An Architecture for Constructing Mulsemedia Applications Using Real-Time Fluid Simulation

PEDRO JOAQUIM GONÇALVES DE CASTRO, LUÍS CLAUDIO RIBEIRO JUNIOR, RENATO DE OLIVEIRA RODRIGUES, and THIAGO RIBEIRO FURTADO, MultiSenS Lab, Federal Center for Technological Education Celso Suckow da Fonseca, Rio de Janeiro, Brazil
CELSO ALBERTO SAIBEL SANTOS, Federal University of Espirito Santo, Vitoria, Brazil
DÉBORA C. MUCHALUAT-SAADE, MidiaCom Lab, Fluminense Federal University, Niterói, Brazil
GHEORGHITA GHINEA, Department of Computer Science, Brunel University, Uxbridge, United Kingdom of Great Britain and Northern Ireland
DIEGO NUNES BRANDÃO, Federal Center for Technological Education Celso Suckow da Fonseca, Rio de Janeiro, Brazil
JOEL A. F. DOS SANTOS, MultiSenS Lab, Federal Center for Technological Education Celso Suckow da Fonseca, Rio de Janeiro, Brazil

The integration of sensory effects—such as wind, heat, and scent—into multimedia applications holds great promise for enhancing user immersion and Quality of Experience (QoE). However, simulating the propagation of such effects in dynamic environments presents considerable challenges, particularly in achieving real-time interactivity, physical plausibility, and perceptual coherence. This article introduces an interactive simulation system based on Smoothed-Particle Hydrodynamics (SPH), designed to model gaseous-like sensory effects for real-time applications. The proposed architecture allows authors to configure virtual environments, define sensor and actuator properties, and simulate the behavior of sensory effects under different conditions. A wind-with-temperature simulation scenario is presented as a proof of concept, demonstrating the feasibility of the approach. Experimental results indicate that the proposed model captures key physical trends observed in real-world measurements. Thus, the proposed system provides a foundation for the design and evaluation of future mulsemedia experiences.

This work was supported in part by Coordenação de Aperfeiçoamento de Pessoal de Nível Superior, Conselho Nacional de Desenvolvimento Cient fico e Tecnol gico, Fundação Carlos Chagas Filho de Amparo à Pesquisa do Estado do Rio de Janeiro, and FINEP funding.

Authors' Contact Information: Pedro Joaquim Gonçalves de Castro, MultiSenS Lab, Federal Center for Technological Education Celso Suckow da Fonseca, Rio de Janeiro, Brazil; e-mail: pedro.goncalves@aluno.cefet-rj.br; Luís Claudio Ribeiro Junior, MultiSenS Lab, Federal Center for Technological Education Celso Suckow da Fonseca, Rio de Janeiro, Brazil; e-mail: luis.junior.1@aluno.cefet-rj.br; Renato de Oliveira Rodrigues, MultiSenS Lab, Federal Center for Technological Education Celso Suckow da Fonseca, Rio de Janeiro, Brazil; e-mail: renato.rodrigues@aluno.cefet-rj.br; Thiago Ribeiro Furtado, MultiSenS Lab, Federal Center for Technological Education Celso Suckow da Fonseca, Rio de Janeiro, Brazil; e-mail: thiago.furtado@aluno.cefet-rj.br; Celso Alberto Saibel Santos, Federal University of Espirito Santo, Vitoria, Brazil; e-mail: celso.santos@ufes.br; Débora C. Muchaluat-Saade, MidiaCom Lab, Fluminense Federal University, Niterói, Brazil; e-mail: debora@midiacom.uff.br; Gheorghita Ghinea, Department of Computer Science, Brunel University, Uxbridge, United Kingdom of Great Britain and Northern Ireland; e-mail: george.ghinea@brunel.ac.uk; Diego Nunes Brandão, Federal Center for Technological Education Celso Suckow da Fonseca, Rio de Janeiro, Brazil; e-mail: diego.brandao@cefet-rj.br; Joel A. F. Dos Santos (corresponding author), MultiSenS Lab, Federal Center for Technological Education Celso Suckow da Fonseca, Rio de Janeiro, Brazil; e-mail: jsantos@eic.cefet-rj.br.



This work is licensed under Creative Commons Attribution International 4.0.

© 2026 Copyright held by the owner/author(s).
ACM 1551-6865/2026/10-ART286
https://doi.org/10.1145/3843224

CCS Concepts: • **Software and its engineering** → *Virtual worlds software*; • **Information systems** → **Multimedia information systems**;

Additional Key Words and Phrases: Mulsemedia, Gaseous-like Sensory Effects, Computational Fluid Dynamics, SPH

ACM Reference format:
Pedro Joaquim Gonçalves de Castro, Luís Claudio Ribeiro Junior, Renato de Oliveira Rodrigues, Thiago Ribeiro Furtado, Celso Alberto Saibel Santos, Débora C. Muchaluat-Saade, Gheorghita Ghinea, Diego Nunes Brandão, and Joel A. F. Dos Santos. 2026. An Architecture for Constructing Mulsemedia Applications Using Real-Time Fluid Simulation. *ACM Trans. Multimedia Comput. Commun. Appl.* 22, 10, Article 286 (October 2026), 21 pages. https://doi.org/10.1145/3843224

1 Introduction

A treasure hunter follows a series of clues that lead her to the middle of a desert. There, gusts of hot air are constant, creating an unsettling feeling of heat. As she searches for the treasure, our hunter encounters a thin, fickle source of cool air. This becomes stronger as she approaches until the treasure appears before her eyes.

The above scene would be delivered through the senses of hearing and vision in traditional multimedia applications. However, given that human communication is multisensory (a mixture of sight, hearing, touch, smell, and taste), it is desirable to use communication tools that explore other senses in addition to classical audiovisual interaction. **Multiple Sensorial Media (Mulsemedia)** [12] aims to fill that gap by including stimuli to other human senses in multimedia applications. As a Mulsemedia application, our previous treasure hunting scene would benefit greatly by allowing the user to feel the same hot and fresh gusts of air as our protagonist.

There are studies in the literature approaching the potential to improve **Quality of Experience (QoE)** [3, 8, 23] and information retention [1, 2, 50] of Mulsemedia applications in relation to multimedia ones. Others focus on creating devices to support sensory effects' rendering [41], extending presentation engines to support sensory effects [22, 43], or providing communication support [45, 46] between application and rendering devices.

As discussed by Jalal and Murroni [21] and Rodrigues et al. [41], it is not simple to include sensory effects in multimedia applications. Actuators responsible for rendering sensory effects have to be controlled by the Mulsemedia presentation engine in such a way that effects are perceived as in sync with the audiovisual content. Different approaches to dealing with this problem are presented in the literature according to the scenario at hand. Murray et al. [34], for example, ask users to place their heads in a fixed position (water gallon) while experimenting with an olfaction-based Mulsemedia application. This maintains the dispersion delay constant such that actuator commands can be sent in advance to compensate for network delay, actuator processing time, and effect dispersion. A similar approach is proposed by Ivanov et al. [20] by providing presentation engines with the ability to prepare actuators before the moment of execution of a sensory effect.

As presented by Rodrigues et al. [42], defining the amount of time to anticipate an effect and its intensity is also not an easy task. A first attempt would be to calculate the average time spent from the moment an actuator is activated to the moment the user perceives the sensory effect. Similarly, one would need to test different effect intensities (at the actuator) and identify the one that most closely matches what should be presented to the user. A problem that arises, however, is how to

cope with effect disambiguation when different effects in the environment affect the dispersion of each other. Understanding the interaction between effects, however, is not as straightforward as their delay and intensity.

Rodrigues et al. [41] make the point that understanding the propagation of sensory effects is key for solving the above issues. However, sensory effects may reach the user in different ways, depending on its type. As for olfactory and tactile sensations such as wind and fog, a previous study [42] showed that they propagate to the user using gaseous sources (air). This fluid-like behavior enables us to understand how they disperse in a given environment (e.g., the physical place where the application is being rendered), by simulating those fluids computationally. **Computational Fluid Dynamics (CFD)** methods implement the Navier–Stokes equations, which mathematically model the movement of most fluids occurring in nature [54].

This article extends the work of Rodrigues et al. [42] by presenting in more detail the proposed architecture for the simulation of fluid-like sensory effects, its implementation, and an evaluation indicating its feasibility as a way to calculate the effect behavior in a Mulsemedia presentation engine.

It is important to note that this work focuses on the physical modeling and simulation of sensory effect propagation, rather than on perceptual validation. While perceptual coherence and QoE are important goals in Mulsemedia systems, their evaluation requires user studies, which are outside the scope of this work. Instead, we focus on providing a physically based simulation tool that can support the design and calibration of such systems.

The remainder of this article is organized as follows. Section 2 presents related work that focuses on simulating sensory effects. Section 3 presents the proposed architecture and its instantiation. Section 4 presents the evaluation conducted with the proposed simulation system as an indication of its efficiency. Section 5 discusses limitations of this work. Finally, Section 6 concludes this article and presents future work.

2 Related Work

We conducted a systematic search in order to find studies related to our proposal in the literature. The search focused on studies that present sensory effect propagation simulation for controlling Mulsemedia applications. Below, we present the terms used to construct the search string:

- *Goal*: Simulation and emulation
- *Focus*: Propagation, dispersion, diffusion, and concentration distribution
- *Object*: Sensory effects, wind, airflow, spray, scent, aroma, haptics, (de)gustatory, odor, smell, and olfactory
- *Application*: Multimedia, Mulsemedia, multisensory, human-bound communication, **Virtual Reality (VR)**, virtual environment, extended reality, and augmented reality

The search string constructed from the above terms resulted in 65 papers. There was no limitation regarding the publication period or publication area. The search was conducted in Scopus to guarantee indexed papers. From that result, 15 papers were removed for being conference reviews, bundles of articles, repeated, or previous work. Besides, three papers were included from a previous search.

After further reading, 16 papers were considered out of the context of our work, and the remaining 36 were classified into 4 groups, as shown in Table 1: (i) source location, (ii) hardware construction, (iii) training/education, and (iv) propagation simulation. Section 2.1 presents related work in groups (i)–(iii), while Section 2.2 presents the ones in group (iv).

Table 1. Related Work Classification into Groups

Main Subject	List of References
Source Location	[4, 6, 7, 11, 25, 37, 38, 49, 55–57, 59, 60]
Hardware Construction	[9, 10, 16, 35, 47, 51]
Training/Education	[5, 14, 17, 18, 24, 32, 52, 53, 58]
Propagation Simulation	[19, 26, 28–31, 36, 39]

2.1 Source Location, Hardware Construction, and Training/Education

Thirteen papers present studies related to providing tools or techniques for locating a contaminant dispersion source in a given environment [4, 6, 7, 11, 25, 37, 38, 49, 55–57, 59, 60]. In those papers, simulators are built on top of CFD techniques for equipment training or testing. Chen et al. [6], for example, build a simulation to analyze the dispersion of chemical contaminants from an industry in Nanjing, China.

Six papers present studies related to building devices for sensory effect simulation [9, 10, 16, 35, 47, 51]. In those papers, CFD is used in the construction phase of actuator devices. Hu et al. [16], for example, use fluid dynamics formulae to create a vortex device for delivering scent effects near the user's face. The formulae and data collected in experiments, however, are specific to the device creation and cannot be used to simulate different wind effects.

Finally, nine papers present studies related to sensory effect simulation for medical or firefighter training and education [5, 14, 17, 18, 24, 32, 52, 53, 58]. Webb et al. [52], for example, perform an experiment with children using gloves to simulate tactile effects together with a VR environment for teaching cell biology. Despite positive feedback, the study presented no statistical evidence for improved learning performance.

2.2 Sensory Effect Propagation Simulation

Considering the studies identified as related to the one presented in this article, eight papers [19, 26, 28–31, 36, 39] have devoted considerable time and effort on using CFD simulations together with a scent emitter to calculate the airflow in an environment and control the scent dispersion.

Ishida et al. [19] simulate the variable airflow in a virtual environment with CFD to control the scent diffusion. Matsukura and Ishida [28] describe a set of best practices to obtain a more realistic experience of a scent sensory effect, given the results of experiments they conducted. One of the issues considered in the article indicates that the perception of smell is different for each person. Therefore, the intensity of a sensory effect must be adapted given the person's olfactory sensitivity. Another important issue is to consider the presence of the human body in the simulation, the convection caused by the body heat inducts significant changes in the scent concentration. Besides, Matsukura et al. [29] present the use of the scent emitter to deliver odor effects in sync with video playback. The software used to calculate the fluid simulation in the above studies is *CFD2000*.[1] The effect data generated with the simulation in *CFD2000* is stored for later use with the application. Furthermore, Matsukura et al. [30] present the device to be used together with the simulation data to deliver the intended effect.

In a further advance, Nakamoto et al. [36] developed a VR application where the user has to search for a scent source in the virtual environment. The virtual environment consists of a building with two floors, four rooms on each floor, and a scent source placed in one of those rooms. The user, equipped with a Head-Mounted Display and a scent-emitting device, walks around the virtual

[1] http://www.adaptive-research.com/.

environment to search for the scent source. The CFD simulation calculates the scent diffusion in the building, and its data determine the scent intensity to be emitted by the device. The authors conclude that using the proposed method decreased the time taken for users to find the scent source. Distinct from previous studies, the software used for the CFD simulation is *ANSYS18.1*[2] in the supercomputer *TSUBAME3.0*.

Qian et al. [39] intend to simulate a fire in a subway station to study smoke dispersion and evacuation practices. The authors performed real experiments in an English subway station by dispersing sulfur hexafluoride (SF_6) and then measuring its concentration at different points in the station. The simulation is built according to that station and uses CFD to determine the smoke behavior. Although data collection was performed only once, the authors extrapolate the simulation data to consider different climate conditions. For the evacuation study, the authors combine the CFD software with a pedestrian simulator. The result is a tool and methodology for firefighter training, construction planning, and evacuation routines.

Miao et al. [31] propose a CFD model to analyze pollutant dispersion in urban areas. The authors built a 3D map of a city on top of which they ran the CFD simulation. CFD simulation was performed using OpenFOAM.[3] A similar study is presented by Li et al. [26], whereby CFD techniques are used to study pollutant dispersion in environments with and without trees. The simulation is performed using Ansys Fluent.[4] The main results presented by Li et al. are that trees reduce the concentration of CO_2 and help diminish wind velocity in urban areas.

Related work presented in the previous paragraphs differs from our proposal in relation to the purpose of the simulation and the use of CFD. The presented studies build CFD models to perform a specific task, while the model proposed in this work is intended for general use. Moreover, sensory effect simulation is performed before application execution. Simulation data are stored for later use in the application execution. That approach presents some limitations when there are changes in the virtual environment during the application execution, and those changes affect the sensory effect simulation. It would be interesting if the simulation happened in real time to account for those changes. However, fluid simulation is a complex task that demands a lot of computational processing [36]. Yet, some fluid simulation methods make it possible to run the simulations in real-time. Although such methods slightly compromise the fidelity of the physics involved in fluid behavior (for example, turbulence is not reproduced), they produce results that represent fundamental characteristics of the phenomenon, such as dispersion and advection [48], which is sufficient to control Mulsemedia application actuators.

3 Sensory Effect Simulation

This section presents in detail the proposed system for simulating fluid-like sensory effects called *FluidSense*. Its proposed architecture, first presented by Rodrigues et al. [42], is depicted in Figure 1. With *FluidSense* we aim to provide support for the creation of Mulsemedia applications such as the treasure hunting one presented in Section 1.

The architecture presented in Figure 1 has two inputs: the simulation specification and user interaction or sensor data. The simulation specification allows the user to specify the virtual environment on top of which the simulation is executed, including the actuators and sensors in that environment. User interaction and/or sensor data can also be used as input by the simulation architecture for interactive (Human-in-the-loop [44]) simulations.

[2] https://www.ansys.com/.
[3] https://www.openfoam.com.
[4] https://www.ansys.com/products/fluids/ansys-fluent.

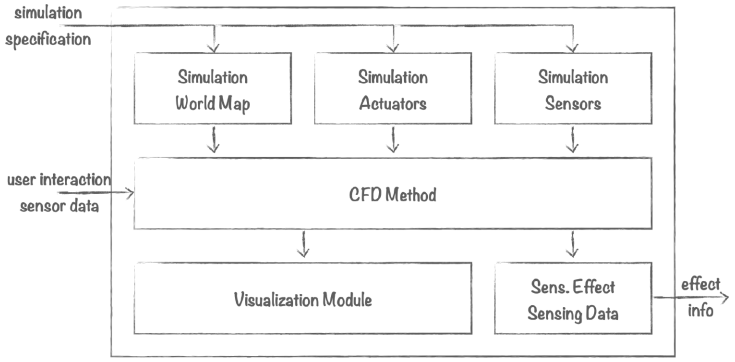


Fig. 1. FluidSense simulator architecture.

The CFD method component uses an implementation of **Smoothed-Particle Hydrodynamics (SPH)** that enables one to simulate gaseous-like sensory effects. More details about the CFD method are presented in Section 3.1.

As the simulation unfolds, the visualization module renders the virtual environment and the particles present in the simulation. It was developed in *Rust* using the graphics rendering library *wgpu*. The module gets the world map provided by the user to create a 3D environment and places a camera in the user's initial position. The simulation user controls the camera while the simulation unfolds in a first-person fashion. The simulation may also be executed in *headless* mode. In such a case, the visualization module is not executed. More details about the simulation specification are presented in Section 3.2.

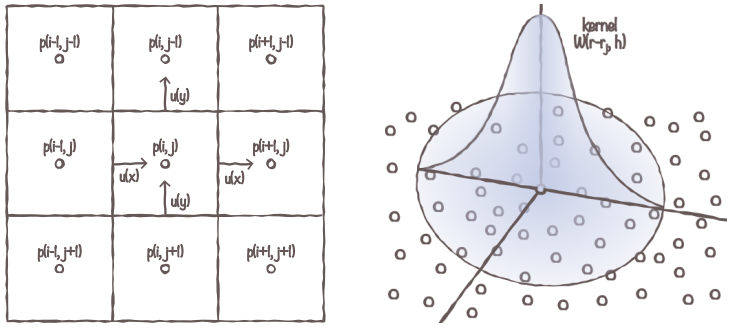
Particle data collected by sensors are treated by the Sensory Effect Sensing Data module, which writes the data in one or more CSV files so they can be used by third-party applications. Section 3.3 describes the simulation output.

3.1 CFD Method Background

Two common methods for CFD are mesh- and particle-based methods [27]. In a nutshell, mesh-based methods represent the fluid surface by a mesh. Each cell (i, j) of the mesh contains a vector field of the forces that are applied to the fluid inside the area of a given cell. Figure 2(a) shows a 2D view of the mesh, where p is the pressure and u is the velocity. In contrast, particle-based methods represent the fluid by a set of particles. Each particle in the system stores data about a given area of the system, i.e., the forces applied to it and the forces that cause the interaction with other particles within a given range represented by a kernel function. Figure 2(b) presents an arrangement of the particles in space and a representation of the kernel function.

Both mesh- and particle-based methods have their advantages and disadvantages according to different scenarios [15]. In general, mesh-based methods are more computationally expensive because the mesh must encompass the whole simulation domain in order to calculate the fluid advection. Particle-based methods, on the other hand, tend to be more efficient because it is fixed on the fluid, not in space. However, mesh-based methods produce more realistic simulations than particle-based methods. Furthermore, the precision and stability of particle methods are dependent on the particle density in the area.

For the simulation of sensory effects, we are not interested in simulating the whole domain, e.g., the whole room, but the places where the user may occupy. Furthermore, given that the available actuators for sensory effects are usually experimental hardware that does not provide much control in sensory effect rendering, in opposition to audiovisual content, a high-fidelity simulation is



(a) Bidimensional mesh representation

(b) Particle representation

Fig. 2. CFD methods. Based on the work by Giorgi [13].

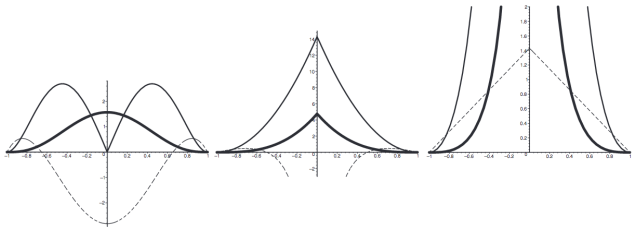


Fig. 3. Kernel functions W_{poly6}, W_{spiky}, and $W_{viscosity}$ (from left to right). Thick lines show the kernels, the thin lines show their gradients in the direction toward the center, and the dashed lines show the Laplacian. The figure is taken from [33].

not required. Therefore, given the tradeoff between processing time and simulation fidelity, it is preferred to produce real-time interactive simulations that present results that guarantee the fundamental physical characteristics of dispersion and advection to allow controlling actuators. Thus, a particle-based method was chosen for sensory effect simulation in our approach.

The choice of a particle-based method, in addition to the above reasons, enables us to combine different effects in the same environment, just by including more information about them in the particles. The simulator is intended to combine effects that behave more like a gas, such as fog, scent, and wind.

3.1.1 Particle-Based Method. Müller et al. [33] present a particle-based method for interactive fluid simulation and rendering. The physical method is based on SPH and uses special-purpose kernels to increase stability and speed. For each force field defined by the Navier–Stokes equations, i.e., external forces, pressure, and viscosity, there is a kernel function. The kernel functions are defined in terms of the particle position r and its vicinity radius h. Figure 3 shows the visualizations of the three smoothing kernels W_{poly6}, W_{spiky} and $W_{viscosity}$ used in the simulations of Müller et al. [33]. The thick lines show the kernels, the thin lines show their gradients in the direction toward the center, and the dashed lines show the Laplacian.

The Navier–Stokes equation of motion defines the acceleration of the fluid given by the pressure, gravity, and viscosity forces applied to the fluid. In SPH, the values of these forces are interpolated at a particle location (r) considering the weighted sum of contributions from all particles with mass

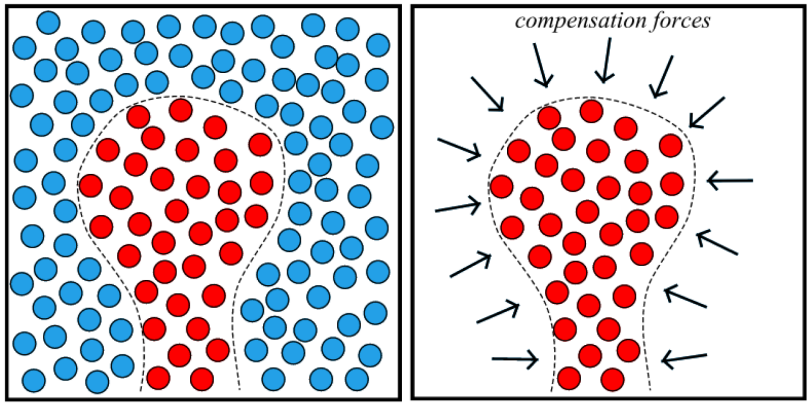


Fig. 4. (a) The standard approach fills the simulation domain with ambient air particles (blue). (b) Ren et al. [40] approach removes these ambient air particles and instead adds compensation forces to the gas particles (red). The figure is taken from Ren et al. [40].

m. This results in Equations (1)–(3) defined by Müller et al. [33]:

$$\rho_i = \sum_j m_j W_{poly6}(r_i - r_j, h), \tag{1}$$

$$f_i^{pressure} = -\sum_j m_j \frac{p_i + p_j}{2\rho_j} \nabla W_{spiky}(r_i - r_j, h), \tag{2}$$

$$f_i^{viscosity} = \mu \sum_j m_j \frac{v_j - v_i}{\rho_j} \nabla^2 W_{viscosity}(r_i - r_j, h). \tag{3}$$

Equation (1) is used to calculate the density ρ of the particle that is used to compute the external forces. Equation (2) is used to calculate the pressure at the particle's location, given that $p = k(\rho - \rho_0)$ where k is a gas constant and ρ_0 is the rest density. Equation (3) is used to calculate the viscosity at the particle location, given the fluid viscosity μ and velocity v.

The method proposed by Müller et al. [33] is more appropriate for the simulation of non-compressible fluids, e.g., water or other liquids. When it comes to simulating sensory effects, however, some effects behave akin to gaseous fluids, e.g., smell, fog, and wind. In such cases, a gas-specific approach is more suitable.

3.1.2 Gaseous Fluid Simulation. The standard approach to simulating gaseous fluids using SPH involves filling the environment with air particles. These air particles would have different parameters (mass, viscosity, rest pressure, etc.) than the particles representing the fluid of interest. During simulation, the air particles apply their forces to the particles of the gaseous fluid of interest. The problem with that approach is that the addition of extra particles makes the simulation more expensive to compute. In this context, Ren et al. [40] follow an approach in which compensation forces are used, so there is no need for air particles. That approach is presented in Figure 4.

In order to account for the compensation forces, an extra term is included in Equations (1)–(3). That term considers a virtual particle k added to the weighted sum of contributions from all particles,

such that:

$$r_i - r_k = \delta h, \tag{4}$$

$$\rho_k = \rho_i, \tag{5}$$

$$m_k = V_0 \bar{\rho}_i, \tag{6}$$

$$V_0 = \frac{| - \sum \frac{m_j}{\bar{\rho}_j} \nabla W_{ij} |}{| \nabla W_{ik} |}. \tag{7}$$

Equation (4) defines the distance from the virtual particle k to its paired particle i. It is given by a parameter δ that Ren et al. [40] heuristically set as $\delta \in [0.4, 0.7]$. Equation (5) defines the density of the virtual particle to be the same as the density of its paired particle i. Equation (6) defines the mass of the virtual particle given the predetermined value V_0 (Equation (7)) and the uncorrected density $\bar{\rho}_i$ of its paired particle i from Equation (1). Equation (8) corrects the density of the particles in the absence of surrounding ambient air particles:

$$\rho_i = \bar{\rho}_i (1 + V_0 W_{ik}). \tag{8}$$

Using the virtual particles mentioned previously, an extra term is added to Equation (2) to compensate for the local force balance, resulting in Equation (9) to calculate the pressure force:

$$\tilde{F}_{p,i} = -\sum_j m_j \frac{p_i + p_j}{2\rho_j} \nabla W(r_i - r_j, h) + m_k \frac{p_i + p_k}{2\rho_k} \nabla W(r_i - r_k, h). \tag{9}$$

The atmospheric pressure pushes the gas toward its inner direction. Equation (10) defines the atmospheric pressure force, where P_a is a user-defined strength coefficient to reflect ambient air pressure:

$$\tilde{F}'_{p,i} = P_a \sum \frac{m_j}{\rho_j} \nabla W_{ij}. \tag{10}$$

To account for the effect of local gas viscosity, Equation (11) defines a damping effect proportional to the particle velocity, where C_d is the damping coefficient and v_i is the velocity of the particle i. The damping effect is applied only in the cases where the norm of the particle exceeds a given threshold value, i.e., $|n_i| > C_N$:

$$\tilde{a}_{damp,i} = -C_d v_i. \tag{11}$$

A buoyancy force is also introduced by this method. The buoyancy is related to the temperature of the gas particles. According to Ren et al. [40], it is represented by $C_b T_i b$, where C_b is the buoyancy coefficient, T_i is the temperature of the particle i, and b is a unit vector pointing to the buoyancy direction.

To calculate the acceleration of each particle, the atmospheric pressure defined by Equation (10) is added to the local pressure defined by Equation (2) to form the pressure term; the damping and buoyancy accelerations are also included. Equation (12) defines the complete equation of this method:

$$a_i = \frac{1}{\rho_i}(\tilde{F}_{p,i} + \tilde{F}'_{p,i}) + \frac{1}{\rho_i} F_{v,i} + g + \tilde{a}_{damp,i} + C_b T_i b. \tag{12}$$

Both methods presented in [33, 40] may be used to simulate different types of fluids. In Mulsemedia applications, there can be sensory effects that behave more like liquids (e.g., water spray) and others that behave more like gaseous fluids (e.g., wind, fog, and scent). Therefore, both proposals can be used according to the sensory effects that are going to be used in each application.

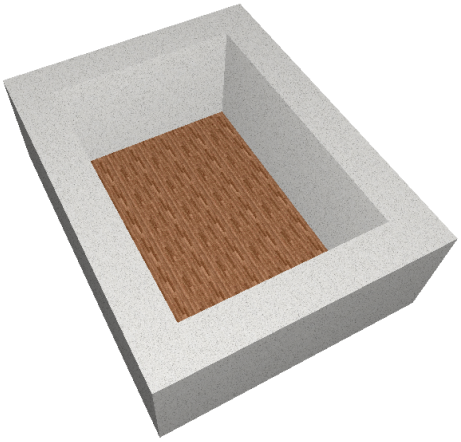


Fig. 5. Virtual environment relative to specification in Listing 1.

3.2 Simulation Specification

The simulation configuration is a YAML file containing sections for the environment, devices (actuators and sensors), and global simulation parameters.

3.2.1 Environment Section. This section contains a blueprint of the virtual environment delimiting walls, the position of the user, actuators, and sensors. The user draws the blueprint using ASCII characters, where each character represents a $1m \times 1m$ block. Listing 1 presents an environment definition example.

```
#########
# . . . . . a#
# . . . . . . . #
#.@. c . . . #
# . . . . . . . #
# . . . . . b#
#########
```

Listing 1. An example of environment configuration.

Chars "#", ". ", and "@" are reserved to represent walls, the floor, and the user, respectively. The user is free to use any other characters to mark regions of interest to place actuators and/or sensors. Listing 1 defines a $7m \times 9m$ room, with the user's initial position at the center-left of the room. In the right corners, there are regions of interest labeled "a" and "b." Next to the user, there is a region of interest labeled "c." Figure 5 depicts the resulting virtual environment.

The room's inner size is $5m \times 7m$, surrounded by 1 m-thick walls, making the total size of the room $7m \times 9m$. Overall, the image provides a clear and detailed representation of the layout of the room, making it easy to visualize how the space might look in reality.

It is worth noticing that any character shall be used for labels: symbols, numbers, letters with small, upper caps, and even accents. Moreover, a character representing a region of interest may be placed in more than one position, such that the same settings shall be used in each position. Finally, walls and the floor restrict the simulation region. It is assumed there is a floor below the user and every region of interest. A ceiling is placed at $3m$ high. In our proposal, particles emitted

by actuators are absorbed by boundaries (walls, floor, and ceiling) or any obstacle present in the environment.

3.2.2 Device Section. This section provides configuration for actuators and sensors, associating the given labels present in the blueprint with each device configuration. It is separated into two parts: actuators and sensors.

The actuators have a *height* attribute to complete its 3D position in the virtual environment. The blueprint gives the x and z axes, and the attribute *height* gives the y-axis. The *direction* attribute provides a unit vector pointing to the direction in which particles are emitted by the actuator. Particles are emitted with a velocity defined in attribute *initial_velocity* and at an interval defined in attribute *interval*. An additional attribute *range* provides a distance from the actuator position for the three axes. That distance is used in a jitter function to place new particles as they are emitted by the actuator. This avoids creating instability in the simulation by placing two particles in the same position.

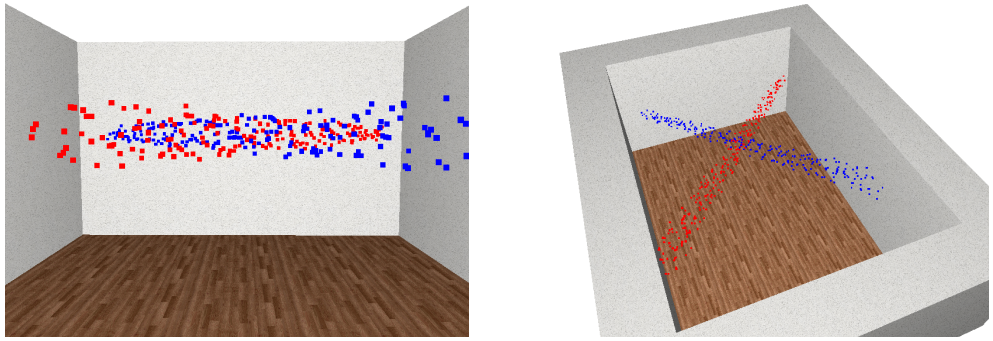
```
actuators:
  a:
    height: 1.5
    direction: [-1.0, 0.0, 1.0]
    initial_velocity: 5.0
    temperature: 25.0
    range: [0.2, 0.2, 0.2]
    interval: 0.05
    particle:
      size: 0.02
      color: [0.0, 0.0, 1.0]
  b:
    height: 1.5
    direction: [-1.0, 0.0, -1.0]
    initial_velocity: 5.0
    range: [0.2, 0.2, 0.2]
    interval: 0.05
    temperature: 25.0
    particle:
      size: 0.02
      color: [1.0, 0.0, 0.0]
```

Listing 2. Actuators definition.

The example presented in Listing 2 defines two actuators that share some of their attributes. Both actuators are positioned at a height of $1.5m$. They emit particles at each $0.05s$ with a velocity of $5.0m/s$ and a temperature of $25°C$. The particles represent a gaseous fluid, and particle creation uses a $0.2m$ maximum jitter in all three axes.

What differs between the two actuators is their particle color and the emission direction; actuator "a" emits blue particles and actuator "b" emits red particles. They emit particles in an opposite diagonal direction such that particles emitted by them will interact with each other at some point in the center of the environment.

Figure 6(a) shows the user view at a given moment of the simulation, in which it is possible to visualize the particles from both actuators passing through the convergence point in the center of the environment. Figure 6(b) shows an overhead view of the simulation, in which it is possible to observe the actuators emitting the particles in a diagonal trajectory toward the center of the simulated environment.



(a) Front user view (b) Over head view

Fig. 6. Particles emitted from the actuators.

Sensors capture data of all particles inside a given volume whose center position is given by the blueprint and the *height* attribute. The *range* attribute defines its volume. Finally, attribute *output* indicates the CSV file where to store the particle data captured by the sensor.

```
sensors:
  c:
    height: 1.0
    range: [1.0, 1.0, 1.0]
    output: "sensdata.csv"
```

Listing 3. Sensor definition.

The example presented in Listing 3 defines a sensor that collects data in a $1m \times 1m \times 1m$ volume centered at $[4.5m, 3.5m, 1m]$. The collected data is saved in file *sensdata.csv*.

At each iteration of the simulation, the sensor probes its perimeter to identify the presence of any particles. If any particle is detected, the particle data are inspected in order to extract precise information regarding the specific moment of the simulation. After obtaining the relevant particle data, conversion to CSV format takes place, which is then recorded in the output file previously defined in the settings. If no file has been specified, the same information is presented in the application's *log* record.

```
Label, TET, ETSE, Position, Velocity, Acceleration, Forces, Density, Temp
c, 5.50, 5.37, [5.93, 1.73, 3.00], [-4.96, 0.05, 4.84], [0.00, 5.38,
    0.00], [0.00, 88336.60, 0.00], 16414.67, 19.00
c, 5.52, 5.38, [5.92, 1.73, 3.01], [-4.96, 0.05, 4.84], [0.00, 5.38,
    0.00], [0.00, 88336.60, 0.00], 16414.67, 19.00
```

Listing 4. Output example.

Listing 4 presents three lines of the output file of the simulation presented above, relative to the sensor represented by symbol "c" in the example. Each line of the file contains the main information of the particle at that instant of the simulation, such as the label of the sensor that detected the particle, the total elapsed time of the simulation, the **Elapsed Time Since the Emission (ETSE)** of the particle, its position, velocity, acceleration, sum total of the acting forces, density, and temperature.

Table 2. General Simulation Parameters

Parameter	Definition
Step	Simulation step on seconds
Radius	Particle radius in meters used for computing the neighbors of a particle
Mass	Particle mass in grams used for computing the fluid density at the particle position
Viscosity	Fluid viscosity constant
Rest density	Fluid density at rest
Gravity force	Vector representing the gravity force
Gas constant	Gas constant that depends on the temperature
δ	Distance of the virtual particle in terms of its radius
Virtual particle direction	Direction of the virtual particle to the current particle, used to calculate the compensation forces for gaseous fluids
Damping coefficient	Rate of the damping effect
Damping limit	Limit to trigger the damping effect
Thermal conductivity	Rate the particles transfer heat
Radiation half-life	Rate a particle loses heat with at great speeds
Buoyancy direction	User defined direction of buoyancy
Buoyancy coefficient	User defined coefficient for buoyancy

3.2.3 Global Simulation Parameters. The general simulation parameters are defined in the last section of the file. The parameters are used throughout the simulation in order to calculate the simulation model and are applied to every particle. The list of general parameters is presented in Table 2.

3.3 Simulation Output

Given a configuration file as input, the simulation generates a dynamic and interactive environment as presented in Figures 5 and 6. Recalling the example presented in Section 3.2, it defines the blueprint of a $7m \times 9m$ room with a sensor in the center left, two actuators in the right corners, and the device configuration.

By using the CFD method component, we calculate the propagation of the gas particles emitted by the actuators. As particles move, they disperse, interact with the environment, and with each other at a specific point where they meet. The visualization module then renders at the screen particles allowing the users to explore the simulation in real time (Figure 6). Using the keyboard and mouse the user can walk through the simulation to analyze how the simulation behaves from different positions and angles.

It is also possible to gain insights about the simulation by analyzing the data captured by the sensors that are logged during the execution (Listing 4). It is possible to observe how these interactions affect particle data, for example, temperature exchange, both with the environment and with the interactions between themselves, causing a variation of the initial temperature of the emission of the particles.

Based on the simulation data (that is partially presented in Listing 4), we can draw some conclusions about the behavior of the simulation. A user located at the sensor position would take around $5s$ to perceive the wind sensory effect, given the ETSE of the particle. Therefore, for the perception of the sensory effect to be synchronized with the multimedia content, this delay could

(i) be taken into account during the development of the application or (ii) be used by a Mulsemedia presentation engine to adapt the effect start time. The density values of the sensory effect can be used as references to calculate the intensity that a user would perceive a sensory effect. It can be noted that there was a loss of temperature to the environment and with the interaction between the particles, which is different from the temperature of emission of the particles; therefore, this parameter can also be adjusted so that the user perceives a sensory effect with the temperature desired by the application authors. All the above conclusions would be valuable when constructing our treasure hunting Mulsemedia application.

4 Simulation Architecture Evaluation

This section presents an evaluation conducted with the simulation architecture presented in Section 3. The following sections present how the model parameters were tuned and data collected about its efficiency. The code used for parameter tuning and model evaluation is available at GitHub.[5]

4.1 Empirical Data Acquisition

To support simulation parameter tuning and validate the proposed architecture, a set of controlled physical experiments was conducted. The objective was to gather empirical data related to temperature and wind propagation under conditions analogous to those represented in the virtual environment.

At the first stage, data were collected to characterize the heat wind source. To that end, a DHT-11 temperature sensor and an anemometer were positioned directly in front of a heat-emitting device—a hairdryer controlled by an Arduino microcontroller. Temperature was sampled for 60*s* to observe thermal variation over time. Wind velocity data were obtained by recording the anemometer with a camera, extracting frames at 1-second intervals, and manually analyzing readings to construct a velocity curve.

At the second stage, data were collected to characterize the wind temperature dispersion. A 3×5 grid of 15 DHT-11 sensors was used. In the grid, each sensor is spaced 20 cm apart and is used to capture the spatial dispersion of the heat generated by the heat wind source. The heat source was placed at distances of 30 cm, 60 cm, 90 cm, and 120 cm from the sensor grid. For each distance, 10 repetitions of 60 s recordings were conducted, totaling 600 measurements ($15 \times 4 \times 10$). Between runs, air conditioning was used to restore the environment to baseline temperature. The Arduino managed both heat source activation and synchronized sensor data acquisition.

As part of the data exploration phase, baseline tests were also conducted with the heat source turned off to verify sensor variability. While most sensors showed consistent readings, slight discrepancies (up to $1°C$) were observed, likely due to handling, human proximity, and intrinsic sensor variation. Under active heat conditions, the data exhibited patterns consistent with forced convection: sensors aligned with the airflow registered increased temperatures near the source, while at greater distances, adjacent and upper sensors began to register rising temperatures due to heat dispersion. Figure 7 depicts this behavior in the collected data. These findings confirmed the expected thermal dynamics and provided reliable reference patterns for validating and refining the virtual simulation.

4.2 Model Optimization

The collected datasets served as empirical ground-truth to evaluate simulation realism and adjust the simulation parameters presented in Table 2 (e.g., particle radius, emission velocity, and thermal

[5] https://github.com/multisens/fluid-sense.

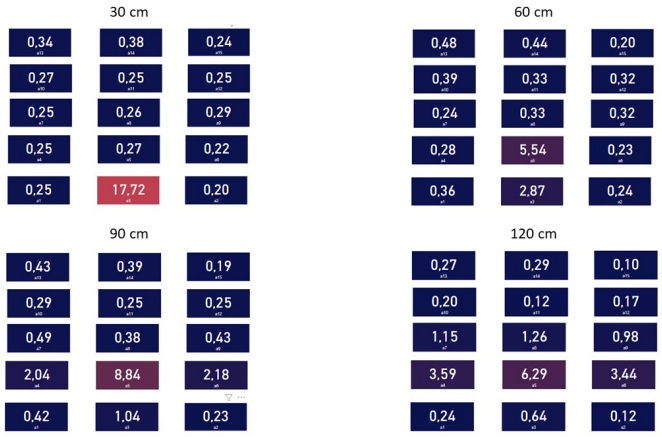


Fig. 7. Graphical visualization of grid-sensor temperature at different distances.

properties). To do so, the 600 measurements were used to create a time series with the temperature at the 60 positions (15 sensors at 4 distances) along 60 s of measurement. This was done by concatenating each 60 s individual time series into a single time series for the whole scenario.

Then a simulation was built that mimics the same conditions. It included a sensor positioned in a way that allows us to collect particle data at the same 60 positions and generate a time series with temperature data for the 60 positions over 60 s. The construction of this time series matches the one for the data captured from sensors.

Model parameter optimization was then conducted using a genetic algorithm to choose the simulation parameter values. At each step, the Kendall correlation between the ground-truth time series and the one obtained from the simulation was calculated to evaluate the simulation accuracy. The parameters were then optimized to search for greater correlation between the ground-truth time series and the one obtained from the simulation.

The genetic algorithm used 1,000 generations, with a total population of 10, 90% mutation rate, 1.5 mutation strength, and arithmetic crossover. The algorithm was used to optimize the following simulation parameters: buoyancy coefficient, damping coefficient, damping threshold, gas constant, mass, radiation half-life, radius, rest density, small positive, step, thermal conductivity, and viscosity. After 5 hours of execution, a set of optimal parameters for the simulation was found, with a Kendall correlation value of 0.6005, a Pearson correlation of 0.9969, and a cumulative mean **Dynamic Time Warping (DTW)** distance of 66.8162. The results indicate that the simulation is able to reproduce the general behavior of the experiment with the sensors, as evidenced by the Pearson correlation; however, its Kendall correlation shows possible inconsistencies in the particle records. Furthermore, the relatively high DTW distance results show a possible temporal misalignment between the observations of the experiment and the simulation.

4.3 Model Performance

Finally, after defining the simulation parameters using the optimization algorithm, an evaluation was performed with the simulation architecture. The model performance evaluation scenario was similar to the one used for parameter optimization, with an emitter pointed at a group of sensors in a room, mimicking the previous experiment. The main differences in this scenario were the increased simulation time and the number of emitters, which were varied to evaluate simulation

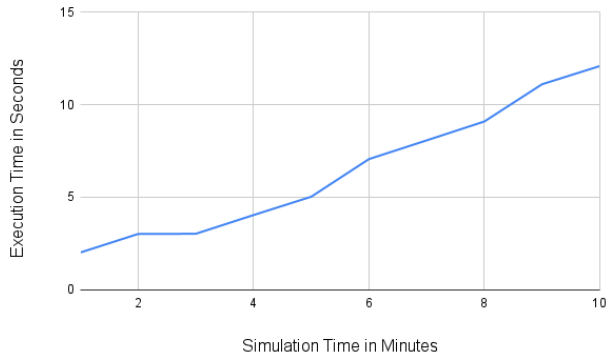


Fig. 8. Graphical visualization of execution time of the simulation by the time defined for the simulation.

Table 3. Simulation Performance with Different Numbers of Wind Sources

Number of sources	Execution Time	Memory Usage	CPU Usage	Number of Particles
1	03.03 s	12.79 MB	4.60%	1,017
2	07.03 s	12.85 MB	4.60%	2,015
3	09.08 s	12.66 MB	4.70%	2,999
4	18.09 s	12.61 MB	4.70%	4,074
5	35.27 s	12.84 MB	6.10%	5,018

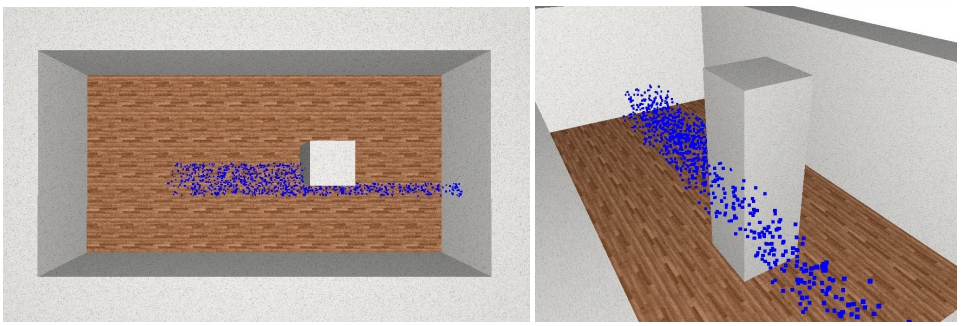


Fig. 9. Virtual environment of the second simulation scenario.

performance with different configurations. The evaluation was conducted on a commercial PC, i7 processor 10th generation, 32 GB RAM, NVIDIA GeForce GTX 1650, with Windows 11.

When evaluating the simulation time in relation to the execution time, we can observe a linear trend, with the execution time increasing as the configured simulation time increases, as shown in Figure 8. Using the **Real-Time Factor (RTF)** metric ($RTF = simulation\ time/execution\ time$) considering the simulation and executed times presented in Figure 8, it results on a RTF value of ≈50 across simulations.

The performance of the simulation was also assessed with one or more wind source points, analyzing CPU and memory usage, as well as verifying whether there is interference in the evaluation results between simulation and experiment. In Table 3, we can see an increase in execution time as more particle emitters are added, while CPU and RAM usage remain consistent.

4.4 Simulation Scenarios

The second simulation scenario was conducted to present the model reuse across different configurations. It models a $7m \times 12m$ room with a column in front of the actuator as depicted in Figure 9.

The scenario uses the same parameter values as the one presented earlier. As one can observe, the result is that part of the particles are absorbed by the obstacle (column). Consequently, less heat from the wind arrives at the far side of the room.

5 Limitations

This work presents a physically based simulation framework for sensory effect propagation. However, some limitations must be acknowledged.

First, the evaluation focuses on physical consistency rather than perceptual validation. While the results indicate that the model captures key physical trends, user studies are required to assess perceptual coherence and QoE.

Second, the model parameters were calibrated using the same dataset employed in evaluation. Although the goal was parameter calibration rather than predictive modeling, future work should include independent validation datasets to assess generalization.

Third, the current evaluation focuses on gaseous-like effects in controlled environments. More complex scenarios involving diverse geometries and environmental conditions should be explored.

Finally, comparisons with high-fidelity CFD solvers were not conducted due to computational constraints. They could also consider simpler heuristic models as a baseline approach for sensory effect dispersion simulation. Such comparisons represent an important direction for future validation whether the SPH approach yields a sufficient accuracy in relation to high-fidelity solvers and an accuracy gain in complex scenarios that justify its computational cost over simpler methods.

6 Conclusion

Sensory effects can be integrated in multimedia applications to enhance QoE and user immersion. However, achieving smooth integration requires a deep understanding of the challenges involved in synchronizing the rendering of sensory effects in the physical environment with multimedia presentation, particularly in the context of dynamic user interactions and environmental variability.

This study addressed the complexity of simulating fluid-like sensory effects within multimedia systems by proposing an SPH simulation architecture. Although traditional CFD approaches offer high physical accuracy, their computational cost limits real-time interactivity and flexibility. Our solution prioritizes responsiveness and flexibility of configuration, favoring fast, interactive simulations suitable for authoring and exploratory use.

Despite its architectural and computational contributions, this work still presents limitations that future research needs to address. One, and notably so, is collecting more data for empirical validation through comparison with real-world measurements, crucial for grounding simulations in physical reality. Moreover, no user studies or perceptual evaluations were conducted to assess how users interpret or perceive the simulated effects—an essential step when aiming to improve immersive experiences and QoE. Again, this should be the focus of future endeavors.

Another challenge lies in the imprecision of hardware actuators responsible for delivering sensory effects. These devices often exhibit limited resolution and control accuracy, introducing discrepancies between simulated and perceived stimuli. Addressing this gap requires not only actuator calibration but also adaptive algorithms that consider hardware limitations during simulation and effect rendering.

To mitigate some of these challenges, our simulator provides an authoring environment in which developers can define virtual scenes, configure fluid parameters, and observe the dynamic behavior of sensory effects. Sensor feedback collected during simulations enables performance analysis, optimization, and iterative refinement of parameters.

As a proof of concept, we presented a wind simulation scenario based on parameters tuned according to real-world data. Based on that, we presented an evaluation of the time spent to perform the simulation, the memory, and CPU usage. Planned future work includes collecting more real-world fluid data and incorporating perceptual metrics and user testing to assess and optimize the quality of the immersive experience from both technical and experiential perspectives.

Ultimately, we envision this tool as a foundation for content creators and application developers seeking to integrate rich sensory modalities into multimedia systems. By facilitating iterative development, fostering experimentation, and enabling real-time exploration, we aim to contribute meaningfully to the evolution of Mulsemedia applications. Through continued research and refinement, we hope to unlock more compelling and realistic Mulsemedia experiences.

References

[1] O. A. Ademoye and G. Ghinea. 2013. Information recall task impact in olfaction-enhanced multimedia. *ACM Transactions on Multimedia Computing, Communications, and Applications* 9, 3 (2013), 1–16.

[2] A. A. Alkasasbeh, F. Spyridonis, and G. Ghinea. 2021. When scents help me remember my password. *ACM Transactions on Applied Perception* 18, 3 (2021), 1–18.

[3] J. G. Apostolopoulos, P. A. Chou, B. Culbertson, T. Kalker, M. D. Trott, and S. Wee. 2012. The road to immersive communication. *Proceedings of the IEEE* 100, 4 (2012), 974–990. DOI: https://doi.org/10.1109/JPROC.2011.2182069

[4] H. Cao, S. Tao, F. Xu, R. Coveney Jr., J. Cao, B. Li, W. Liu, X. Wang, J. Hu, W. Shen, et al. 2004. Multimedia fate model for hexachlorocyclohexane in Tianjin, China. *Environmental Science and Technology* 38, 7 (2004), 2126–2132. DOI: https://doi.org/10.1021/es0305860

[5] M. Castrillón, P. Jorge, I. López, A. Macías, D. Martín, R. Nebot, I. Sabbagh, F. Quintana, J. Sánchez, A. Sánchez, et al. 2011. Forecasting and visualization of wildfires in a 3D geographical information system. *Computers and Geosciences* 37, 3 (2011), 390–396. DOI: https://doi.org/10.1016/j.cageo.2010.04.011

[6] J. Chen, J. Xu, Y. Qi, J. Tong, and Y. Wu. 2008. Simulation of waste gas pollution diffusion for urban planning: Take Nanjing as an example. In *Proceedings of the SPIE—The International Society for Optical Engineering*, Vol. 7143. DOI: https://doi.org/10.1117/12.812579

[7] S. Chen and S. Dayan. 2013. Integration of tree models into a real-time pollution dispersion simulation. *Applied Mechanics and Materials* 385–386 (2013), 378–381. DOI: https://doi.org/10.4028/www.scientific.net/AMM.385-386.378

[8] A. Covaci, R. Trestian, E. B. Saleme, I.-S. Comsa, G. Assres, C. A. Santos, and G. Ghinea. 2019. 360 Mulsemedia: A way to improve subjective QoE in 360 videos. In *Proceedings of the 27th ACM International Conference on Multimedia*, 2378–2386.

[9] M. de Paiva Guimarães, J. Martins, D. Dias, R. Guimarães, and B. Gnecco. 2022. An olfactory display for virtual reality glasses. *Multimedia Systems* 28 (2022), 1573–1583. DOI: https://doi.org/10.1007/s00530-022-00908-8

[10] M. Garcia-Ruiz, B. Kapralos, and G. Rebolledo-Mendez. 2021. Towards effective odor diffusion with fuzzy logic in an olfactory interface for a serious game. In *Proceedings of the Late Breaking Papers: Multimodality, eXtended Reality, and Artificial Intelligence (HCII '21)*. Lecture Notes in Computer Science, Vol. 13095, 3–16. DOI: https://doi.org/10.1007/978-3-030-90963-5_1

[11] A. Genovese, R. Labati, V. Piuri, and F. Scotti. 2011. Virtual environment for synthetic smoke clouds generation. In *Proceedings of the 2011 IEEE International Conference on Virtual Environments, Human-Computer Interfaces and Measurement Systems (VECIMS '11)*, 32–37. DOI: https://doi.org/10.1109/VECIMS.2011.6053841

[12] G. Ghinea, C. Timmerer, W. Lin, and S. Gulliver. 2014. Mulsemedia: State of the art, perspectives, and challenges. *ACM Transactions on Multimedia Computing, Communications, and Applications* 11 (2014), 1–23. DOI: https://doi.org/10.1145/2617994

[13] G. Giorgi. 2018. *Nonlinear Hydrodynamic Modelling of Wave Energy Converters under Controlled Conditions*. Ph.D. thesis. National University of Ireland Maynooth.

[14] T. Grunwald, T. Krummel, and R. Sherman. 2004. Advanced technologies in plastic surgery: How new innovations can improve our training and practice. *Plastic and Reconstructive Surgery* 114, 6 (2004), 1556–1567. DOI: https://doi.org/10.1097/01.PRS.0000138242.60324.1D

[15] M. L. Hosain and R. B. Fdhila. 2015. Literature review of accelerated CFD simulation methods towards online application. *Energy Procedia* 75 (2015), 3307–3314. DOI: https://doi.org/10.1016/j.egypro.2015.07.714

[16] Y.-Y. Hu, Y.-F. Jan, K.-W. Tseng, Y.-S. Tsai, H.-M. Sung, J.-Y. Lin, and Y.-P. Hung. 2021. ABio: Active bi-olfactory display using subwoofers for virtual reality. In *Proceedings of the 29th ACM International Conference on Multimedia*. ACM, New York, NY, 2065–2073. DOI: https://doi.org/10.1145/3474085.3475678

[17] D. Huang, P. Tang, X. Wang, T. Wan, and W. Tang. 2019. New haptic syringe device for virtual angiography training. *Computers and Graphics* 80 (2019), 97–103.

[18] H. Huang, L. Tang, J. Li, and C. Chen. 2012. Simulation and visualization of forest fire growth in an integrated 3D virtual geographical environment—A preliminary study. In *Proceedings of the 2012 20th International Conference on Geoinformatics, Geoinformatics*. DOI: https://doi.org/10.1109/Geoinformatics.2012.6270344

[19] H. Ishida, H. Matsukura, H. Yoshida, and T. Nakamoto. 2008. Application of computational fluid dynamics simulation to olfactory display. In *Proceedings of the 18th International Conference on Artificial Reality and Telexistence (ICAT)*, 285–288.

[20] M. Ivanov, M. F. Moreno, and D. C. Muchaluat-Saade. 2024. Automatic preparation of sensory effects: Managing synchronization in mulsemedia applications. In *Proceedings of the 15th ACM Multimedia Systems Conference (MMSys '24)*. ACM, New York, NY, 100–108. DOI: https://doi.org/10.1145/3625468.3647615

[21] L. Jalal and M. Murroni. 2017. Enhancing TV broadcasting services: A survey on mulsemedia quality of experience. In *Proceedings of the IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB)*. DOI: https://doi.org/10.1109/BMSB.2017.7986192

[22] M. Josué, R. Abreu, F. Barreto, D. Mattos, G. Amorim, J. dos Santos, and D. Muchaluat-Saade. 2018. Modeling sensory effects as first-class entities in multimedia applications. In *Proceedings of the 9th ACM Multimedia Systems Conference (MMSys '18)*. ACM, New York, NY, 225–236. DOI: https://doi.org/10.1145/3204949.3204967

[23] K. Kahol, P. Tripathi, T. Mcdaniel, L. Bratton, and S. Panchanathan. 2006. Modeling context in haptic perception, rendering, and visualization. *ACM Transactions on Multimedia Computing, Communications, and Applications* 2, 3 (2006), 219–240. DOI: https://doi.org/10.1145/1152149.1152153

[24] T. Knott, S. Ullrich, O. Grottke, T. Fries, M. Gerressen, T. Deserno, R. Rossaint, and T. Kuhlen. 2010. Medical training simulators for bilateral sagittal split osteotomy and regional aneasthesia in virtual environments. In *CEUR Workshop Proceedings*, Vol. 1475, 137–140.

[25] T. Lewis and K. Bhaganagar. 2022. Configurable simulation strategies for testing pollutant plume source localization algorithms using autonomous multisensor mobile robots. *International Journal of Advanced Robotic Systems* 19, 2 (2022). DOI: https://doi.org/10.1177/17298806221081325

[26] Q. Li, J. Liang, Q. Wang, Y. Chen, H. Yang, H. Ling, Z. Luo, and J. Hang. 2022. Numerical investigations of urban pollutant dispersion and building intake fraction with various 3D building configurations and tree plantings. *International Journal of Environmental Research and Public Health* 19, 6 (2022), 3524. DOI: https://doi.org/10.3390/ijerph19063524.

[27] D. Markauskas, H. Kruggel-Emden, R. Sivanesapillai, and H. Steeb. 2017. Comparative study on mesh-based and mesh-less coupled CFD-DEM methods to model particle-laden flow. *Powder Technology* 305 (2017), 78–88.

[28] H. Matsukura and H. Ishida. 2009. Olfactory display: Fluid dynamic considerations for realistic odor presentation. In *Proceedings of the 15th Joint Virtual Reality Eurographics Conference on Virtual Environments*, 61–64. DOI: https://doi.org/10.2312/EGVE/JVRC09/061-064

[29] H. Matsukura, H. Yoshida, H. Ishida, and T. Nakamoto. 2009. Interactive odor playback based on fluid dynamics simulation. In *Proceedings of the 2009 IEEE Virtual Reality Conference*, 255–256. DOI: https://doi.org/10.1109/VR.2009.4811042

[30] H. Matsukura, H. Yoshida, H. Ishida, A. Saitoh, and T. Nakamoto. 2009. Odor presentation with a vivid sense of reality: Incorporating fluid dynamics simulation into olfactory display. In *Proceedings of the 2009 IEEE Virtual Reality Conference*, 295–296. DOI: https://doi.org/10.1109/VR.2009.4811062

[31] H. Miao, V. Raghavan, H. Gopalan, C. Kang, and J. Lou. 2017. Developing three-dimensional CFD city model based on public accessible information for green mark building ventilation assessment. In *Proceedings of the 19th Australasian Fluid Mechanics Conference*.

[32] A. Moreno, J. Posada, A. Segura, A. Arbelaiz, and A. García-Alonso. 2014. Interactive fire spread simulations with extinguishment support for virtual reality training tools. *Fire Safety Journal* 64 (2014), 48–60. DOI: https://doi.org/10.1016/j.firesaf.2014.01.005

[33] M. Müller, D. Charypar, and M. Gross. 2003. Particle-based fluid simulation for interactive applications. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 154–159.

[34] N. Murray, O. A. Ademoye, G. Ghinea, and G.-M. Muntean. 2017. A tutorial for olfaction-based multisensorial media application design and evaluation. *ACM Computing Surveys* 50, 5 (2017), 1–30. DOI: https://doi.org/10.1145/3108243

[35] T. Nakamoto, H. Ishida, and H. Matsukura. 2011. Olfactory display using solenoid valves and fluid dynamics simulation. In *Multiple Sensorial Media Advances and Applications: New Developments in MulSeMedia*. IGI Global Scientific Publishing, 140–163. DOI: https://doi.org/10.4018/978-1-60960-821-7.ch007

[36] T. Nakamoto, T. Hirasawa, and Y. Hanyu. 2020. Virtual environment with smell using wearable olfactory display and computational fluid dynamics simulation. In *Proceedings of the 2020 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, 713–720. DOI: https://doi.org/10.1109/VR46266.2020.00094

[37] N. Platt, D. De Riggi, S. Warner, P. Bieringer, G. Bieberbach, A. Wyszogrodzki, and J. Weil. 2010. Method for comparison of large eddy simulation-generated wind fluctuations with short-range observations. In *Proceedings of the 13th International Conference on Harmonisation within Atmospheric Dispersion Modelling for Regulatory Purposes (HARMO '10)*.

[38] N. Platt, D. DeRiggi, S. Warner, P. Bieringer, G. Bieberbach, A. Wyszogrodzki, and J. Weil. 2012. Method for comparison of large eddy simulation generated wind fluctuations with short-range observations. *International Journal of Environment and Pollution* 48, 1–4 (2012), 22–30. DOI: https://doi.org/10.1504/IJEP.2012.049648

[39] Z. Qian, B. Agnew, and E. Thompson. 2014. Simulation of air flow, smoke dispersion and evacuation of the monument metro station based on subway climatology. In *Fusion—Proceedings of the 32nd International Conference on Education and Research in Computer Aided Architectural Design in Europe (eCAADe)*, Vol. 1, 119–128.

[40] B. Ren, X. Yan, T. Yang, C.-F. Li, M. C. Lin, and S.-M. Hu. 2016. Fast SPH simulation for gaseous fluids. *The Visual Computer* 32, 4 (2016), 523–534. DOI: https://doi.org/10.1007/s00371-015-1086-y

[41] E. Rodrigues, E. Saleme, and C. Santos. 2021. A haptic system for switching wind temperatures based on ultrasonic vibrations, peltier elements, and electrical resistances for multisensory applications. In *Proceedings of the Brazilian Symposium on Multimedia and the Web*, 73–80. DOI: https://doi.org/10.1145/3470482.3479638

[42] R. Rodrigues, J. R. da Silva Jr., D. Brandão, and J. dos Santos. 2022. An approach for sensory effects dispersion simulation with computational fluid dynamics. In *Proceedings of the Brazilian Symposium on Multimedia and the Web (WebMedia '22)*. ACM, New York, NY, 368–376. DOI: https://doi.org/10.1145/3539637.3556999

[43] R. O. Rodrigues, M. I. P. Josué, R. S. Abreu, G. F. Amorim, D. C. Muchaluat-Saade, and J. A. F. dos Santos. 2019. A proposal for supporting sensory effect rendering in Ginga-NCL. In *Proceedings of the 25th Brazilian Symposium on Multimedia and the Web (WebMedia '19)*. ACM, New York, NY, 273–280. DOI: https://doi.org/10.1145/3323503.3349544

[44] L. Rothrock and S. Narayanan (Eds.). 2011. *Human-in-the-Loop Simulations: Methods and Practice*. Springer-Verlag, London.

[45] E. B. Saleme and C. A. S. Santos. 2015. PlaySEM: A platform for rendering MulSeMedia compatible with MPEG-V. In *Proceedings of the 21st Brazilian Symposium on Multimedia and the Web*, 145–148. DOI: https://doi.org/10.1145/2820426.2820450

[46] E. B. Saleme, C. A. Santos, and G. Ghinea. 2019. A mulsemedia framework for delivering sensory effects to heterogeneous systems. *Multimedia Systems* 25, 4 (2019), 421–447.

[47] N. Shibano, P. Hareesh, M. Kashiwagi, K. Sawada, and H. Takemura. 2001. Development of VR experiencing system with hemi-spherical immersive projection display for urban environment design. In *Proceedings of the 7th International Conference on Virtual Systems and Multimedia (VSMM '01)*, 499–505. DOI: https://doi.org/10.1109/VSMM.2001.969705

[48] L. D. G. Sigalotti, J. Klapp, and M. G. Gesteira. 2021. The mathematics of smoothed particle hydrodynamics (SPH) consistency. *Frontiers in Applied Mathematics and Statistics* 7 (2021), 797455.

[49] B. Singh, E. Pardyjak, A. Norgren, and P. Willemsen. 2011. Accelerating urban fast response Lagrangian dispersion simulations using inexpensive graphics processor parallelism. *Environmental Modelling and Software* 26, 6 (2011), 739–750. DOI: https://doi.org/10.1016/j.envsoft.2010.12.011

[50] R. Tortell, D. Luigi, A. Dozois, S. Bouchard, J. F. Morie, and D. Ilan. 2007. The effects of scent and game play experience on memory of a virtual environment. *Virtual Reality* 11, 1 (2007), 61–68. DOI: https://doi.org/10.1007/s10055-006-0056-0

[51] J. Wang and W. Lu. 2010. A real-time haptics-based deformable model for virtual prototyping and simulations. In *Proceedings of the ASME International Design Engineering Technical Conferences and Computers and Information in Engineering Conference 2009 (DETC '09)*, Vol. 2, 1601–1610. DOI: https://doi.org/10.1115/DETC2009-87275

[52] M. Webb, M. Tracey, W. Harwin, O. Tokatli, F. Hwang, N. Barrett, C. Jones, and R. Johnson. 2019. An investigation of the impact of haptics for promoting understanding of difficult concepts in cell biology. In *IFIP Advances in Information and Communication Technology*, 524, 197–206. DOI: https://doi.org/10.1007/978-3-030-23513-0_20

[53] F. Wellmann, S. Virgo, D. Escallon, M. de la Varga, A. Jüstel, F. Wagner, J. Kowalski, H. Zhao, R. Fehling, and Q. Chen. 2022. Open AR-sandbox: A haptic interface for geoscience education and outreach. *Geosphere* 18, 2 (2022), 732–749. DOI: https://doi.org/10.1130/GES02455.1

[54] P. Wesseling. 2009. *Principles of Computational Fluid Dynamics*, Vol. 29. Springer Science and Business Media.

[55] Y.-Y. Xiong, Y.-B. Chen, Q.-L. Qian, and S.-S. Wang. 2013. 3D simulation of typical pollutants diffusion in chemical industrial park based on computer graphics techniques. *International Journal of Earth Sciences and Engineering* 6, 6 (2013), 1453–1459.

[56] Y. Yang, B. Zhang, Q. Feng, H. Cai, M. Jiang, K. Zhou, F. Li, S. Liu, and X. Li. 2019. Towards locating time-varying indoor particle sources: Development of two multi-robot olfaction methods based on whale optimization algorithm. *Building and Environment* 166 (2019), 106413. DOI: https://doi.org/10.1016/j.buildenv.2019.106413

[57] S. Yim, J. Fung, A. Lau, and S. Kot. 2009. Air ventilation impacts of the "wall effect" resulting from the alignment of high-rise buildings. *Atmospheric Environment* 43, 32 (2009), 4982–4994. DOI: https://doi.org/10.1016/j.atmosenv.2009.07.002

[58] S. Yun, C. Chen, J. Li, and L. Tang. 2011. Wildfire spread simulation and visualization in virtual environments. In *Proceedings 2011 IEEE International Conference on Spatial Data Mining and Geographical Knowledge Services (ICSDM '11)*, 315–319. DOI: https://doi.org/10.1109/ICSDM.2011.5969054

[59] Y. Zhang, Y.-F. Ma, S. Song, Y.-L. Lü, S. Zhang, and Q. Wu. 2020. Fate simulation of 2, 4, 4′-Trichlorobiphenyl in the Bohai Rim using the multimedia model. *Huanjing Kexue/Environmental Science* 41, 6 (2020), 2625–2634. DOI: https://doi.org/10.13227/j.hjkx.201911140

[60] L. Zhong, W. Cheng, Y. Jiang, P. Wu, and H.-W. Feng. 2009. Study of three-dimensional dynamic simulation on dense gas leakage and dispersion. *Dongbei Daxue Xuebao/Journal of Northeastern University* 30, SUPPL. (2009), 131–133.

Received 17 October 2025; revised 6 July 2026; accepted 13 August 2026