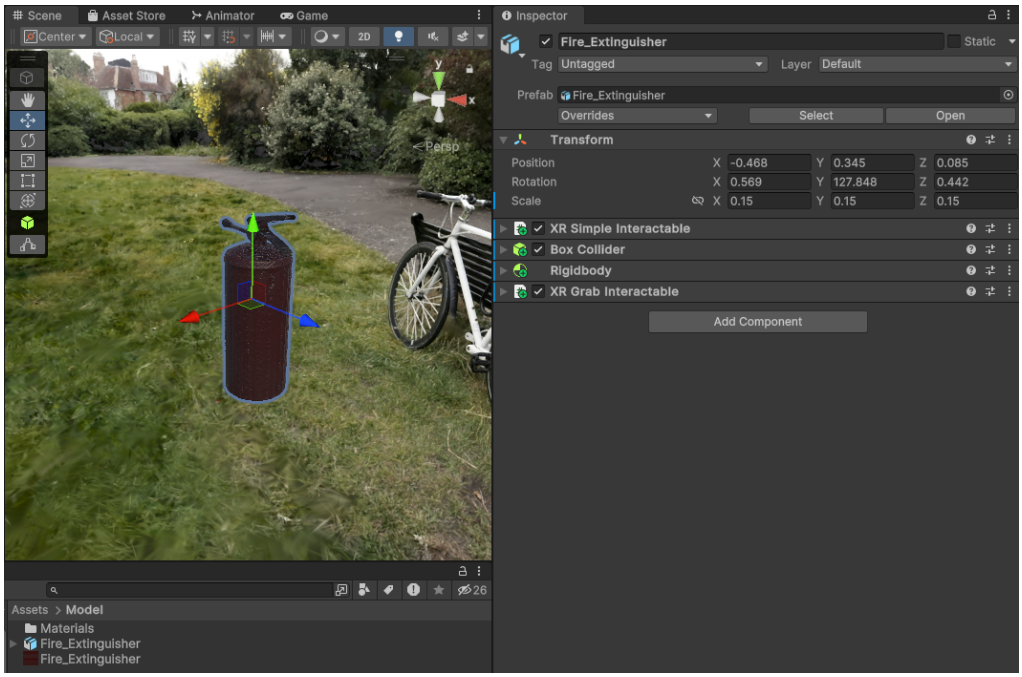


Figure 12. Incorporating fire suppression equipment into Unity3D.

2.3.4. Realization of the Fire Extinguishing Function

Importing the Jet Effect Particle System. The realization of fire extinguishing function is one of the key contents of this system. This part will introduce how to use Unity3D's "particle collision detection" function to realize fire extinguishing. We use the particle system in the standard repository provided by Unity3D and use the Hose prefab in this folder to add this prefab to the scene as a child object of the fire extinguisher, and adjust the position of the spray particles to the nozzle position of the fire extinguisher. The Figure 13 shows the result after adding.

Interactive fire suppression using particle system collisions. In this project, the principle of using particle collision to realize fire extinguishing interaction is: according to the cumulative number of times the fire extinguisher spray particles collide with the fire object in a certain period of time to update the extinguishing coefficient of the fire object. The more accurate the direction of the spray particles, the greater the extinguishing coefficient; as long as the extinguishing coefficient is greater than the recovery coefficient, the fire can be made smaller.

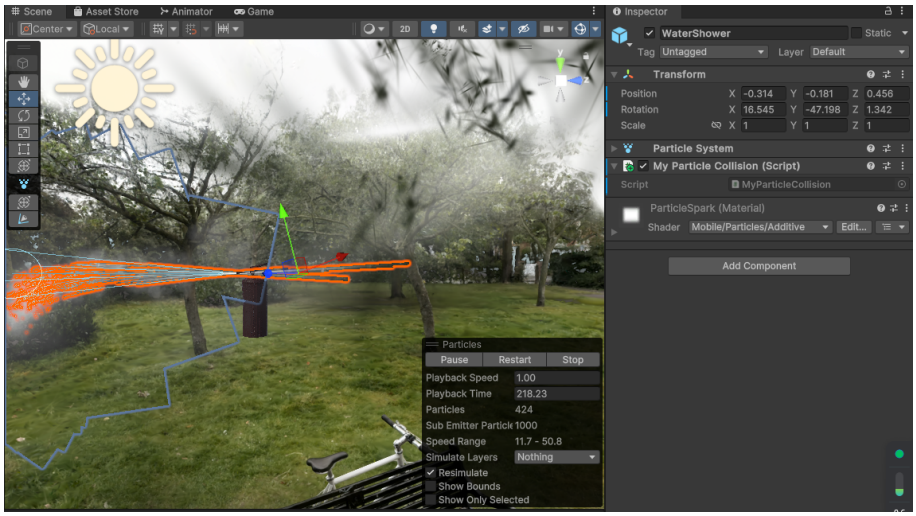


Figure 13. Align the spray particles position relative to the fire extinguisher.

When extinguishing particles emitted from the fire extinguisher collide with the collider of the "Fire" object, a particle collision event is triggered. In the callback function for this event, the number of particles that have collided with the fire object during this event is recorded and passed to the "MyExtinguishableFire" component of the fire object. To achieve this, it is necessary to modify the "MyExtinguishableFire" script by adding a function to receive the number of particle collisions. This function will calculate the new extinguishing rate based on the accumulated number of particle collisions over a certain period (e.g., 1 s). Additionally, the accumulated collision count and extinguishing rate will be reset at regular intervals (e.g., every 1 s). Algorithm 2 describes the callback function for particle collision events.

Algorithm 2 Particle Collision Handling

Function OnParticleCollision(other)
firehit ← null {Variable to store the collided object's MyExtinguishableFire component}
hitcount ← 0 {Variable to store the number of particles colliding with the fire object in one collision event}
numCollisionEvents ← m_ParticleSystem.GetCollisionEvents(other, m_CollisionEvents) {Get the list of collision events for all particles colliding with the object}
for $i = 0$ to numCollisionEvents **do**
    col ← m_CollisionEvents[i].colliderComponent
    fire ← col.GetComponent(MyExtinguishableFire)
    if fire $\neq$ null **then**
        hitcount ← hitcount + 1
        firehit ← fire
    end if
end for
if firehit $\neq$ null **then**
    firehit.HitByExtinguishParticleCollidor(hitcount) {Pass the collision count to the fire object}
end if

2.3.5. Multi-User Fire Fighting

The photon multiplayer game engine enables multi-user applications, so the computers must synchronize their environments, and in addition, the fires must be synchronized because the participants must recognize when other players are using fire extinguishers to put out the fires. We created a photon unified network (PUN) server on the photon website. PUN is a Unity3D package for multiplayer applications that implements and extends

the networking capabilities integrated into Unity3D. At the beginning of the program, all participants are automatically logged into the same room. After that, the scenes will be synchronized.

Our scene supports multiple people sharing, so we made a prefabricated component "Network Player" (see Figure 14a) in our program, which contains the camera and left and right hand controllers on top of the prefabricated component. This prefab object is equivalent to our real user, and when we package the program, each time we run it, a new player object is added to the scene.

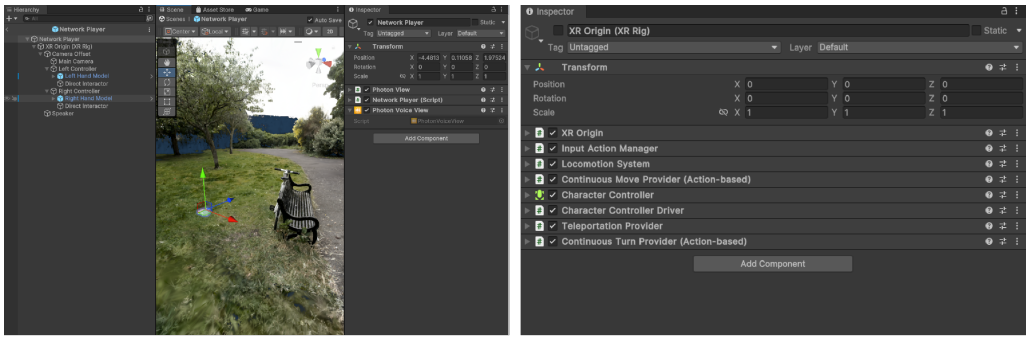


Figure 14. Create a "Network Player" prefab. (**a**) This prefab serves as a representation of the user, and upon launching the application on various devices, an instance of this prefab is instantiated within the scene. (**b**) We have incorporated scripts into the prefab to facilitate functionalities such as scene navigation and fire extinguisher control.

To enhance the functionality and realism of our VR firefighting system, we have integrated a communication exchange module. Utilizing Photon Voice [44], we seamlessly added real-time voice communication features to the system without the need for complex network coding. Photon Voice provides a comprehensive set of APIs and components, enabling developers to implement voice calls between players, voice chat rooms, and other communication functionalities with ease. This addition significantly benefits the firefighting training experience by allowing real-time coordination and communication among users. Firefighting often requires precise teamwork and rapid decision-making, both of which are facilitated by clear and immediate verbal communication. The voice communication module enables users to simulate real-world firefighting scenarios more accurately, improving their ability to respond to emergencies collaboratively. Figure 15 illustrates the component that integrates voice communication into the VR scene, thereby enhancing the overall immersive experience and training efficacy.

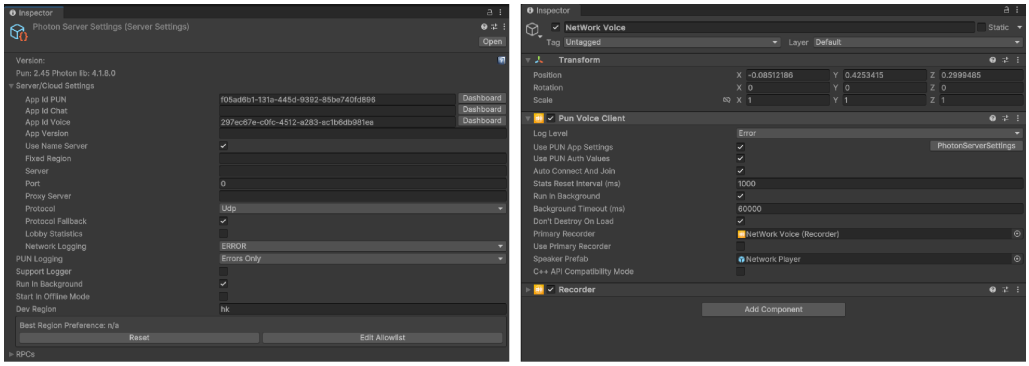


Figure 15. Adding Photon Voice Components.

3. Results

This section focuses on testing the developed immersive virtual reality firefighting experience system, introducing the hardware equipment and user experience of the system

respectively, and evaluating the performance of the system based on user feedback. We chose Oculus Quest 2 with a native rendering resolution of 1920 × 1832 and Oculus Quest 3 with a native rendering resolution of 2208 × 2064 for the tests and used streaming to connect to the workstations to ensure data streaming performance. All experiments were conducted on workstations with the following hardware specifications:

- CPU: Intel Core i9-10920X CPU @ 3.50 GHz
- RAM: 64 GB (3200 MHz/3200 MHz)
- GPU: NVIDIA GeForce RTX 3090
- Hard-Disk Drive: 2 TB SSD

In order to test the virtual fire drill system proposed in this paper, which supports multiplayer collaboration, we recruited 20 graduate students, divided into 10 groups of 2 users each, to test the usability of the system. Before the test, a simple pre-training training was conducted for 20 trainees to let them understand the basic functions of the system, familiarize with the interactive interface, and learn how to use the VR head-mounted display, so as to avoid encountering barriers to the use of hardware during the training test, which would affect the training effect and cause bias in the evaluation results.

The test tasks were as follows: (1) each user successively used different VR devices to move freely in the scene, control the virtual roaming, and perform the fire-fighting task individually; (2) two users cooperated to fight the fire. After the test is completed, each student is asked to fill out the questionnaire. The questionnaire is divided into 3 parts: user experience, training effect, and system optimization, with a total of 9 evaluation indicators and a total of 10 points for each indicator. According to the questionnaire, the corresponding scores of each index are organized so as to evaluate the effectiveness of the system (Table 1).

In order to further evaluate the difference between our fire extinguishing system and actual fire extinguishing, we invited 10 trainees who have actually used fire extinguishers to complete fire extinguishing to experience our system. After the experience, we filled out a questionnaire and analyzed the results. The Likert scale's 7-level scale [45] was adopted in the evaluation criteria, in which 1 represents "not realistic at all" and 7 represents "very realistic". In statistical analysis, the mean and standard deviation of the ratings given by all users are calculated, and the specific evaluation results are shown in Table 2.

Table 1. Evaluation results.

Firefighting System	Items in Question	Oculus Quest 2	Oculus Quest 3
user experience	Does it have a good sense of immersion?	8.1	8.5
	How does the interaction feel?	8.1	8.3
	How comfortable is it to wear?	7.5	8
training effect	What about the authenticity of the fire?	8.1	8.4
	Is there an understanding of the use of fire extinguishers?	6	6
	How difficult is it to extinguish a fire?	3.5	3.2
system optimization	Do you think the system is complete?	7.7	7.7
	Do you feel a sense of vertigo in the scene?	4	3.5
	Is multiplayer collaboration well-designed?	7.6	7.6

Table 2. Subjective evaluation of users.

Evaluation Index	VR Firefighting Training	Average	Standard Deviation
Environment	Is the simulated smoke and sound real in the virtual environment?	4.8	1.1
Physical Sensation	Are virtual fire extinguishers realistic compared to real ones?	5.1	1.1
Emergency Response	Does the virtual fire extinguishing environment lack real emergency perception?	5.2	1.7
User Security	Does the body feel unwell in the virtual fire fighting environment?	2.5	1.2
Learning Curve	Can virtual fire fighting training develop fire fighting skills?	4.9	1.4

4. Discussion

From Table 1, we can conclude that the Oculus Quest 3 is a better experience both in terms of user experience and training effectiveness. This shows that different VR devices have a big impact on the user's experience. In terms of system usability, the overall system, system quality, information quality, and interface quality met relatively consistent standards. In terms of stream experience [46], interest and satisfaction met basic criteria. In addition, users perceived the system to have good synergy and to make performing firefighting tasks more interesting. Based on the Table 2, the virtual fire drill system offers several advantages, such as high realism in simulating smoke and sound, realistic virtual fire extinguishers, and good user safety. This indicates that it effectively meets practical training needs. However, there are limitations, particularly in the emergency response realism, where users felt a lack of urgency. The higher standard deviation in this area suggests varied user experiences. Actual firefighting missions may face a variety of environmental changes and uncertainties, such as changes in wind direction and obstacles. Virtual environments may not be able to fully simulate these variations and uncertainties and, therefore, may not provide sufficient training in response capabilities. To enhance the sense of urgency, personalizing training and continuously collecting feedback for system optimization can help make virtual training more comprehensive and effective in replacing or supplementing real-world training. In summary, the system demonstrates good usability, fosters a strong sense of collaboration and enhances the user experience [47].

We compare the findings of our study with other recent research on the virtual simulation of forest fires. Several studies have highlighted the effectiveness of these technologies in enhancing training outcomes and preparedness.

For instance, Moreno et al. [48] conducted a study using VR to simulate forest fire scenarios, focusing on the training of emergency response teams. They used a combination of high-resolution terrain modeling and dynamic fire spread algorithms to create realistic training environments. Their results indicated significant improvements in the participants' decision-making skills and response times. Compared to our study, which utilizes 3D Gaussian Splatting and Neural Radiance Fields (NeRFs) for more detailed and immersive fire scene reconstruction, Moreno et al.'s approach primarily focused on terrain accuracy and fire dynamics.

Similarly, Heyao et al. [49] explored the use of VR in community education on forest fire safety. Their VR simulations included interactive scenarios where users could practice evacuation procedures and use firefighting equipment. They found that participants who underwent VR training had a better understanding of fire safety protocols compared to those who received traditional training. Our study differs in its emphasis on integrating multi-user experiences and collaborative firefighting tasks, which we believe further enhances the realism and practical application of the training.

Our results align with these studies in demonstrating the potential of VR to improve fire training outcomes. However, our use of advanced modeling techniques such as NeRF offers a higher degree of visual fidelity and interaction, potentially leading to even greater training benefits. One notable challenge in VR-based fire training highlighted by De Lorenzis et al. [50] is the need for scenarios that closely mimic real-world conditions to ensure transferability of skills. Our study addresses this by using NeRF to create highly realistic textures and dynamic interactions within the VR environment.

In summary, while existing studies provide strong evidence supporting the efficacy of VR in fire training, our research contributes to this body of knowledge by introducing more sophisticated modeling techniques and emphasizing collaborative training environments. Future work should continue to explore these advancements, focusing on longitudinal studies to assess the long-term impact of VR training on real-world firefighting performance.

5. Conclusions

In conclusion, using neural radiance fields (NeRFs) and 3D Gaussian splatting in VR firefighting training enhances immersion and effectiveness. Our system, integrating these techniques with Unity3D and head-mounted displays (HMDs), reconstructs realistic environments and firefighting equipment. This allows users to interact within virtual fire scenes, boosting engagement. The Photon PUN2 framework enables multi-user collaboration, improving training and teamwork skills. Experiments and surveys confirm that 3D Gaussian splatting enhances user experience and training outcomes. This VR application conserves resources and provides a realistic training experience with significant implications for emergency response training.

However, it is important to acknowledge the current limitations of this technology. The realism of fire behavior simulations in VR is not yet on par with the complex fluid dynamics observed in real-world fires, particularly in diverse forest ecosystems. Additionally, the transferability of skills learned in VR to actual firefighting scenarios requires further empirical study.

Our future work will focus on the development and refinement of these VR techniques to better simulate the varying conditions found in different forest ecosystems worldwide. We aim to create more accurate and effective training programs that can significantly aid in controlling forest fires globally. Collaboration with experts in forest ecology, fire science, and firefighting professionals will be essential to achieve these advancements.

Author Contributions: Conceptualization, H.L., R.C. and L.C.; Methodology, H.L. and R.C.; Software, H.L. and K.L.; Validation, K.L., X.W. and Z.F.; Formal Analysis, K.L., X.W. and Z.F.; Investigation, H.L., K.L. and Z.F.; Resources, X.W. and L.C.; Data Curation, K.L., Z.F. and X.W.; Writing—Original Draft Preparation, H.L., K.L. and R.C.; Writing—Review & Editing, H.L. and K.L.; Visualization, K.L.; Supervision, H.L., R.C. and L.C.; Project Administration, R.C. and L.C.; Funding Acquisition, H.L. and R.C. All authors have read and agreed to the published version of the manuscript.

Funding: This study was funded by the National Natural Science Foundation of China (NSFC) (No. 52274222), research project supported by Shanxi Scholarship Council of China (No. 2023-036), and the Natural Science Foundation of Shanxi (No. 202303021222020).

Data Availability Statement: The original contributions presented in the study are included in the article, further inquiries can be directed to the corresponding author.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Perović, I. Analysis of the publics' competence levels in preventive fire protection measures, firefighting and rescue of the people and property endangered by fire. *Vatrog. Upravlj. Pozar.* **2023**, *13*, 5–16.
2. Bellemans, M.; Lamrnens, D.; De Sloover, J.; De Vleeschauwer, T.; Schoofs, E.; Jordens, W.; Van Steenhuyse, B.; Mangelschots, J.; Selleri, S.; Hamesse, C. Training Firefighters in Virtual Reality. In Proceedings of the 2020 International Conference on 3D Immersion (IC3D), Brussels, Belgium, 15 December 2020; IEEE: New York, NY, USA, 2020; pp. 1–6.
3. Klačková, I.; Kuric, I.; Zajačko, I.; Tlach, V.; Wiecek, D. Virtual reality in Industry. In *Proceedings of the IOP Conference Series: Materials Science and Engineering*; IOP Publishing: Bristol, UK, 2021; Volume 1199, p. 012005.

4. Blackman, T. Virtual reality and videogames: Immersion, presence, and the performative spatiality of 'being there' in virtual worlds. *Soc. Cult. Geogr.* **2024**, *25*, 404–422. [CrossRef]
5. Romanoni, A.; Fiorenti, D.; Matteucci, M. Mesh-based 3D textured urban mapping. In Proceedings of the 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vancouver, BC, Canada, 24–28 September 2017; pp. 3460–3466. [CrossRef]
6. Kerbl, B.; Kopanas, G.; Leimkühler, T.; Drettakis, G. 3D Gaussian Splatting for Real-Time Radiance Field Rendering. *ACM Trans. Graph.* **2023**, *42*, 1–14. [CrossRef]
7. Mildenhall, B.; Srinivasan, P.P.; Tancik, M.; Barron, J.T.; Ramamoorthi, R.; Ng, R. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. *Commun. ACM* **2021**, *65*, 99–106. [CrossRef]
8. Müller, T.; Evans, A.; Schied, C.; Keller, A. Instant Neural Graphics Primitives with a Multiresolution Hash Encoding. *ACM Trans. Graph. (TOG)* **2022**, *41*, 1–15. [CrossRef]
9. Aaltonen, P. Networking Tools Performance Evaluation in a VR Application: Mirror vs. Photon PUN2. 2022. Available online: https://www.theseus.fi/bitstream/handle/10024/755310/Aaltonen_Pasi.pdf?sequence=2&isAllowed=y' (accessed on 30 June 2024).
10. Dangi, P.; Jain, A.; Samanta, D.; Dutta, S.; Bhattacharya, A. 3D Modelling and Rendering Using Autodesk 3ds Max. In Proceedings of the 2023 11th International Conference on Internet of Everything, Microwave Engineering, Communication and Networks (IEMECON), Jaipur, India, 10–11 February 2023; IEEE: New York, NY, USA, 2023; pp. 1–5.
11. Kajiya, J.T.; Von Herzen, B.P. Ray Tracing Volume Densities. *ACM SIGGRAPH Comput. Graph.* **1984**, *18*, 165–174. [CrossRef]
12. Tancik, M.; Srinivasan, P.; Mildenhall, B.; Fridovich-Keil, S.; Raghavan, N.; Singhal, U.; Ramamoorthi, R.; Barron, J.; Ng, R. Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 7537–7547.
13. Wang, H.; Yu, T.; Yang, T.; Qiao, H.; Dai, Q. Neural Physical Simulation with Multi-Resolution Hash Grid Encoding. In Proceedings of the AAAI Conference on Artificial Intelligence, Vancouver, BC, Canada, 20–27 February 2024; Volume 38, pp. 5410–5418.
14. Xu, L.; Agrawal, V.; Laney, W.; Garcia, T.; Bansal, A.; Kim, C.; Rota Bulò, S.; Porzi, L.; Kontschieder, P.; Božič, A.; et al. VR-NeRF: High-Fidelity Virtualized Walkable Spaces. In Proceedings of the SIGGRAPH Asia 2023 Conference Papers, Sydney, NSW, Australia, 12–15 December 2023; pp. 1–12.
15. Lu, C.Y.; Zhou, P.; Xing, A.; Pokhariya, C.; Dey, A.; Shah, I.N.; Mavidipalli, R.; Hu, D.; Comport, A.I.; Chen, K.; et al. DiVa-360: The Dynamic Visual Dataset for Immersive Neural Fields. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition 2024, CVPR 2024, Seattle, DC, USA, 17–21 June 2024.
16. Jing, W.; Wang, S.; Zhang, W.; Li, C. Reconstruction of Neural Radiance Fields with Vivid Scenes in the Metaverse. *IEEE Trans. Consum. Electron.* **2023**. [CrossRef]
17. Li, K.; Rolff, T.; Schmidt, S.; Bacher, R.; Frintrop, S.; Leemans, W.; Steinicke, F. Immersive Neural Graphics Primitives. *arXiv* **2022**, arXiv:2211.13494.
18. Li, K.; Rolff, T.; Schmidt, S.; Bacher, R.; Leemans, W.; Steinicke, F. Interacting with Neural Radiance Fields in Immersive Virtual Reality. In Proceedings of the Extended Abstracts of the 2023 CHI Conference on Human Factors in Computing Systems, Hamburg, Germany, 23–28 April 2023; pp. 1–4.
19. Li, K.; Schmidt, S.; Rolff, T.; Bacher, R.; Leemans, W.; Steinicke, F. Magic NeRF Lens: Interactive Fusion of Neural Radiance Fields for Virtual Facility Inspection. *Front. Virtual Real.* **2024**, *5*, 1377245. [CrossRef]
20. Dalal, A.; Hagen, D.; Robbersmyr, K.G.; Knausgård, K.M. Gaussian Splatting: 3D Reconstruction and Novel View Synthesis, a Review. *arXiv* **2024**, arXiv:2405.03417.
21. Jiang, Y.; Yu, C.; Xie, T.; Li, X.; Feng, Y.; Wang, H.; Li, M.; Lau, H.; Gao, F.; Yang, Y.; et al. VR-GS: A Physical Dynamics-Aware Interactive Gaussian Splatting System in Virtual Reality. *arXiv* **2024**, arXiv:2401.16663.
22. Schonberger, J.L.; Frahm, J.M. Structure-From-Motion Revisited. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 4104–4113.
23. Fu, Y.; Liu, S.; Kulkarni, A.; Kautz, J.; Efros, A.A.; Wang, X. COLMAP-Free 3D Gaussian Splatting. *arXiv* **2023**, arXiv:2312.07504.
24. Turkulainen, M.; Ren, X.; Melekhov, I.; Seiskari, O.; Rahtu, E.; Kannala, J. DN-Splatter: Depth and Normal Priors for Gaussian Splatting and Meshing. *arXiv* **2024**, arXiv:2403.17822.
25. Xu, Q.; Xu, Z.; Philip, J.; Bi, S.; Shu, Z.; Sunkavalli, K.; Neumann, U. Point-NeRF: Point-Based Neural Radiance Fields. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 5438–5448.
26. Hore, A.; Ziou, D. Image Quality Metrics: PSNR vs. SSIM. In Proceedings of the 2010 20th International Conference on Pattern Recognition, Istanbul, Turkey, 23–26 August 2010; IEEE: New York, NY, USA, 2010; pp. 2366–2369.
27. Barron, J.T.; Mildenhall, B.; Verbin, D.; Srinivasan, P.P.; Hedman, P. Mip-NeRF 360: Unbounded Anti-Aliased Neural Radiance Fields. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 5470–5479.
28. Nurym, N.; Sambetova, R.; Azybaev, M.; Kerimbayev, N. Virtual Reality and Using the Unity 3D Platform for Android Games. In Proceedings of the 2020 IEEE 10th International Conference on Intelligent Systems (IS), Varna, Bulgaria, 28–30 August 2020; IEEE: New York, NY, USA, 2020; pp. 539–544.

29. UnityGaussianSplatting. Available online: https://github.com/aras-p/UnityGaussianSplatting (accessed on 30 June 2024).
30. Lewis, C.; Harris, F. An Overview of Virtual Reality. *EPiC Ser. Comput.* **2023**, *88*.
31. Sayyad, E.; Sra, M.; Höllerer, T. Walking and Teleportation in Wide-area Virtual Reality Experiences. In Proceedings of the 2020 IEEE International Symposium on Mixed and Augmented Reality (ISMAR), Virtual, 9–13 November 2020; IEEE: New York, NY, USA, 2020; pp. 608–617.
32. Juránek, V. Virtual Reality Toolkit for the Unity Game Engine. Bakalárskáprace, Masarykova Univerzita, Fakulta informatiky. 2021. Available online: https://is.muni.cz/th/qyryn/thesis.pdf (accessed on 30 June 2024).
33. Zhang, B.; Hu, W. Game Special Effect Simulation Based on Particle System of Unity3D. In Proceedings of the 2017 IEEE/ACIS 16th International Conference on Computer and Information Science (ICIS), Wuhan, China, 24–26 May 2017; IEEE: New York, NY, USA, 2017; pp. 595–598.
34. Assetstore. Available online: https://assetstore.unity.com/ (accessed on 30 June 2024).
35. Smoke. Available online: https://cn.comsol.com/model/smoke-from-an-incense-stick-8212-visualizing-the-laminar-to-turbulent-transition-97501 (accessed on 30 June 2024).
36. Ayachit, U. *The ParaView Guide: A Parallel Visualization Application*; Kitware, Inc.: New York, NY, USA, 2015.
37. Raissi, M.; Perdikaris, P.; Karniadakis, G.E. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *J. Comput. Phys.* **2019**, *378*, 686–707. [CrossRef]
38. Zhu, D.; Yu, B.; Wang, D.; Zhang, Y. Fusion of finite element and machine learning methods to predict rock shear strength parameters. *J. Geophys. Eng.* **2024**, gxae064. [CrossRef]
39. Chen, L.; Wang, Z.; Lian, H.; Ma, Y.; Meng, Z.; Li, P.; Ding, C.; Bordas, S.P. Reduced order isogeometric boundary element methods for CAD-integrated shape optimization in electromagnetic scattering. *Comput. Methods Appl. Mech. Eng.* **2024**, *419*, 116654. [CrossRef]
40. Chen, L.; Lian, H.; Dong, H.W.; Yu, P.; Jiang, S.; Bordas, S.P.A. Broadband topology optimization of three-dimensional structural-acoustic interaction with reduced order isogeometric FEM/BEM. *J. Comput. Phys.* **2024**, *509*, 113051. [CrossRef]
41. Blenderco. Available online: https://blenderco.cn/ (accessed on 30 June 2024).
42. BlenderNeRF. Available online: https://github.com/maximeraafat/BlenderNeRF (accessed on 30 June 2024).
43. Dongqing, W.; Jian, G.; Pengfei, G.; Zhengtao, X. Research on Design Pattern of 3D Model Software Development Based on Meshlab. *Acad. J. Manuf. Eng.* **2020**, *18*, 164–172.
44. Wang, C.Y.; Sakashita, M.; Ehsan, U.; Li, J.; Won, A.S. Again, Together: Socially Reliving Virtual Reality Experiences When Separated. In Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems, Honolulu, HI, USA, 25–30 April 2020; pp. 1–12.
45. Batterton, K.A.; Hale, K.N. The Likert Scale What It Is and How to Use It. *Phalanx* **2017**, *50*, 32–39.
46. Bian, Y.; Yang, C.; Gao, F.; Li, H.; Zhou, S.; Li, H.; Sun, X.; Meng, X. A framework for physiological indicators of flow in VR games: Construction and preliminary evaluation. *Pers. Ubiquitous Comput.* **2016**, *20*, 821–832. [CrossRef]
47. Halbig, A.; Latoschik, M.E. A Systematic Review of Physiological Measurements, Factors, Methods, and Applications in Virtual Reality. *Front. Virtual Real.* **2021**, *2*, 694567. [CrossRef]
48. Moreno, A.; Posada, J.; Segura, Á.; Arbelaiz, A.; García-Alonso, A. Interactive fire spread simulations with extinguishment support for Virtual Reality training tools. *Fire Saf. J.* **2014**, *64*, 48–60. [CrossRef]
49. Heyao, H.; Tetsuro, O. Assessing the Sense of Presence to Evaluate the Effectiveness of Virtual Reality Wildfire Training. In *Advances in Networked-Based Information Systems: The 24th International Conference on Network-Based Information Systems (NBiS-2021)*; Springer: Cham, Switzerland, 2022; pp. 289–298.
50. De Lorenzis, F.; Pratticò, F.G.; Lamberti, F. Work-in-Progress—Blower VR: A Virtual Reality Experience to Support the Training of Forest Firefighters. In Proceedings of the 2022 8th International Conference of the Immersive Learning Research Network (iLRN), Vienna, Austria, 30 May–4 June 2022; IEEE: New York, NY, USA, 2022; pp. 1–3.