



UNIVERSITY OF RIJEKA
FACULTY OF MARITIME STUDIES

Darko Glujić

**ADVANCED MODEL OF FIRE SPREAD
IN SHIP ENGINE ROOM BASED ON
VIRTUAL REALITY**

Ph.D. Thesis

Rijeka, 2024.



UNIVERSITY OF RIJEKA
FACULTY OF MARITIME STUDIES

Darko Glujić

ADVANCED MODEL OF FIRE SPREAD IN SHIP ENGINE ROOM BASED ON VIRTUAL REALITY

Ph.D. Thesis

Mentor: Prof. Dean Bernečić, Ph.D.

Co-mentor: Prof. Goran Vukelić, Ph.D.

Rijeka, 2024.



SVEUČILIŠTE U RIJECI

POMORSKI FAKULTET

Darko Glujić

NAPREDNI MODEL ŠIRENJA POŽARA U BRODSKOJ STROJARNICI PRIMJENOM VIRTUALNE STVARNOSTI

Doktorska disertacija

Mentor: Prof. dr. sc. Dean Bernečić

Komentor: Prof. dr. sc. Goran Vukelić

Rijeka, 2024.

Mentor: Prof. Dean Bernečić, Ph.D., University of Rijeka, Faculty of Maritime studies, Croatia

Co-mentor: Prof. Goran Vukelić, Ph.D., University of Rijeka, Faculty of Maritime studies, Croatia

The doctoral thesis was defended on _____ at the University of Rijeka, Faculty of Maritime Studies, Croatia, in front of the following Evaluation Committee:

- 1.
- 2.
- 3.

Preface

This thesis has been written with the great and unselfish help of Prof. Dean Bernečić, mentor, and Prof. Goran Vukelić, co-mentor, each providing a unique approach.

A great friend, Assist. Prof. Goran Vizentin, was always there for me to offer any assistance or just a word of encouragement.

Daniel Milin greatly helped with his unique Unreal engine and VR knowledge.

Assoc. Prof. Dario Ogrizović saved me a lot of nerves and time with digitalizing questionnaires and always pushed me forward.

A special thanks to Assist. Prof. Asmir Gračanin for help with questionnaire design and result interpreting.

Abstract

The occurrence of fire on a ship is a significant safety concern, owing to the restricted space and limited evacuation options, which can quickly intensify the risk. Implementing robust fire prevention and management strategies is crucial in marine settings to safeguard lives, the ship's cargo, and the vessel itself from the destructive impact of fire.

To reduce the risk of deadly situations, ship owners and operators, among various approaches, focus on consistently enhancing their crew members' firefighting and evacuation skills.

Fire training inside the virtual reality (VR) provides a risk-free and regulated setting, enabling maritime personnel to practice handling lifelike fire emergencies without the dangers inherent in real fire situations. This approach to training improves overall safety by offering practical experience and instant feedback, essential for equipping crews with the skills needed to deal with onboard emergencies effectively.

The primary limitation of conventional VR training scenarios lies in their inability to simulate the dynamics of fire spread accurately. Typically, these VR scenarios utilize generic, computer game-based fire and smoke simulations that fall short of replicating the true nature of real-life fires. However, this shortfall can be effectively addressed by integrating computational fluid dynamics-based (CFD) models of fire spread into VR training simulations, thereby significantly enhancing the realism and educational value of these scenarios.

Based on the probability of occurrence, CFD software defines and models the three most likely fire scenarios (fire of main engine fuel line, fire of fuel oil purifier, and fire of oily rags left in a bucket) in the ship engine room. Software interface to couple CFD results and VR environment is developed, creating an advanced fire spread model in the ship engine room.

Additionally, a survey was conducted to determine the difference between CFD-modelled fire represented in VR and generic computer game fire in VR. According to the survey results, a group of test subjects with previous experience in fires in enclosed

spaces has noted that CFD-modelled fire gives a more realistic appearance than generic fire in VR.

Sažetak

Pojava požara na brodu predstavlja značajan sigurnosni rizik zbog ograničenog prostora i limitiranih opcija evakuacije. Primjena strogih strategija prevencije i upravljanja požarima ključna je u pomorskom okruženju kako bi se zaštitili životi, teret i sam brod od destruktivnog utjecaja požara. Da bi se smanjio rizik od smrtonosnih situacija, vlasnici i posade brodova, među različitim pristupima, fokusiraju se i na kontinuirano poboljšanje vještina gašenja požara i evakuacije svojih članova posade.

Protupožarni trening unutar virtualne stvarnosti (engl. virtual reality, VR) pruža sigurno i regulirano okruženje, omogućujući posadi uvježbavanje upravljanjem stvarnim hitnim situacijama s požarom, bez opasnosti koje dolaze u stvarnim požarnim situacijama. Ovaj pristup poboljšava sigurnost nudeći praktično iskustvo i trenutnu povratnu informaciju, što je bitno za usvajanje vještina potrebnih za učinkovito suočavanje s hitnim situacijama na brodu.

Glavno ograničenje konvencionalnih scenarija VR treninga leži u nesposobnosti točnog simuliranja dinamike širenja požara unutar VR okruženja. Takvi VR scenariji koriste generičke simulacije požara i dima, bazirane na principima računalnih igara, koji ne uspijevaju adekvatno replicirati pravu prirodu požara. Međutim, ovaj nedostatak se može učinkovito riješiti integracijom modela širenja požara baziranog na računalnoj dinamici fluida (engl. computational fluid dynamics, CFD) u VR okruženje, čime se značajno pojačava realizam i edukativna vrijednost ovih scenarija protupožarnog treninga.

Koristeći takav pristup, u ovom su radu modelirana tri scenarija požara u strojarnici broda, temeljena na iskustvenoj vjerojatnosti njihova nastanka (požar na glavnoj liniji goriva motora, požar na pročišćivaču goriva i požar na zauljenim krpama u kanti). Rezultati širenja požara dobiveni računalnom dinamikom fluida preneseni su u VR okruženje posebno razvijenim programskim sučeljem, što predstavlja jedan od

doprinosa ovoga rada. Time je stvoren napredni model širenja požara u brodskoj strojarnici u VR okruženju temeljen na računalnoj dinamici fluida.

Nadalje, provedena je anketa među korisnicima kako bi se utvrdila uspješnost odabranog pristupa modeliranja požara nasuprot onom generičkom koji je baziran na principima računalnih igara. Prema rezultatima ankete, grupa ispitanika koji su imali prethodno iskustvo požara u zatvorenim prostorima primijetila je da je požar u VR okruženju modeliran pomoću računalne dinamike fluida daje realističniji dojam od onoga generički modeliranog.

Keywords:

computational fluid dynamics; virtual reality; fire; ship engine room.

Ključne riječi:

računalna dinamika fluida; virtualna stvarnost; požar; brodska strojarnica.

TABLE OF CONTENTS

Preface	I
Abstract	II
Sažetak	III
Keywords:.....	IV
Ključne riječi:	IV
1. INTRODUCTION	1
1.1. Research problem and hypothesis	3
1.2. Research aims	4
1.3. Research objectives	5
1.4. Research methods	6
1.5. Thesis outline	6
2. LITERATURE OVERVIEW	8
2.1. Computational fluid dynamics (CFD) for fire modeling	8
2.2. Virtual reality (VR) technology for firefighting training.....	10
2.3. Integration of CFD simulation results into VR environment	12
2.4. VR and serious game firefighting training.....	13
2.5. Application of VR technology for maritime firefighting training.....	15
2.6. Research gap	17
3. THEORY OF A FIRE MODELLING USING CFD	18
3.1. Introduction to CFD	18
3.2. Theoretical background of CFD fire modeling	21
3.2.1. Mass conservation.....	21
3.2.2. Momentum conservation	21
3.2.3. Energy conservation.....	21
3.2.4. Turbulence model.....	22
3.3. Radiation Models.....	24

3.3.1.	Six-Flux Radiation Model	26
3.3.2.	Absorption Coefficient	26
3.3.3.	Species conservation	27
3.3.4.	Combustion model	28
3.4.	Auxiliary equations	30
3.4.1.	Density	30
3.4.2.	Buoyancy	31
4.	THEORY OF FIRE MODELLING IN VR ENVIRONMENT	31
5.	A MODEL OF A SHIP ENGINE ROOM	36
5.1.	Model of engine room to be used for CFD simulation.....	37
5.2.	Scenarios to be run in CFD	39
6.	CFD MODELING OF FIRE INSIDE ENGINE ROOM.....	40
6.1.	Smartfire CFD software	40
6.2.	Smartfire mode of usage	41
6.3.	Geometry creation using Smartfire scenario designer	42
6.4.	Case specification using the case specification tool	44
6.4.1.	Defining common properties for the created case	46
6.5.	Fire modelling.....	52
6.5.1.	Fire of the main engine fuel oil pipeline	52
6.5.2.	Fire of the fuel oil purifier.....	54
6.5.3.	Fire of oily rags left in a bucket.....	56
6.6.	Mesh generation.....	59
6.6.1.	Smartfire interactive meshing system.....	59
6.6.2.	The meshing of the main engine fuel oil pipeline fire scenario	60
6.6.3.	Meshing of the purifier fire scenario	63
6.6.4.	The meshing of the oily rags left in a bucket scenario	64
6.7.	Running scenarios.....	64

6.8.	Results of CFD analysis	66
6.8.1.	Results of the main engine fuel oil pipeline fire scenario.....	67
6.8.2.	Results of the fuel oil purifier fire scenario.....	68
6.8.3.	Oily rags left in a bucket fire scenario results	70
6.9.	Research validation	71
6.10.	Stecklers experiment.....	71
6.11.	Simulation of the Stecklers experiment	75
6.12.	Comparison of results for Steckler's experiment and simulation	77
7.	COUPLING CFD AND VR	84
7.1.	Voxels.....	84
7.2.	.VTU filetype	87
7.3.	.VTK filetype	88
7.4.	Unreal Engine.....	89
7.5.	Compatibility issues within Unreal engine and .vtk and .vtu files	91
7.5.1.	.VDB file type	91
7.5.2.	Conversion of .VTU to .VDB.....	92
7.5.2.1.	ParaView software	93
7.5.2.2.	ParaView conversion	94
7.5.2.3.	Import of .VDB file with smoke data to Unreal engine	95
7.5.2.4.	SideFX's Houdini	95
7.5.2.5.	Import of data into Houdini software	96
7.6.	C++ code to convert .vtu data to .vdb.....	97
7.7.	Description of converter files	97
7.8.	Import of converted data into Unreal engine.....	107
8.	USER EXPERIENCE OF ADVANCED VR FIRE SPREAD MODEL...	110
8.1.	Contents of the questionnaires	111
8.2.	Survey results.....	112

8.3. Firefighter results.....	112
8.4. Seamen results.....	115
8.5. Conclusion of survey results.....	120
9. CONCLUSION.....	120
Table of figures:.....	123
Table of tables:.....	126
References:	127

1. INTRODUCTION

The global intensification of maritime traffic, coupled with the determined pursuit of profit, has increased the number of naval vessels [1], mainly designed to be cost-effective and profitable [2]. Consequently, various accidents occur, resulting not only in material damage but also, all too often, in the loss of human lives [3]. One form of such an accident is a fire on a maritime vessel. To avoid the causes and minimize the consequences of such accidents, the regulatory organization (International Maritime Organization, IMO) develops various strategies for fire prevention and firefighting. It also prescribes minimum technical requirements for ship design and fire safety systems, as well as crew training and exercises. These regulations can be found in Chapter II of SOLAS (The International Convention for the Safety of Life At Sea) and the FSS Code (Fire Safety Systems Code).

Data [4] reveals a notable surge in engine room fires aboard container ships between 2019 and 2021. Initial data from public outlets suggests that this pattern extends to the worldwide fleet, indicating that engine room fires in 2022 to 2023 remain consistent with the levels observed in 2021. This information underscores certain prevalent causes and factors contributing to engine room fires, Figure 1.

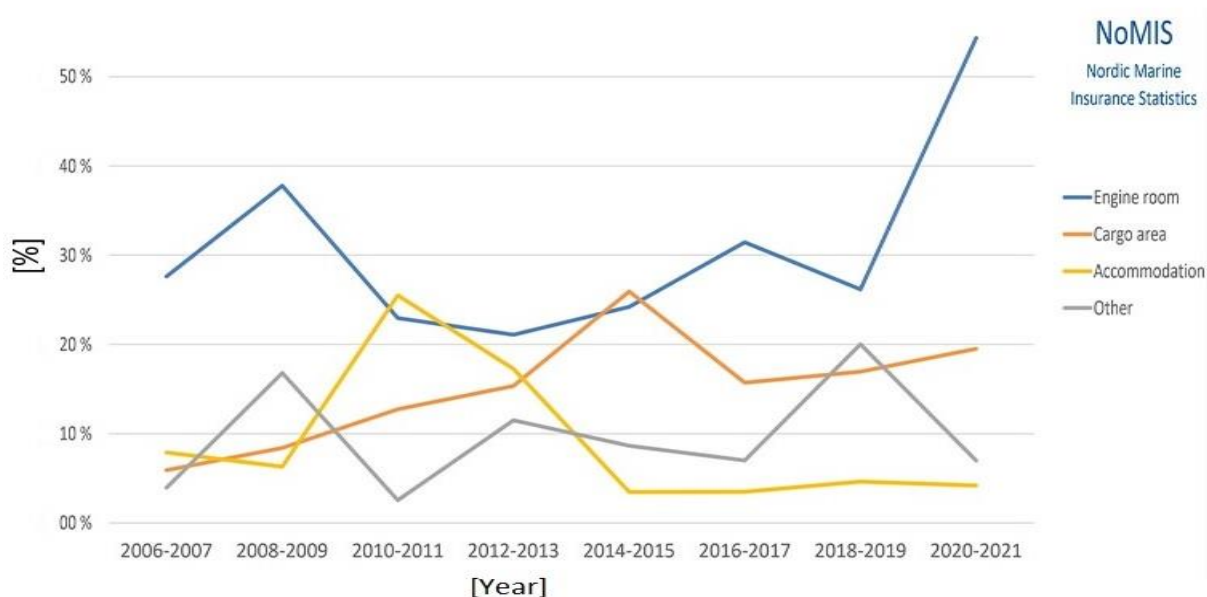


Figure 1 – Number of fires in relation to a part of a ship where it originated [4].

Fire drills and abandon ship drills are crucial for preserving life on board in the event of a fire. These drills are regularly conducted on board, albeit without an actual fire, rarely with fire in a safe environment. The crew's compliance with safety procedures and knowledge of behavior during a fire is achieved through these exercises, contributing to a more successful firefighting or evacuation when necessary [5]. However, a significant detail missing in such exercises is the absence of real danger, i.e., human reaction to a genuinely present threat. There are fundamental behavioral differences in facing an imaginary threat compared to an immediate, real danger [6]. In the experience of immediate danger but in controlled and safe conditions, virtual reality (VR) technology can help.

Today's computer technology allows for a simulated representation of objects and phenomena in virtual reality with a high degree of "realism." Virtual reality is a three-dimensional multimedia environment achieved through visualizing a real or imagined environment displayed on a screen or special stereoscopic devices (glasses or helmets with two built-in liquid crystal displays). The experience is complemented by sounds and vibrations, and it is possible to simulate tactile and olfactory sensations. Virtual reality has proven particularly useful for educational purposes, leading to significant cost savings in training for complex tasks [7] while preserving the health and safety of trainees.

Conducting firefighting exercises in a real environment would mean risking the destruction of the vessel by fire or, at the very least, severe damage and endangering the crew. This is unacceptable from economic, ethical, and educational perspectives. Insurance companies and ship owners have turned to new technologies, specifically VR technology, to provide better tools for maritime education and thus reduce their costs or losses in the event of an accident. In virtual reality, it is possible to create a computerized 3D model of the entire or part of a maritime vessel, with any features and events in real-time, such as a fire as an occurrence that requires a response. VR technology provides an almost immediate experience for users who react differently than in exercises with an imagined threat [8]. An additional advantage is that the crew can start training before boarding the vessel or even while it is still under construction if a virtual model is created for that space. This can reduce the time needed to familiarize oneself with the vessel, the layout of rooms, and the position of various machinery and equipment.

For some users, the use of digital technologies may create resistance. Additionally, using digital technologies can cause nausea, dizziness, headaches, and fainting [9]. Despite these drawbacks, conducting exercises in a VR environment contributes to comprehensive and timely crew readiness and should certainly become standard [10].

The main drawback of conducting firefighting exercises in a VR environment is that the real spread of fire, i.e., heat and smoke through space, is not considered [11]. In commercially available VR models for firefighting training, the fire is depicted as a localized blaze that does not spread through space. This gives the user time to acclimate to such a "threat" and raises questions about the purpose of such training. Another drawback is that the fire is often depicted exaggeratedly to intensify a user's engagement, resulting from VR being used primarily for computer games. One reason for this current level of development is that the spread of fire must be modeled using fluid dynamics, which is often avoided due to its complexity. Furthermore, such modeling would need to be carried out for each new space built in the virtual environment, leading to additional costs and increased duration in the process. Therefore, the graphic designer often leaves it to decide how to represent the fire in the virtual environment.

1.1. Research problem and hypothesis

It is clear that there is no existing virtual model of ship engine room with employees CFD for fire spread visualization, which is the main problem to solve.

The subject of this research is to explore and determine the crucial factors that influence the spread of heat and smoke gases in a ship engine room, and to use the gathered knowledge to build a numerical model of fire spread and transfer the results into a virtual environment.

The final result of the research and the scientific contribution will be the establishment of an advanced fire spread model in a virtual ship engine room based on the results of numerical analysis of the spread of heat and smoke gases, and the introduction of a new methodology for fire-fighting training in the ship's engine room.

The research results will be applicable in a way that the advanced virtual fire spread model in the ship's engine room can be used for fire-fighting training and education of future and current seafarers in a safe environment. Additionally, the model also offers the possibility of conducting behavioral studies on subjects.

From the defined problem and subject, and the selected goal of scientific research, the fundamental scientific hypothesis arises:

H0: It is possible to transfer the fire spread model obtained through computational fluid dynamics into advanced virtual reality.

This hypothesis also indicates several auxiliary hypotheses:

AH1: It is possible to improve fire training in the ship's engine room by using a numerical model of fire spread transferred into advanced virtual reality.

AH2: Response times of subjects in the advanced virtual reality model will significantly differ from responses in the basic model.

AH3: The adoption of fire-fighting skills will be more successful using the advanced virtual reality model compared to the basic one.

The hypothesis will be attempted to be proven with an advanced numerical model of fire spread in the ship's engine room transferred to a VR environment. Such an advanced model will be used for conducting fire drills and will be tested and evaluated through a survey by a group of subjects.

1.2. Research aims

Creating a sophisticated model for simulating fire spread in a ship's engine room, utilizing Virtual Reality (VR) in conjunction with Computational Fluid Dynamics (CFD), represents an innovative method to improve our comprehension and handling of fire safety in the maritime sector.

The fundamental goal of this initiative is to forge a highly authentic and interactive simulation that replicates fire incidents within the ship's engine room. The fusion of VR and CFD in the model will render the dynamics of fire and smoke in a lifelike 3D environment and enable users to interact with the scenario, observing real-time outcomes of various actions and conditions.

VR's immersive capabilities serve as a potent educational and training tool. It enables trainees to engage with fire scenarios safely, learn to maneuver through smoke-obscured environments, identify fire origins, and operate firefighting gear. This practical experience is invaluable in preparing crew members for real-world emergencies.

CFD is integral to the project, offering a scientific base for the simulation of fire and smoke movements. Fire propagation in a confined area like a ship's engine room is influenced by numerous factors, including ventilation, heat sources, and flammable materials. CFD mathematically models these aspects, facilitating the simulation of fire and smoke spread under various conditions.

1.3. Research objectives

Creating an advanced VR-CFD model for simulating fire in a ship's engine room is an ambitious project with the potential to transform fire safety training and readiness in the maritime field. By offering a realistic, interactive, and secure setting for learning and practicing fire response tactics, this technology promises to significantly improve onboard safety, preparing crews more effectively for emergency scenarios. Research objectives include:

- The first step is to conduct a literature review to determine what studies have been conducted in this field to detect possible research gap(s),
- Next, three fire spread scenarios in the ship engine room will be defined based on the probability of fire occurrence,
- Using fluid dynamics principles, those three scenarios will be modeled in one of the CFD modeling software,

- VR environment of the engine room will be developed using software that best suits the needs of research aims,
- A plausible way to transfer CFD results to VR will be determined and executed,
- The fire spread model will be transferred from CFD to VR using a method that has proven to be the best,
- The obtained advanced model will be tested with specific users using a questionnaire incorporating video of the results transferred to VR.

1.4. Research methods

In scientific research, the processing and analyzing results will involve combining numerous scientific methods. These methods will be used for problem investigation and to demonstrate the formulated hypothesis. The methods include:

- Compilation method for a comprehensive review of existing scientific works,
- The descriptive method, along with a deductive method for describing the problem,
- Finite volume method in creating a numerical model,
- Comparative method for validating the numerical model,
- Computer programming to transfer the results of the numerical model into a VR environment,
- Inductive-deductive method in drawing general and specific conclusions about the transfer of results from the numerical model to the VR environment,
- Statistical processing method for analyzing the results of questionnaires for participants in fire spread scenarios.

1.5. Thesis outline

This thesis comprises eight chapters as follows:

- Literature overview: where up-to-date research in the field will be studied to gain a unique insight into the problem and possibly get some ideas or solutions to the issues that will occur during research,

- Theory of fire modeling using CFD: the foundation of fluid dynamics using computer routines with governing equations,
- Theory of fire modeling in VR environment: An outline of fire modeling inside game engines is given,
- A model of a ship engine room: foundations for building a numerical 3D environment of a ship engine room are presented, along with sketches and general dimensions,
- CFD modeling of fire inside engine room: detailed overview of the making of CFD models for three scenarios,
- Coupling CFD and VR: all methods that have been tested and used are to be explained,
- The user experience of advanced VR fire spread model: questionnaires and results are presented and discussed,
- Conclusion: An outline of the results with a discussion of future work is presented.

2. LITERATURE OVERVIEW

The proposed research encompasses applying computational fluid dynamics to fire spread, using virtual reality technology for firefighting training, and transferring results obtained from CFD into VR. This review focuses specifically on these topics. Additionally, behavioral challenges arising from VR usage are addressed.

2.1. Computational fluid dynamics (CFD) for fire modeling

To prevent potentially fatal scenarios, ship owners and operators invest in continuous training for firefighting and evacuation capabilities among crews. The International Convention on Standards of Training, Certification, and Watchkeeping for Seafarers (STCW) [12], established by the International Maritime Organization (IMO), provides principles for firefighting training. Additionally, the IMO issued the Guidelines on Evacuation Analysis for New and Existing Passenger Ships [13]. Understanding fire dynamics and human behavior is essential for creating credible firefighting and evacuation scenarios.

Traditionally, this understanding relied on historical case studies, analytical calculations, and limited experiments [14–16]. However, with advanced numerical routines and powerful computers, CFD simulations allow the simulation of fire spread scenarios. CFD fire spread analysis is proven effective in early-stage ship design [17] and risk assessment for fire-prone ship systems such as engine rooms [18] or LNG tanks [19]. Moreover, CFD fire dynamics analysis has been integrated with evacuation models to assess passenger evacuation performance in ship fires [20]. An algorithm was developed to consider the interaction between the CFD model of an LNG gas spill and a crew evacuating on a floating LNG bunkering terminal [21].

In the realm of numerical simulation of evacuation, scientists have developed various models focusing on human pedestrian behavior, including cellular automata models [22], agent-based models [23], CFD models adjusted for pedestrians [24], and the recent multi-granularity quality function deployment model [25]. Research has also

been conducted on developing risk models for the evacuation processes of passengers on cruise ships [26,27], as summarized in recent review papers [28,29].

However, pure numerical methods have their weaknesses. Numerically obtained results need experimental validation for credibility, and these results are often challenging for the average user to visualize, especially in fire spread scenarios. This is where the innovative concept of virtual reality (VR) comes into play, particularly in the last decade. VR provides a fully immersive experience, placing the user in a virtual environment of digitally created objects and blocking the perception of real surroundings [10]. Virtual content is typically displayed via stereoscopic displays, such as wearable hardware, large screens, or wall projections. VR technology has been adopted for firefighting and evacuation training using game engines to create immersive 3D environments impacted by fires [30]. Although successfully applied in fire evacuation training for buildings and urban spaces [31–33], its impact in the maritime sector is yet to be fully realized.

In prior research on computational fire spread modeling, scientists primarily concentrated on developing new and utilizing commercially available computational fluid dynamics (CFD) software. A comparison of available software solutions (Ansys CFX, Fluent, and ADINA) [34] revealed that finite element method-based solutions depend more on the mesh type and quality than finite volume method-based solutions. The finite volume method was also proven to be a more resource-efficient solution. Analyzing 126 scientific articles published from 2010 to 2020 dealing with CFD applications in fire, gas, and explosion modeling [35], there is an increased effort in using computational models to assess risky industrial situations. It was demonstrated that in enclosed space fires, the greatest threat is inhaling toxic gases. Challenges include reduced visibility, high temperature, heat transfer, and oxygen concentration reduction. Lawmakers impose stricter regulations on new building constructions to minimize risks, particularly in fire safety requirements. Given the growing complexity of structures, three-dimensional CFD simulations for fire calculations are time-consuming and resource-intensive. Typically, simulation models reference parts of structures to conserve resources, contributing to more efficient and safer construction [15]. Various software solutions and approaches exist for CFD fire simulations. Kevin McGrattan described fire modeling with low Mach numbers and large vortices with a single-step fuel and oxygen reaction, incorporating radiation [36]. Georgios Maragkos focused on

fire and heat transfer modeling through convection, emphasizing boundary layers and turbulent motion [12]. Simulations based on experimental values were explored in works [37][38].

An overview of advancements in comprehensive (CFD) simulations related to fire and explosion within the context of safety science is provided by Zhenghua Yan [39]. Fires are categorized into conventional and spontaneous types, where conventional fires involve undesired combustion in air, and spontaneous ignition fires involve undesired combustion in porous media. The behavior of spontaneous ignition fires significantly differs from that of conventional fires due to the dominant influence of porous media on the flow. While a fire primarily consists of a diffusion flame with initially separated fuel and oxidant, accompanied by low-speed, incompressible flow, explosions typically entail premixed combustion with initially well-mixed fuel and oxidant, along with high-speed flow requiring consideration of compressible effects. Given the intricacies of fires and explosions, a comprehensive CFD simulation must meticulously address turbulence, turbulent combustion, two-phase flow (for cases involving liquid droplets and/or solid particles), conjugate heat transfer between gas and solid (encompassing thermal radiation, convective heat transfer, and heat conduction within solids), and the pyrolysis of combustible solids. The interactive nature of these processes is also discussed. Additionally, the study presents certain developments by the author, accompanied by illustrative simulations conducted using Simtec software, which serves as the implementation platform for these advancements.

2.2. Virtual reality (VR) technology for firefighting training

In recent years, the rapid progression of information technology has led to advancements in the visualization and interaction capabilities of virtual reality (VR) technology. This has increased interest in incorporating VR into educational practices, drawing the attention of scholars. A literature analysis approach explores VR technology in higher education [40]. Several dozen empirical studies from the Web of Science indexing database, conducting thorough readings and analyses to extract insights into the application of VR in higher education, are studied. The findings

highlight that undergraduates are the primary focus of VR applications in higher education, particularly in science, engineering, and medical majors. Humanities and social sciences receive relatively less attention. Currently, VR devices in higher education mainly involve computers and headsets, lacking the desired level of portability. Furthermore, there is a recognized absence of pre-class guidance and training for students using VR equipment. Comparative assessments with traditional education consistently indicate the positive effects of VR on higher education and teaching. These effects primarily influence student behaviors, subsequently affecting learning outcomes. Researchers predominantly employ traditional approaches for evaluation, using questionnaires and tests for data collection. Analytical techniques mainly include difference analysis and descriptive analysis. Given these research outcomes, the paper concludes with further recommendations to enhance VR integration in higher education.

The findings from [41] underscore the effectiveness of VR in training scenarios that are difficult to execute in real-life settings. Particularly, motion-capture-based VR technologies show promise in firefighting training, enabling participants to collaborate in environments that are typically hard to access. Despite these advantages, the study highlights that enhancing immersion in such training scenarios remains a significant challenge. The study provides a prototype for a VR application tailored for the multi-user training of maritime firefighters. Future research directions should focus on evaluating the initial findings, expanding the training scenarios, and quantifying the progress made in training. This approach represents a significant advancement in emergency training, leveraging the latest VR and motion-capture technology to prepare individuals for real-world challenges.

The research [42] revealed that using more realistic devices, which closely simulate actual tools, does not necessarily lead to an enhanced user experience. This finding underscores the importance of a strategic approach in developing and applying VR control devices. The study offers valuable empirical data on determining the most effective combinations of VR control device modalities, especially in VR simulations and training designed to replicate field-based scenarios. This insight is significant for advancing the integration of control devices in VR systems, particularly for training purposes that mimic real-life situations.

A study [43] shows that the use of the VR fire extinguisher in the VR application was well-received, with an approval rating of over 80%. This high rating indicates that the application successfully raises user awareness about the importance of fire extinguishers and the proper steps to use them in extinguishing fires. Furthermore, expert feedback has offered additional insights and suggestions, which will be instrumental in further enhancing and refining the quality of the application in future iterations. This ongoing improvement process is crucial for the continued effectiveness and relevance of the VR-based educational tool in fire safety training.

Integrating VR techniques into firefighting training presents a highly effective solution to reducing the consumption of resources such as workforce, materials, and finances [44]. The application of the HTC Vive in conjunction with the Unity3D engine for the design and development of a firefighting training system is discussed. Key technologies utilized in this system include particle systems for simulating flame and smoke diffusion, navigation grids for non-experiencer tracking, an animation system for content creation, and collision detection for enabling interactions between characters and objects.

A study proposed a series of methods to convert CFD model results and a real-time data processing framework to create a firefighting simulator based on CFD model outcomes. This simulator can calculate heat, smoke, and invisible toxic particle spread. In construction, it is crucial to emphasize connecting Building Information Modeling (BIM) with VR systems to enhance emergency exit training simulations. There is a clear need for a structured method for BIM and VR system integration [45].

2.3. Integration of CFD simulation results into VR environment

Virtual reality training has become indispensable in many professions, including firefighting, where VR provides essential experience for inexperienced firefighters. The challenge lies in aligning VR firefighting training with CFD model results [46].

Since no solution automatically converts CFD model results into VR simulations, a proposed systematic architecture would generate a dedicated workflow for each VR simulation compatible with CFD models. Furthermore, data processing possibilities

were explored through VR data preparation, offering a data fusion strategy for simplified multi-platform integration [47].

Multi-platform integration was described by creating a CFD air and temperature flow model in an agricultural greenhouse, displaying the results in a VR environment using code written in the C programming language. Additionally, results were validated in a real greenhouse, and an interface was created for selecting desired categories of displayed phenomena [48][49]. The potential for integrating CFD models with VR environments and their educational use was verified, presenting a case study illustrating integration through generic methods [50].

The closest attempt to bring fluid physics into the VR environment involved implementing code written in the C programming package, which simplified the calculation of smoke spread using Navier–Stokes equations in a firefighting simulator [11]. The movement of smoke particles was mathematically described and limited for real-time calculations. Visualization was achieved through tracers rotating toward the observer, reducing resource requirements. However, the smoke visualization obtained clear edges, losing realism. Lastly, the Unity programming package was used for VR display, which, in newer versions, qualitatively lags behind the Unreal Engine programming package, resulting in lower display quality but showcasing the potential for substance and phenomenon calculations based on real physics, feasible in real-time as technology advances.

2.4. VR and serious game firefighting training

A review of VR and serious game concepts in firefighting training indicates that technology allows for almost immediate experience, providing novice firefighting personnel with knowledge and skills to reduce incidents and injuries during interventions, considering that 33% of all firefighter injuries are related to fire [51]. VR environment training positively affects procedural learning but not conceptual learning [52]. Previous experience with computer games and computers generally did not affect the quality and quantity of learning. However, prior knowledge about fire deepened the experience in a virtual environment and improved procedural learning. Comparing participant behavior in real fire and VR fire-affected environments revealed a

correlation coefficient of 0.9958 [53]. Given the very high correlation coefficient, it is clear that the VR environment is highly suitable for behavioral stress studies, particularly regarding fires.

Enhancement of firefighter training in decision-making during stressful situations through a serious game solution is addressed [54]. The game, aligned with the Stress Exposure Training (SET) approach and informed by stressors identified in the literature, incorporates relevant game mechanics. The evaluation of the game had a dual focus: assessing its ability to elevate participants' stress levels and observing whether decision-making performance improved for participants under stress. The framework and prototype underwent empirical research, including participants' self-reported stress levels using the State-Trait Anxiety Inventory for Adults (STAI) and physiological measurements (galvanic skin response (GSR) and heart rate (HR)). Results highlight the potential of serious games in enhancing firefighter decision-making in stressful situations. These significant findings contribute to ongoing efforts to improve decision-making under stress across various settings and scenarios.

The absence of a safety culture among students and teachers, coupled with a lack of knowledge and training on safety protocols for dealing with fires, has resulted in adverse outcomes, including avoidable injuries and, in some cases, fatalities. This problem is addressed in [55], where the development of a Serious Game prototype designed to instruct students on fire evacuation procedures in schools is outlined. The primary objectives are to facilitate students in learning fire safety protocols, understanding appropriate behavior in fire emergencies, and initiating discussions to foster a positive fire safety culture. The prototype underwent testing and evaluation involving 35 public school students aged 12 to 16. The findings revealed a significant improvement in students' knowledge of proper behavior during a fire emergency after engaging in the game. Moreover, the discussions prompted by the game contributed to establishing a positive fire safety culture within the school environment.

Fire incidents in tall buildings can lead to substantial property damage, injuries, and loss of life. Enhancing human evacuation behavior is an effective strategy to mitigate injuries and fatalities in such situations. Numerous studies have underscored the efficacy of behavioral skills training (BST) in equipping individuals of diverse age groups with essential safety skills for emergencies. In the study [56], a theoretical model termed "VR-SG-BST" is introduced, integrating virtual reality (VR), serious

games (SGs), and BST for fire safety training. The model's evaluation focuses on older adults in care facilities due to their heightened vulnerability from limited fire safety knowledge and potential health issues. The study anticipates that engagement with the SG will enhance elderly individuals' learning performance in fire evacuation skills, fostering increased interest and a sense of enjoyment in fire safety training. The SG is posited to improve emergency evacuation skills effectively. The investigation encompasses two concurrent components: 1) a comparative analysis of fire safety behavior in older adults between an experimental and a control group, and 2) an assessment of the effects of BST on fire safety by exclusively comparing the experimental group in pre- and post-BST phases. The study employs in situ training (IST) to evaluate knowledge and skill acquisition during the experimental phase. The findings suggest that the VR-SG-BST model holds promise as an alternative approach to enhance the behavioral skills of older individuals during a fire. Additionally, a history of computer game experience improves learning performance and fire safety behavior among older people. The study's outcomes offer valuable suggestions and guidelines for the design of VR-SGs in the practical application of BST.

2.5. Application of VR technology for maritime firefighting training

One instance of applying VR technology is in the research of a ship firefighting training system [57]. This study created systems for extinguishing fires using water and carbon dioxide, allowing users to operate a virtual avatar through the Inverse Kinematics method. The researchers successfully simulated fire and carbon dioxide dispersal using a particle system. Although the study reports positive outcomes from the experimental VR training, it lacks empirical evidence to support this claim. A VR simulation of a ship deck was developed using Unity 3D, a widely used game engine [58]. The same authors' subsequent extension of this research introduced a fully immersive firefighting training experience on the ship, utilizing wearable technology [59]. Another VR project from the same Chinese institution focused on ship life-saving training, offering training in rescue procedures, including a multiplayer option [60].

Another study using the same Unity 3D engine developed a fire simulation training system for the interior of a ship [61]. This study emphasized a training evaluation scheme, considering extinguishing effectiveness, completion time, safe operation, and the search for items and personnel, assigning different weights to these aspects to determine the training's success rate.

A further study concentrated on controlling avatars for firefighting training aboard a ship, using Unity 3D to create a VR ship deck environment [62]. The study detailed Unity 3D's method of simulating fire, which simplifies the physical characteristics of fire through a particle system, controlling the fire's spread and extinguishing time.

Researchers in the UK and Finland introduced a virtual training technology called Immersive Safe Oceans [63,64]. This technology, offering complete immersion in VR, introduced four maritime safety training episodes, including an electric cabin fire simulation. The episode aims to evaluate trainees' behavior in performing specific actions during a fire.

For the Belgian Navy, a fully immersive VR scenario of a kitchen cabinet fire on a ship was developed using the Unreal game engine [65]. This research highlighted challenges in simulating fire and smoke, particularly the high demand for graphical processing units (GPU) due to smoke build-up and numerous draw calls.

German researchers created a firefighting scenario based on the 2019 Yantian Express container ship fire [66]. The ship's size necessitated simulating separate gangways between container bays using the Unity 3D engine. This application allows for multi-role training with advanced controllers resembling water guns.

An Indonesian team developed an Android VR app for firefighting equipment inspectors, featuring a ship tour mode for equipment familiarization and an inspection mode requiring users to locate equipment and complete a checklist within a time limit [67].

Lastly, in 2022, a ship firefighting training simulator was developed, featuring physics-based smoke simulation [11]. This simulator uses a Lagrangian vortex dynamics framework for more realistic fire and smoke representation, marking a progression from earlier research that relied solely on game engine tools for fire

dynamics simulation. The results from this simulator demonstrate enhanced realism in fire spread.

2.6. Research gap

Learning from the conducted literature review, studies that combine CFD results with VR technology are extremely rare. There are some simplified attempts on the subjects, but only a few did the actual coupling of CFD and VR. Out of those few that did the actual coupling, none is focused on the fire, especially not in the engine room area, thus pointing to a gap in the researches which potentially could help save lives if properly investigated.

This issue could be addressed by integrating VR with CFD. CFD tools can model the fire's origin and subsequent spread, including changes in temperature and smoke development. Using these data, a more realistic fire dynamic could be simulated in VR, providing users with enhanced experiences for advanced firefighting and evacuation training.

A noteworthy advancement in this area has been made by a team of Chinese researchers who developed a VR-based fire training simulator integrated with fire dynamics data [46]. This study introduced various data conversion techniques and a real-time processing framework to create a fire training simulator grounded in CFD simulation capable of calculating physical aspects like toxic gases, heat, smoke, and flames. A similar approach, coupling CFD with VR, was employed to simulate airflow in a greenhouse VR environment [49]. Although this research didn't directly address fire spread, the methodology could be adapted to tackle the challenges of fire spread simulation.

3. THEORY OF A FIRE MODELLING USING CFD

3.1. Introduction to CFD

Computational Fluid Dynamics (CFD) is a specialized field within fluid mechanics that utilizes numerical methods and algorithms to resolve and analyze fluid flow-related problems. It has become a potent tool for simulating and comprehending the behavior of fluids, encompassing both liquids and gases, across various engineering and scientific contexts. This in-depth overview explores the fundamental aspects of CFD, including its historical development, underlying principles, numerical methodologies, applications, and challenges.

The origins of CFD can be traced back to the mid-20th century when computers emerged as robust tools for scientific calculations. The evolution of CFD was significantly influenced by the increasing computational capabilities and the demand for more precise and efficient methods to scrutinize fluid flows. Early contributors to the field include Lewis Fry Richardson, known for his pioneering work in numerical weather prediction during the 1920s, and John von Neumann, whose efforts during World War II contributed to the development of numerical methods for fluid dynamics.

Before investigating CFD, comprehending the fundamental principles of fluid dynamics is essential. This branch of science involves the study of fluid motion and the forces acting upon fluids. The foundation of fluid dynamics lies in the Navier-Stokes equations, which delineate the conservation of mass, momentum, and energy for fluid flow. However, these complex equations often defy analytical solutions for practical engineering predicaments.

CFD attempts to provide numerical solutions to the Navier-Stokes equations, enabling the simulation and analysis of fluid flow. Core CFD principles encompass discretizing the fluid domain into a grid or mesh and iteratively solving the governing equations in discrete spatial and temporal steps. This process transforms the continuous equations of fluid dynamics into a set of algebraic equations solvable through numerical methods [68].

Various numerical methods are employed in CFD to approximate solutions to the governing equations. Prominent approaches include the Finite Difference Method (FDM), Finite Volume Method (FVM), and Finite Element Method (FEM) [69].

The Finite Difference Method (FDM) involves discretizing governing equations using finite differences to approximate derivatives. The domain is partitioned into a grid, and equations are solved at discrete points within this grid.

Finite Volume Method (FVM) centers on conserving mass, momentum, and energy within control volumes. Equations are integrated over these volumes, transforming partial differential equations into algebraic equations.

Finite Element Method (FEM): FEM divides the domain into elements, representing small portions of the overall geometry. Governing equations are then solved for each element, and solutions are amalgamated to obtain the general system behavior.

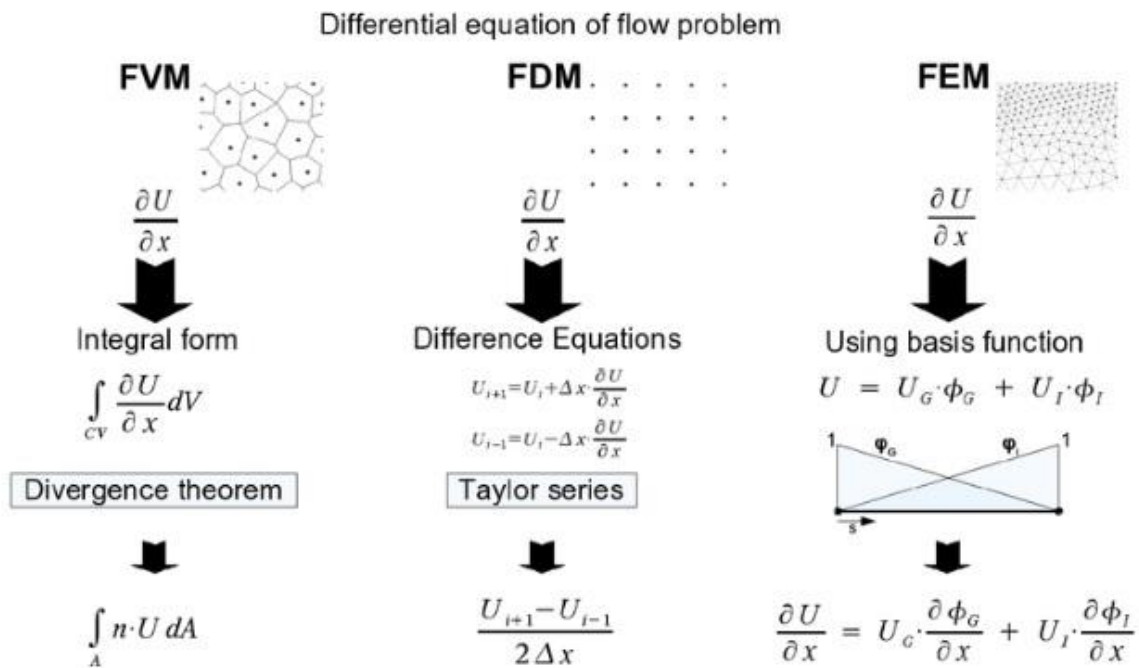


Figure 2 – Difference between CFD methods [70].

The CFD simulation process encompasses several integral stages. Firstly, in problem definition, the physical problem is clearly articulated, detailing aspects like geometry, boundary conditions, and fluid properties. Following this, mesh generation

involves creating a grid that discretizes the physical domain into small elements or cells, with the simulation's accuracy hinging on mesh quality. Subsequently, the modeling phase entails selecting appropriate mathematical models to represent fluid behavior, covering aspects such as turbulence, heat transfer, and species transport. Solving involves the iterative application of numerical methods to solve discretized equations, deploying algorithms to compute flow variables within each cell. Post-processing follows, wherein results are analyzed and visualized with standard tools, including contour plots, vector plots, and quantitative measures like pressure and velocity profiles. The validation step ensures accuracy and reliability by comparing CFD results with experimental data or analytical solutions.

CFD has widespread applications across diverse industries, impacting various engineering systems' design, optimization, and analysis. Notable applications include aerospace (aerodynamics optimization for aircraft and spacecraft), automotive (aerodynamics optimization, efficient cooling system design, and fuel efficiency simulation), energy (designing wind turbines, optimizing heat exchangers, and analyzing fluid flow in nuclear reactors), biomedical (studying blood flow, respiratory airflow, and drug delivery systems), environmental engineering (analysis of pollution dispersion, waste treatment process optimization, and simulation of natural phenomena), and civil engineering (wind load analysis on buildings and bridges, HVAC system optimization, and simulation of natural disasters).

Despite its effectiveness, CFD presents challenges and limitations. Mesh generation can be challenging and time-consuming, requiring accuracy in representing geometry and capturing flow features. Achieving numerical stability, especially for complex flows and turbulent regimes, demands careful consideration of time-step sizes and convergence criteria. Turbulence modeling introduces uncertainties, necessitating the careful selection of appropriate models. Validation against experimental data is crucial but challenging due to the need for reliable data. High-performance computing resources are often essential for computationally intensive simulations, limiting accessibility. Integrating CFD with other engineering disciplines, such as structural analysis, remains challenging yet vital for a comprehensive understanding of complex systems.

3.2. Theoretical background of CFD fire modeling

Every phenomenon can be described by mathematical equations, with no exception to fire. Usually, CFD software would use the equations representing the conservation of mass, momentum, and energy for transient flows in Cartesian coordinates so that it would assume the following form [68]:

3.2.1. Mass conservation

For any flow situation, the flow field should satisfy the mass continuity equation given by:

$$\frac{\partial \rho}{\partial t} + \text{div}(\rho \underline{u}) = 0, \quad (1)$$

Where ρ is density, \underline{u} is velocity, t is time.

3.2.2. Momentum conservation

The equation governing the conservation of momentum in the three coordinate directions is as follows:

$$\frac{\partial(\rho h)}{\partial t} + \text{div}(\rho \underline{u} u_i) = -\frac{\partial P}{\partial x_i} + \text{div}(\mu_{\text{eff}} \text{grad} u_i) + S_{u_i}, \quad (2)$$

where u_i is the velocity in the x, y, and z directions, P is the pressure, S_{u_i} i-th variable/model source term, x_i is the i-th coordinate direction, and μ_{eff} is effective viscosity.

3.2.3. Energy conservation

To ensure energy conservation, enthalpy is addressed in the form of the equation, as expressed by:

$$\frac{\partial(\rho h)}{\partial t} + \text{div}(\rho \underline{u} h) = \text{div} \left\{ \left(\frac{k}{c_p} + \frac{\rho v_i}{\sigma_t} \right) \text{grad}(h) \right\} + S_h, \quad (3)$$

Where h is enthalpy, k is conductivity, c_p is specific heat capacity, and the temperature is evaluated from the expression:

$$T = \frac{h}{c_p}, \quad (4)$$

where T stands for temperature.

3.2.4. Turbulence model

Turbulence refers to the chaotic and disordered motion of fluid particles or flow patterns within a fluid. Turbulence is characterized by irregular fluctuations in velocity, pressure, and other flow properties. It occurs at high Reynolds numbers, a dimensionless parameter used to describe the flow regime of a fluid.

Turbulence is a complex phenomenon that arises due to the interaction of various scales of motion within a fluid flow. These scales of motion range from very small, microscopic eddies to larger, more organized vortices.

To include the effect of turbulence in the simulation, the buoyancy-modified two-equation k -epsilon (k - ϵ) turbulence model is solved, incorporating the turbulent kinetic energy equation [71]:

$$\frac{\partial k}{\partial t} + \text{div}(\rho \underline{u} k) = \text{div} \left(\left[\mu_{lam} + \frac{\rho v_t}{\sigma_k} \right] \text{grad} k \right) + P + G - \rho \epsilon, \quad (5)$$

with k kinetic energy, P turbulent production rate, G buoyancy rate, ϵ dissipation rate, μ_{lam} laminar viscosity, v_t turbulent kinematic viscosity, σ_k turbulent Prandtl number for k , and the dissipation rate equation:

$$\frac{\partial \varepsilon}{\partial t} + \text{div}(\rho \underline{u} \varepsilon) = \text{div} \left(\left[\mu_{lam} + \frac{\rho v_t}{\sigma_\varepsilon} \right] \text{grad} \varepsilon \right) + \frac{\varepsilon}{k} [C_{1\varepsilon} (P + C_3 \max(G, 0)) - C_{2\varepsilon} \rho \varepsilon], \quad (6)$$

where $C_{1\varepsilon}$ is turbulent constant, C_3 is turbulent constant, $C_{2\varepsilon}$ turbulent constant, σ_ε is turbulent Prandtl number for ε , and where P represents the turbulent production rate:

$$P = 2\rho v_t \left\{ \left(\left[\frac{\partial u}{\partial x} \right]^2 + \left[\frac{\partial v}{\partial y} \right]^2 + \left[\frac{\partial w}{\partial z} \right]^2 \right) \right\} + \rho v_t \left\{ \left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} \right)^2 + \left(\frac{\partial u}{\partial z} + \frac{\partial w}{\partial x} \right)^2 + \left(\frac{\partial w}{\partial y} + \frac{\partial v}{\partial z} \right)^2 \right\}, \quad (7)$$

and G is the turbulence buoyancy term, which is given by:

$$G = -\beta g \rho v_t \frac{\partial T}{\partial y}, \quad (8)$$

or:

$$G = g v_t \frac{\partial \rho}{\partial y} \quad (9)$$

where g is gravity and β is expansion coefficient:

$$\beta = -\frac{1}{\rho} \frac{\partial \rho}{\partial T}, \quad (10)$$

and the apparent turbulent viscosity is evaluated by:

$$v_t = C_\mu \frac{k^2}{\varepsilon}, \quad (11)$$

where C_μ is turbulent constant.

The turbulence model incorporates five constants that may be subject to adjustment. In practical application, these constants are typically utilized as standard values tailored to the specific application area of the CFD model. In the context of Fire Field Modeling, the standard k- ε turbulence model adopts the initial five values for the constants presented in the table below. The last value refers to the buoyancy correction applied to the standard turbulence model.

Table 1 – Constants for the turbulence model equations

C_μ	σ_k	σ_ε	$C_{1\varepsilon}$	$C_{2\varepsilon}$	C_3
0.09	1.0	1.22	1.44	1.92	1.0

3.3. Radiation Models

When simulating fires, it is crucial to accurately show the characteristics of heat transfer (Figure 3) and energy balance in the model. The fire model encompasses two primary modes of heat transfer: convection and radiation. While convective heat transfer is addressed by the transport equations, radiative heat transfer necessitates a distinct transport sub-model. Three radiation models are widely used: (a) the Radiosity model, (b) the Six-Flux Radiation model (Figure 4), and (c) the Multiple Ray Radiation model [71].

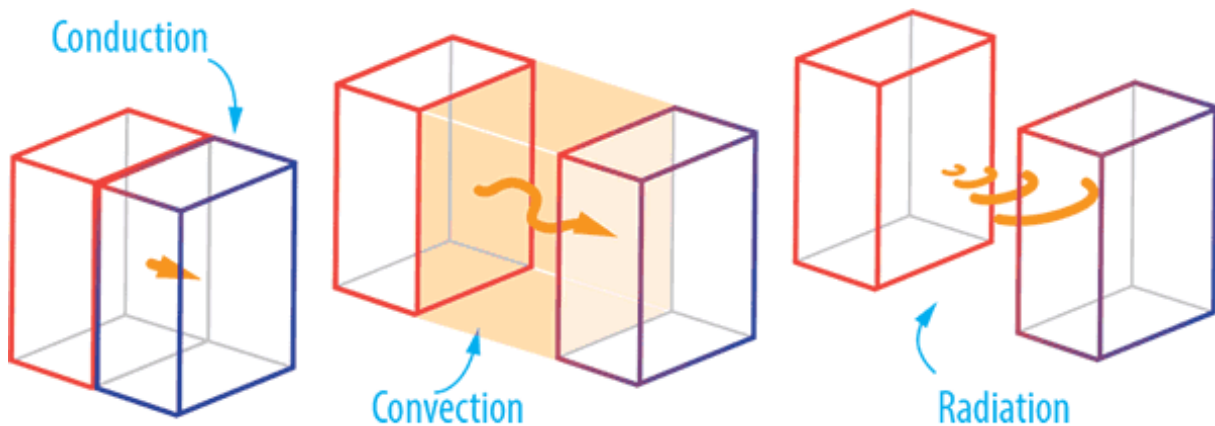


Figure 3 – Heat transfer differences [72].

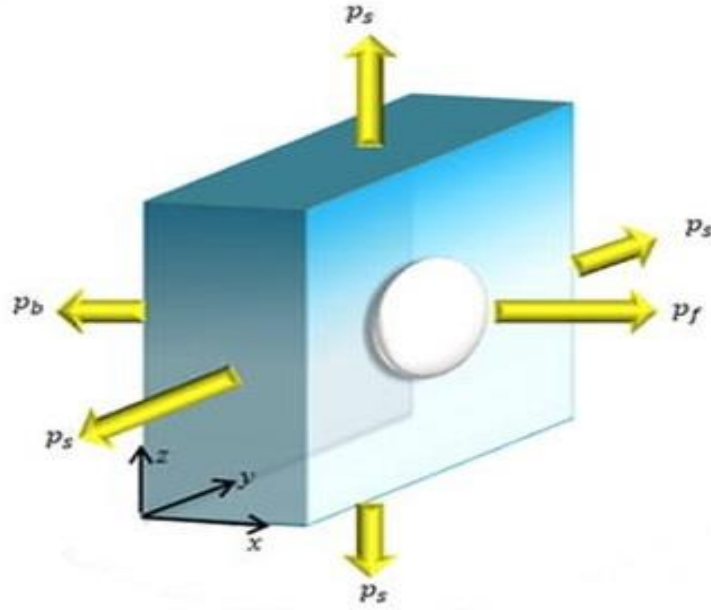


Figure 4 – Six-Flux Radiation model graphic illustration [73].

The Radiosity model is simplistic, involving the solution of a single additional variable representing radiant potential within each cell. While this ensures efficiency regarding central processing unit (CPU) time, it offers an essential representation of radiation. The modified Six-Flux radiation model solves six equations, one for each coordinate direction (both positive and negative), making the model more accurate but less efficient in terms of CPU time compared to the Radiosity model.

The Multiple Ray radiation model calculates ray intensity in several provided ray directions. This model can be computationally expensive with a large number of rays. Still, it benefits from realistic heat spreading when well-chosen ray directions are employed, surpassing the Radiosity or Six Flux radiation models in specific scenarios.

Since only the Six-Flux radiation model is used for this research because of its robustness, greater accuracy than the Radiosity model, and faster and less hardware demanding than the Multiple Ray model, it will be the only one presented in detail as follows.

3.3.1. Six-Flux Radiation Model

In the six-flux radiation model, the heat fluxes (R_i) are determined by solving additional conservation equations in each component direction, formulated as [71]:

$$\left. \begin{aligned} \frac{dI}{dx} &= -(\alpha + s)I + \alpha E + \frac{s}{6}(I + J + K + L + M + N) \\ \frac{dJ}{dx} &= +(\alpha + s)J - \alpha E - \frac{s}{6}(I + J + K + L + M + N) \\ \frac{dK}{dy} &= -(\alpha + s)K + \alpha E + \frac{s}{6}(I + J + K + L + M + N) \\ \frac{dL}{dy} &= +(\alpha + s)L - \alpha E - \frac{s}{6}(I + J + K + L + M + N) \\ \frac{dM}{dz} &= -(\alpha + s)M + \alpha E + \frac{s}{6}(I + J + K + L + M + N) \\ \frac{dN}{dz} &= +(\alpha + s)N - \alpha E - \frac{s}{6}(I + J + K + L + M + N) \end{aligned} \right\}, \quad (12)$$

where α is the absorption coefficient, s is the scattering coefficient, E is the black body emissive power of the fluid, and I, J, K, L, M , and N is the six-coordinate direction radiative fluxes.

Radiative heat transfer results in a source term in the enthalpy equation, expressed as:

$$S_{\text{six-flux}} = \alpha((I - E) + (K - E) + (M - E) + (J - E) + (L - E) + (N - E)) \quad (13)$$

3.3.2. Absorption Coefficient

The absorption coefficient is a parameter to simulate how radiation diminishes or gets absorbed when it passes through a medium that interacts with radiation through absorption, emission, and scattering processes. Participating media refers to materials that engage with radiation in these ways. This coefficient quantifies the speed at which radiation is absorbed during its journey through the medium.

When smoke concentrations are not calculated, the absorption coefficient is determined through the following piecewise linear approximation, dependent on the local temperatures [71]:

$$\begin{aligned}
 T < 50^\circ\text{C} & \quad \forall = \forall_{\text{ambient}} \\
 T > 50^\circ\text{C and } T < (T_{\text{plume}} / 2) & \quad \forall = \forall_{\text{ambient}} + \left(c(T_{\text{plume}} / 2) - \forall_{\text{ambient}} \right) / \left((T_{\text{plume}} / 2) - 50 \right) (T - 50) \\
 T > (T_{\text{plume}} / 2) & \quad \forall = cT
 \end{aligned} \tag{14}$$

When the scalar concentration of smoke is calculated, the determination of the absorption coefficient is as follows:

$$\alpha = \gamma * f_v * T \tag{15}$$

where α is the absorption coefficient, f_v is the volume fraction of smoke, T is temperature and γ is a smoke-specific constant. The γ value is set to 1200, as recommended by the software package developers.

3.3.3. Species conservation

Species conservation involves the modeling and examination of how various chemical species or elements within a fluid flow are preserved. This concept holds particular significance in simulations related to reactive flows, combustion, and chemical processes, as the distribution of these chemical species and their reactions greatly influence the simulation's results.

The conservation of any scalar quantity, f , is represented by the equation [71]:

$$\frac{\partial(\rho f)}{\partial t} + \text{div}(\rho \underline{u} f) = \text{div}(\Gamma_f \text{grad}(f)) + S_f \tag{16}$$

f is a scalar quantity and Γ_f is the f^{th} variable diffusion coefficient.

3.3.4. Combustion model

The combustion model refers to a mathematical description or a system of equations applied to imitate and examine the combustion process occurring within a fluid flow. These models find application in CFD simulations to anticipate the outcomes of chemical reactions, heat release, and various combustion occurrences, including the spread of flames and the generation of pollutants.

A basic one-step global chemical reaction is employed, namely [71],



F represents the fuel, O is the oxidant, P is the product, and s denotes the stoichiometric ratio of oxygen to fuel. The heat released per unit mass of fuel consumed is denoted by H (J/kg).

Some of the models that can be incorporated: diffusion-controlled and kinetically controlled, also known as mixing-controlled. In the diffusion-controlled combustion model, only the mixture fraction, a conserved scalar, is solved using its partial differential governing equation, while the mass fractions of fuel (m_f), air (m_a), and product (m_p) are calculated by:

$$m_f = (f - f_s) / (1 - f_s), \quad m_a = 0, \quad m_p = 1 - m_f, \quad \text{if } f > f_s, \quad (18)$$

$$m_a = 1 - f / f_s, \quad m_f = 0, \quad m_p = 1 - m_a, \quad \text{if } f < f_s, \quad (19)$$

Where f is mixture fraction, f_s is the stoichiometric value of mixture fraction, m_f is fuel mass fraction, m_a is air mass fraction, and m_p is product mass fraction.

In the kinetically controlled model, two extra scalar governing equations are incorporated for the fuel mass fraction (m_f) and the mixture fraction (f). The air mass fraction (m_a) and the product mass fraction (m_p) are determined algebraically using the following equations:

$$m_a = 1 - m_f - (f - m_f) / f_s, \quad (20)$$

$$m_p = 1 - m_f - m_a. \quad (21)$$

Instead of differential equations, where f_s represents the stoichiometric value of f , as defined by

$$f_s = 1 / (1 + s). \quad (22)$$

The mixture fraction is a conserved scalar, so its governing equation lacks a source term. The governing equation for the mass fraction of fuel includes a source term that incorporates the eddy dissipation concept, specifically,

$$R_f = A \min(\bar{C}_f, \bar{C}_o / s) \frac{\varepsilon}{k}, \quad (23)$$

where R_f is the fuel consumption rate (kg/s), A is a constant, and min denotes the minimum of two numbers. \bar{C}_f represents the time-averaged fuel concentration, \bar{C}_o stands for the time-averaged oxidant concentration, and k and ε are the turbulent kinetic energy and turbulence dissipation rate, respectively. The default value of A is 4, based on the software developers recommendation. Given the assumption of compressible flow, the ideal gas law is applied:

$$\rho = \frac{pW_g}{RT} \quad (24)$$

where ρ is density, R is the universal gas constant, p is pressure, T is temperature, and W_g is the molecular weight of the gas mixture, given by:

$$1 / W_g = m_f / W_f + m_a / W_a + m_p / W_p, \quad (25)$$

where W_f , W_a and W_p are the molecular weights of the fuel, air, and products, respectively.

The static enthalpy (h) is calculated using:

$$h = \left(\int_0^T c_p(T') dT' - h_0 \right) + m_f H \quad (26)$$

where $c_p(T')$ is the specific heat of the gas mixture, T is temperature, and

$$h_0 = \int_0^{T_{ref}} c_{pa}(T') dT' \quad (27)$$

c_{pa} is the specific heat of the air and

T_{ref} is a reference temperature at which the static enthalpy of the air is zero. The specific heat of each component is presumed to follow a polynomial function of temperature (T) formulated as

$$c_p = a + bT + cT^2 + dT^3 \quad (28)$$

The specific heat of the gas mixture is defined by

$$c_p = m_f c_{pf} + m_a c_{pa} + m_p c_{pp} \quad (29)$$

where c_{pf} , c_{pa} and c_{pp} represent the specific heat of the fuel, air, and the products, respectively. For simplicity, a constant specific heat is applied to all components, meaning $c_{pf} = c_{pa} = c_{pp}$, and $b = c = d = 0$.

3.4. Auxiliary equations

The subsequent auxiliary equations compute vital derived quantities essential for the simulation results.

3.4.1. Density

The density is determined through the application of the Ideal gas law, expressed as:

$$\rho = PW / RT, \quad (30)$$

where P is pressure, W is molecular weight, R is Universal gas constant, and T is temperature.

3.4.2. Buoyancy

The subsequent "BOUSSINESQ" equation (a partial differential equation that describes fluid flow behavior, specifically the density variations in a fluid due to temperature gradients. It is named after the French physicist Joseph Valentin Boussinesq [74]) and is utilized for computing the Buoyancy source term within each control volume:

$$B = -\alpha(T - T_{ref})\rho_{ref}Vg \quad (31)$$

where B is the buoyancy source term, α is the thermal expansion coefficient, T is the control volume temperature, T_{ref} is the reference temperature, ρ_{ref} is the reference density, V is the volume, and g is the acceleration due to gravity.

In contrast, the Non-Boussinesq approximation is defined as follows for compressible fluids (using the Ideal Gas Law):

$$B = -(\rho_{ref} - \rho)Vg \quad (32)$$

where B is the Buoyancy source term, ρ_{ref} is the reference density, ρ is the current density, V is the volume, and g is the acceleration due to gravity.

4. THEORY OF FIRE MODELLING IN VR ENVIRONMENT

VR is part of the extended reality (XR) concept, which includes augmented reality (AR) and mixed reality (MR). AR allows users to experience virtual content while viewing real surroundings, with virtual elements superimposed or blended in. Compared to VR, AR offers a closer experience of the real environment by enhancing the physical, real-world environment rather than constructing a computer-generated virtual world [75]. MR combines real and virtual worlds, creating environments where physical and digital objects coexist and interact in real-time. MR allows virtual objects to be placed in a real environment or real objects to be incorporated into a virtual world

to various extents [76]. In the field of pedestrian evacuation research, the use of AR and MR is currently limited [77], primarily due to the ongoing development of partly immersive hardware and challenges in GPS tracking of users and recognition of surroundings.

Several game engines and development platforms are commonly employed to create VR (Virtual Reality) content. Some of the most common solutions are:

- Unity3D: Unity stands out as one of the most frequently used game engines for VR development, offering strong VR support and compatibility with various VR headsets, such as Oculus Rift, HTC Vive, and Windows Mixed Reality.
- Unreal Engine: Unreal Engine, another widely adopted option, boasts exceptional graphics quality and a comprehensive VR development toolkit. Epic Games, the creator of Unreal Engine, has actively contributed to the VR industry.
- Godot Engine: Godot, an open-source game engine, supports VR development, making it an attractive choice for indie developers and those with budget constraints.
- CryEngine: Recognized for its impressive graphics capabilities, CryEngine includes VR support, enabling the creation of visually stunning VR experiences.
- Lumberyard: Amazon's Lumberyard, with VR development support, combines seamlessly with Amazon Web Services (AWS) for cloud-based capabilities.
- GameMaker Studio 2: GameMaker, a more user-friendly game development platform, also offers VR support, making it ideal for creating simpler VR experiences.
- Mozilla A-Frame: A-Frame, an open-source web framework, specializes in constructing VR experiences on the web using HTML and JavaScript.
- Roblox Studio: Roblox Studio, primarily designed for generating user-generated content in the Roblox platform, includes support for VR development, allowing developers to craft VR games and experiences within the Roblox ecosystem.
- VR-specific platforms: Some VR platforms, such as Oculus VR (Oculus Quest) and PlayStation VR, provide their own development environments and tools, such as the Oculus VR SDK and Oculus Unity Integration for Oculus Rift and Quest development.

The selection of a game engine depends on factors like project requirements, familiarity with the engine, the target VR platform, and the skill set of the development team. Each engine possesses distinct strengths and weaknesses, necessitating a thorough evaluation based on specific project needs before commencing VR development.

Unreal Engine was selected from all the choices for several reasons. First, Unreal Engine is free for non-commercial use and provides superb graphics, which is crucial in crafting immersive VR experiences. It excels in rendering highly detailed environments, realistic lighting, and impressive visual effects, enhancing the overall sense of immersion.

Unreal Engine offers a complete suite of tools and features designed specifically for VR development. It seamlessly integrates with popular VR headsets like Oculus Rift, HTC Vive, and Windows Mixed Reality, simplifying VR content creation for these platforms.

It has an active and engaged community of developers alongside a wealth of online resources, such as tutorials, forums, and documentation. This robust support ecosystem facilitates problem-solving and assistance during VR project development.

Epic Games, the entity behind Unreal Engine, maintains active involvement in the VR industry, regularly enhancing the engine with new VR-related features and optimizations to keep it competitive and solve common problems and needs.

Unreal Engine 5 offers a suite of tools capable of creating both 2D and 3D fluid simulations in real-time. These tools employ physics-based methods to authentically replicate various effects, including fire, smoke, clouds, river flows, water splashes, and ocean waves crashing onto shores.

Designed with artists in mind, this user-friendly toolset is an open platform for creative experimentation. It features simulation stages, reusable components, and powerful Data Interfaces.

For artists, achieving specific visual effects is streamlined, requiring adjustments to only a few parameters. Meanwhile, advanced users and research and development engineers have the flexibility to dissect and modify the simulators, experimenting with novel algorithms.

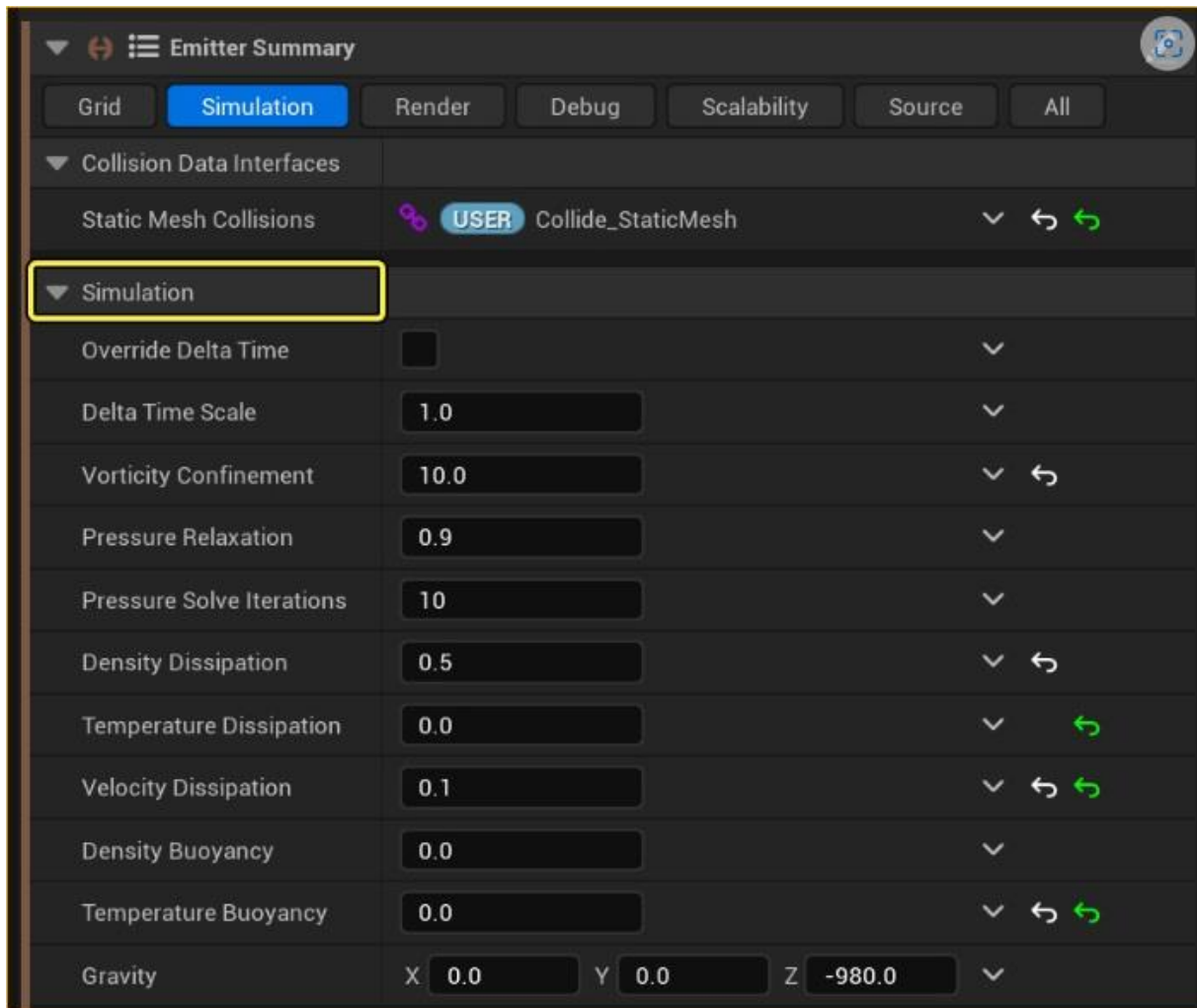


Figure 5 – Unreal Engine Niagara properties for the fluid dynamics calculations

Although no specific equations used to solve fluid dynamics can be found in Unreal Engine documentations, from the properties chosen from (Figure 5), it can be concluded that it uses general fluid dynamics equations. Still, it is crucial to note that Unreal Engine, as a gaming platform, prioritizes real-time performance and visual appeal over scientific precision. Therefore, these equations are optimized for speed, incorporating various approximations and creative adjustments to deliver visually compelling effects within acceptable computational constraints.

The fluid simulators in Unreal Engine, encompassing gas and liquid simulations, are versatile tools catering to a wide range of users, from graphics researchers to effects artists. Here are some ways different user groups can utilize these simulators:

- For effects artists:

- Effortlessly integrate systems into a map for specific elements in games or cinematic sequences,
 - Manipulate, animate, or control various user-defined parameters through Blueprints,
 - Adjust a handful of system parameters to fine-tune the visual appearance,
 - Incorporate forces and collision objects to make the fluid interact realistically with the environment,
 - Implement basic changes in the way fluid is introduced into the solver.
- For effects developers:
- Construct custom systems using pre-existing templates to achieve a specific aesthetic,
 - Integrate new forces, source methods, or unique boundary conditions,
 - Enhance and adjust existing modules in both fluid and sourcing emitters within a system,
 - Optimize performance for particular systems through profiling and adjustments,
 - Develop or adapt systems, making user parameters available to enhance artists' capabilities.
- For R&D developers:
- Create intricate fluid behaviors by writing HLSL shader code to expand the capabilities of base emitters,
 - Alter the simulation algorithm comprehensively by modifying or replacing current modules,
 - Experiment with new algorithms and test them efficiently in the Niagara Editor or within maps,
 - Develop novel systems and group parameters using User Parameters and Summary View for streamlined management and customization.

These varied applications demonstrate the flexibility and power of Unreal Engine's fluid simulation tools, enabling users from different technical backgrounds to achieve their creative and technical goals.

5. A MODEL OF A SHIP ENGINE ROOM

A model of the ship engine room was created based on actual designs and measurements of the engine room of a RO-RO ship. The drawings were provided by the design office with the permission of the shipping company. Since the ship was in the design phase, only some devices were placed in the engine room, while most of the devices were designed independently based on the experience of former chief engineers. Models of some devices were obtained from manufacturers, while the majority were independently modeled based on the available drawings.

The engine room model consists of two main four-stroke engines, each with a shaft generator and three diesel generators. The main engines are connected to the intake air system, fresh and seawater cooling systems, and the fuel system. Oil systems are integral part of the engines and are not shown externally.

The model is divided into two levels, with the main engines on the first level, showing part of the propeller shaft, auxiliary engines, and the piping systems of fresh and sea water cooling with corresponding plate coolers, pumps, and valves.

The main engine's intake air and fuel pipeline come from the upper (second) level, where two starting air compressors and one service air compressor are located, with corresponding starting and service air vessels. There are also two oil purifiers, two heavy fuel oil purifiers, and one diesel oil purifier. A pipeline connects the fuel oil purifiers to the booster module and the settling and service tanks with corresponding valves.

Additionally, for the purpose of this dissertation, it was necessary to create a ventilation system consisting of 16 ventilation ducts that operate as delivery or reversible, with the initial input that the air exchange must be at least 30 exchanges per hour [78]. The ventilation network of pipelines is branched throughout the engine

room as specific devices require air to operate and to avoid unventilated pockets. The direction and strength of the airflow significantly affect the smoke spread in case of a fire.

The VR model of the engine room was created as an educational tool with the idea that each of the corresponding devices can be disassembled to the smallest detail so that the user can see the operation principle. Currently, this can be done with the intake and service air compressors and with all types of valves placed in the engine room. The development and continuation of this approach is in progress.

The model is primarily an interactive tool with which the user can prepare and initiate the operation and experience the environment with visual and sound effects. This approach is essential for users without experience on large ships as they can feel the environment of such a profession without the real danger of moving parts, as in a real environment.

Also, an instructor can simulate a particular malfunction or emergency and observe the participants' behavior in each situation. Based on this principle, a simulation of the occurrence and spread of smoke in the existing VR engine room has also been created.

5.1. Model of engine room to be used for CFD simulation

For CFD modeling, the existing VR engine room is slightly simplified. In the VR version, there are several rooms (i.e., auxiliary engine room), but for the CFD simulation, only the main room is modeled at a size of 16.1x19.6 m (Figure 6, Figure 7, and Figure 8). It is still a two-level room, with main engines on the bottom level, and its geometry is made to be square, while in VR, there are two triangle-like add-ons on each side of the room. Those differences make less than a 1% difference in the overall area but significantly in the CFD simulation's cell economy.

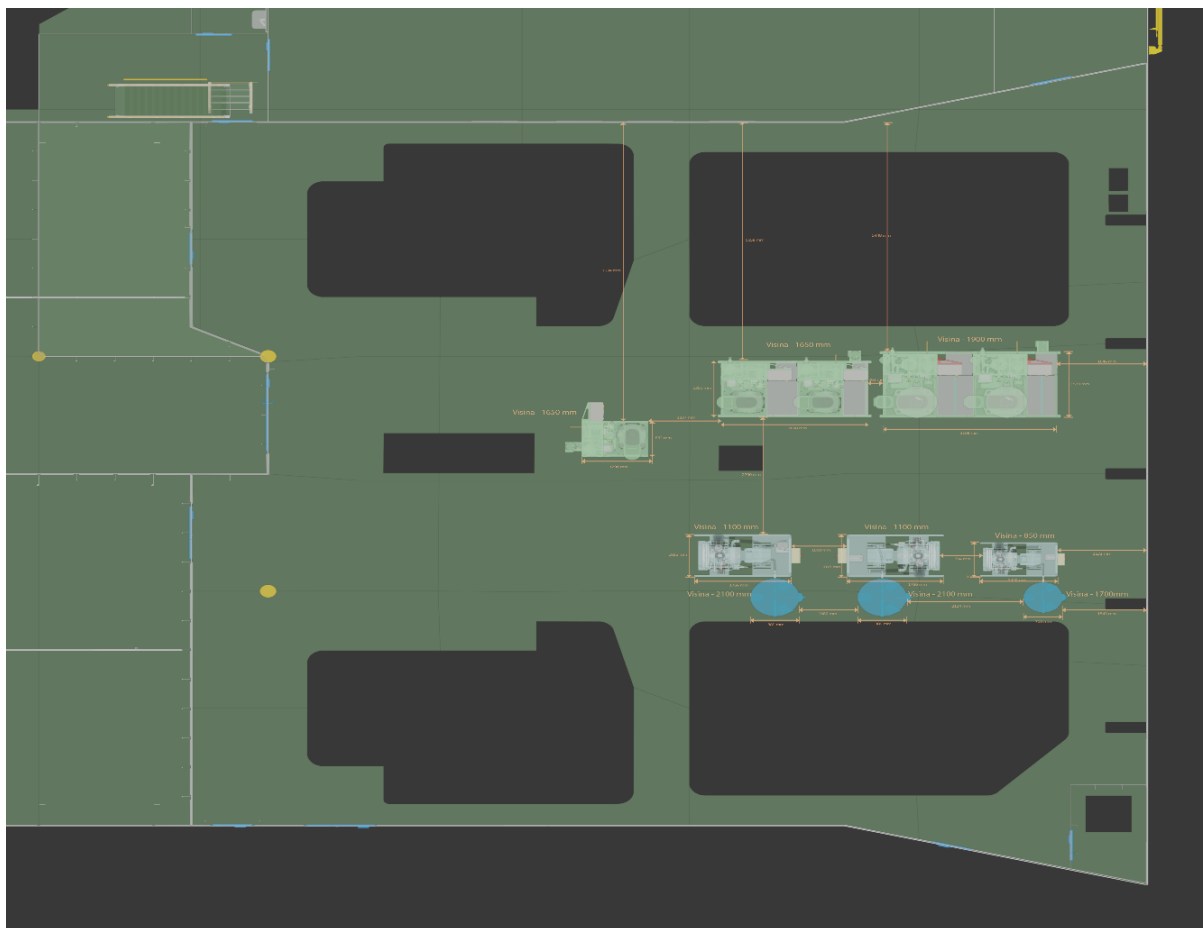


Figure 6 – Layout of the upper deck engine room.

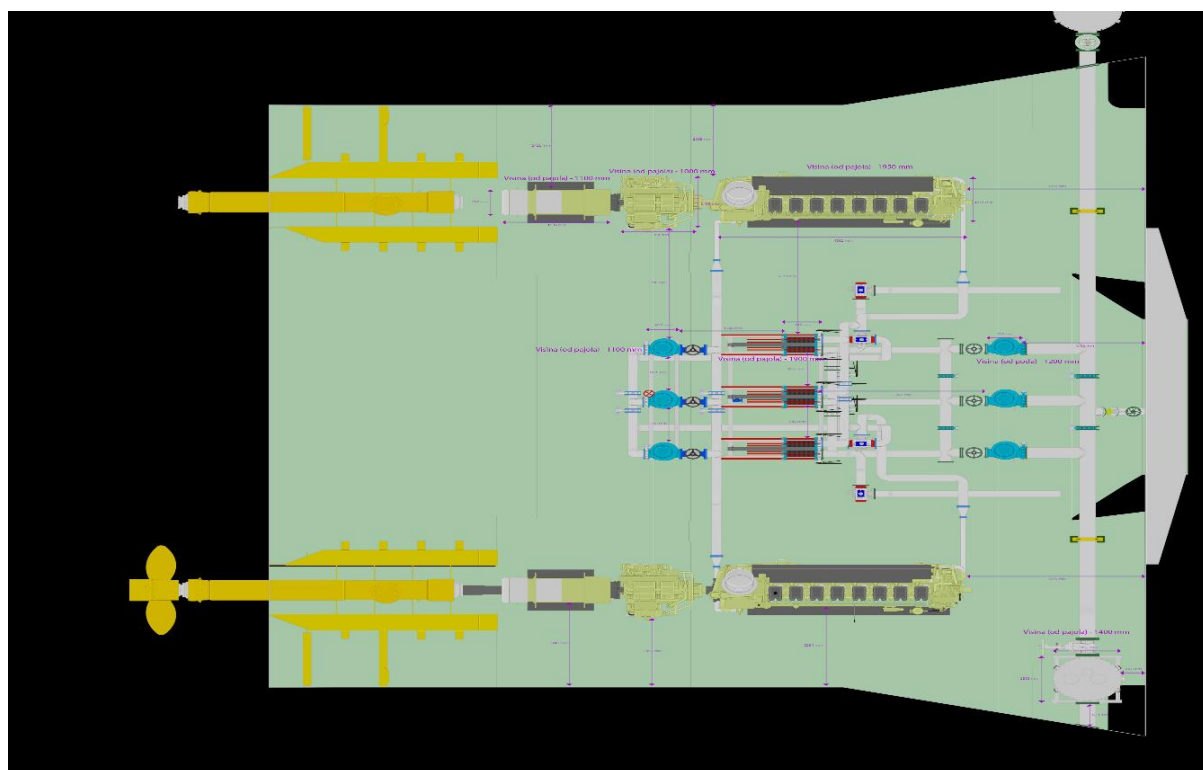


Figure 7 – Layout of the lower deck engine room.

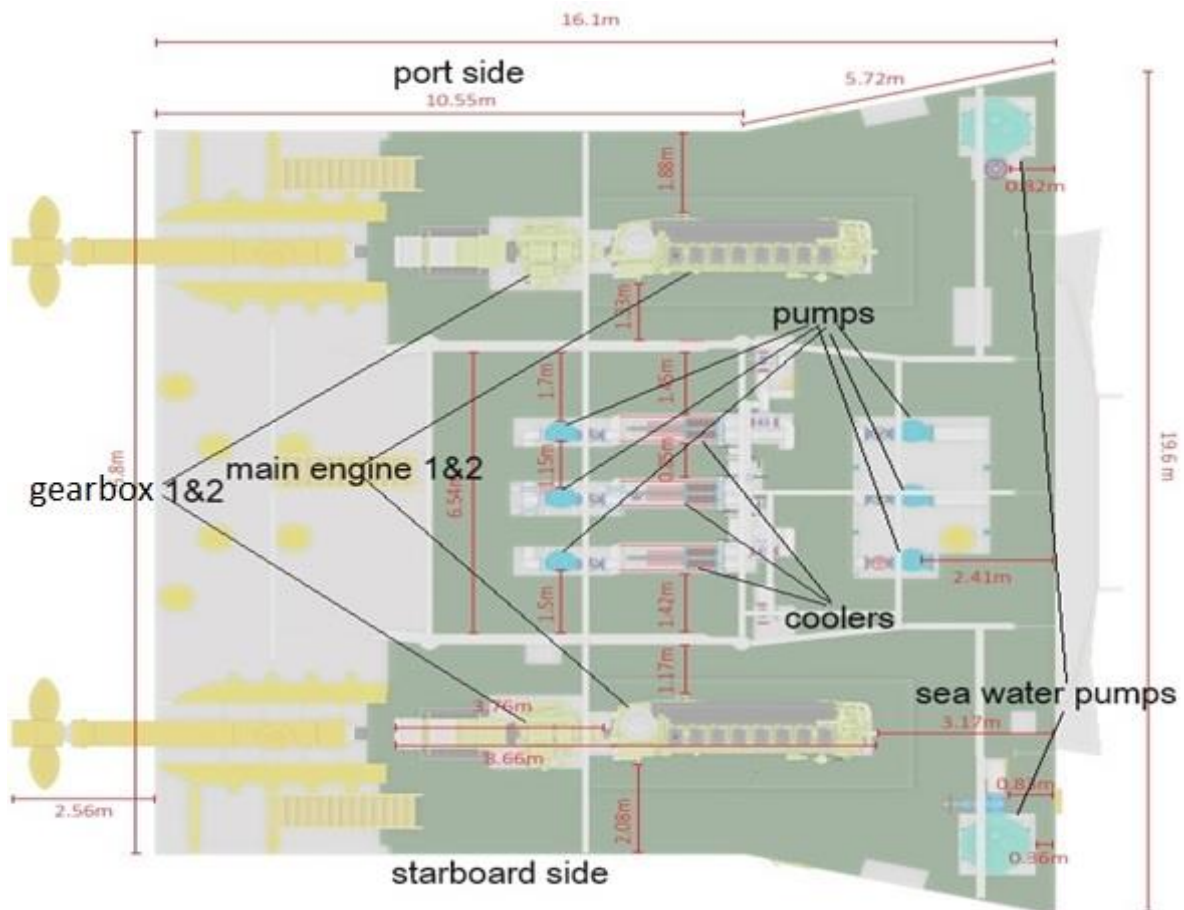


Figure 8 – General dimensions and equipment distances.

All the machines that are modeled are replaced with cuboids of similar size. Purifiers located on the upper level were not modeled since they would minimally affect the smoke, but ventilation to purifiers is modeled.

5.2. Scenarios to be run in CFD

It is decided that three scenarios will be run. Based on a former chief engineer's experience and common knowledge of fire, it is concluded that the fire would most likely start on the main engine fuel oil system, where high pressure is presented due to a leak, on the purifiers due to same reason or in a bucket where oily rags are disposed and that is left unattended anywhere inside the engine room. Those three fires are to be run in CFD.

6. CFD MODELING OF FIRE INSIDE ENGINE ROOM

6.1. Smartfire CFD software

The Smartfire software system comprises a set of interconnected tools designed to enable users to efficiently create, simulate, and analyze fire simulation scenarios with ease and reliability. Its user-friendly components and intuitive interfaces cater to experienced and novice users, facilitating the specification of fire simulation geometry and scenarios. Users can create a suitable Computational Fluid Dynamics (CFD) mesh and simulate the effects of fire scenarios over time or under steady-state conditions.

The specification tools offer significant flexibility for configuring complex simulation scenarios, and the numerical simulation engine is highly interactive, providing real-time visual and numerical displays of intermediate results and various graphs.

The Smartfire CFD software can simulate hot, turbulent, buoyant flows within a single arbitrarily sized region. This region may contain multiple internal compartments separated by walls and partitions. The software employs an unstructured control volume solution technique, although the current automated meshing system is restricted to generating regular hexahedral control volumes. Fires can be represented as volumetric heat sources (simple, time equation, or table fire defined) or as mass sources of gaseous fuel (using the eddy dissipation combustion model). Thermal radiation can be modeled using radiosity, an enhanced six-flux, or multiple-ray radiation models. Turbulence is modeled using a buoyancy-modified k-epsilon model. The general flow can be either compressible or incompressible and is modeled using one of the pressure-correction algorithms (SIMPLE or SIMPLEC).

Practical processing limitations, such as the maximum number of control volumes, are determined by the host computer's capabilities, including available memory, and the processing speed is directly related to the computer's speed.

The Smartfire software environment is a product of an ongoing project exploring intelligent control of CFD software and developing novel CFD techniques. The suite consists of four main logical components: a scenario designer for importing DXF formatted CAD drawings, a front-end case specification environment, an automated interactive meshing system, and a CFD numerical engine.

These components are used in the order presented above in a typical CFD simulation cycle. This involves creating and specifying geometrical information, generating a suitable mesh, and then simulating the problem in the CFD engine. The scenario designer aids in importing building designs from DXF CAD drawings.

The Smartfire software suite aims to assist CFD novices through its automated meshing system component. This system performs a crucial task: specifying a reasonable computational mesh for the case using the chosen cell budget. Users only need to provide the problem's geometry and physics. Once the geometry is specified, the automated meshing system generates most of the setup parameters for the CFD code, allowing users to run simulations and obtain results.

6.2. Smartfire mode of usage

Starting from Smartfire v4.0, two distinct user interfaces exist for constructing a fire modeling scenario. The SMARTFIRE Scenario Designer (smf_sd.exe) and the SMARTFIRE Case Specification Environment (smf_gui.exe) offer significantly different modes of interaction, each more suitable for specific types of problems. It is essential to highlight that the Scenario Designer cannot execute all the functions of the Case Specification Environment. Therefore, it remains vital to load the Scenario Designer geometry into the Case Specification Environment, configure it, and mesh it before proceeding with the CFD simulation.

The diagram in Figure 9 helps to indicate how to use the components most effectively:

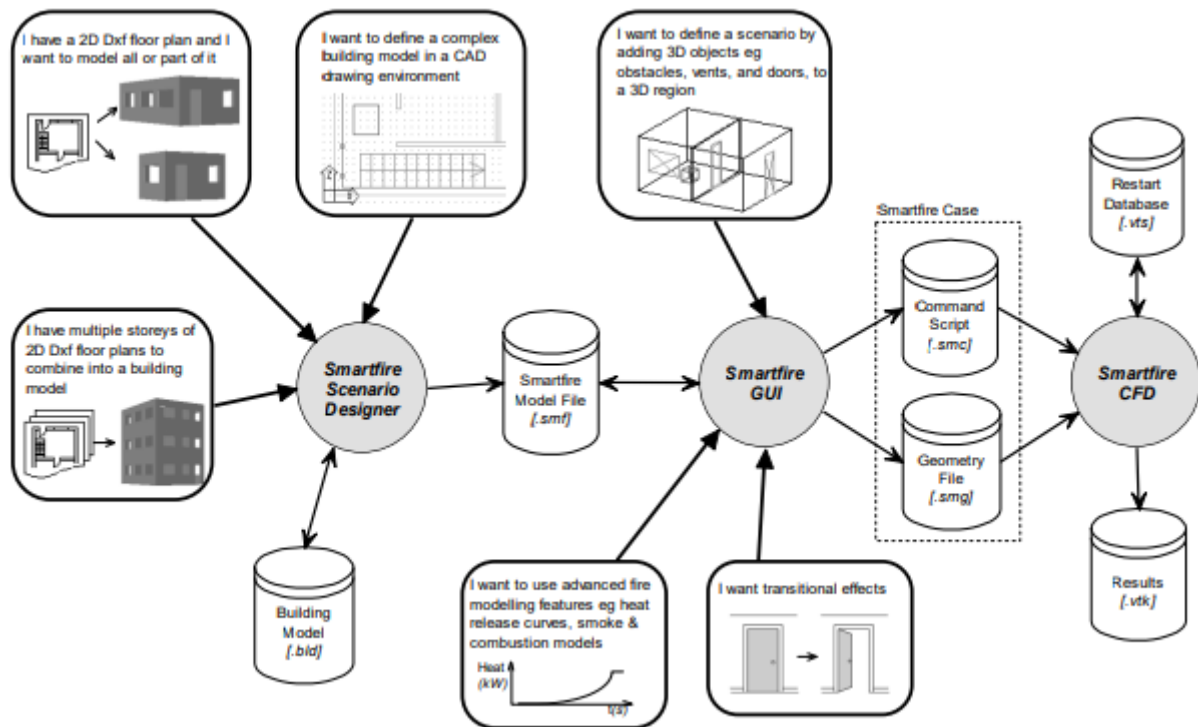


Figure 9 - How to use the Smartfire Environment [71].

To optimize software utilization, it is advisable to assess the nature of geometry. If the scenario involves detailed building floor plans (in DXF format) spanning multiple floors, the Scenario Designer is recommended for its streamlined input of such complex geometries into Smartfire. Conversely, employing the Case Specification Environment is likely more straightforward for scenarios encompassing a small set of rooms or a straightforward structure (e.g., a warehouse) without many obstacles. It's important to highlight that, regardless of how the geometry is generated, specific configuration options (e.g., sub-model activation, complex fire curves, transient and expert settings, transitional effects) and mesh creation can only be executed within the Case Specification Environment. If the geometry originates from the Scenario Designer, it must be saved and subsequently loaded into the Case Specification Environment to simulate the model.

6.3. Geometry creation using Smartfire scenario designer

Since the advanced fire spread model in the virtual engine room is the expansion of the Virtual engine room project, the existing dimensions of the virtual

engine room were used. It is created in the Unreal engine software package from shipyard plans. With lacking CAD plans, it was a challenge to create geometry that would correlate with a virtual engine room. Everything was prepared from sketches shown in Figure 6, Figure 7 and Figure 8.

Since the Smartfire Scenario Designer system is designed to streamline the utilization of 2D CAD floor plans in tandem with the Smartfire fire modeling environment, it was used for geometry creation. This system enables the construction of a building model from a CAD floor plan through a combination of manual and semi-automated tools. Generating simulation scenarios for Smartfire involves selecting and incorporating necessary objects into the building model. After exporting a scenario, users must proceed to configure simulation options, activate physics, and generate a control volume mesh (within the Smartfire Case Specification Environment) before the scenario simulation can occur.

The finished geometry is shown in the Figure 10:

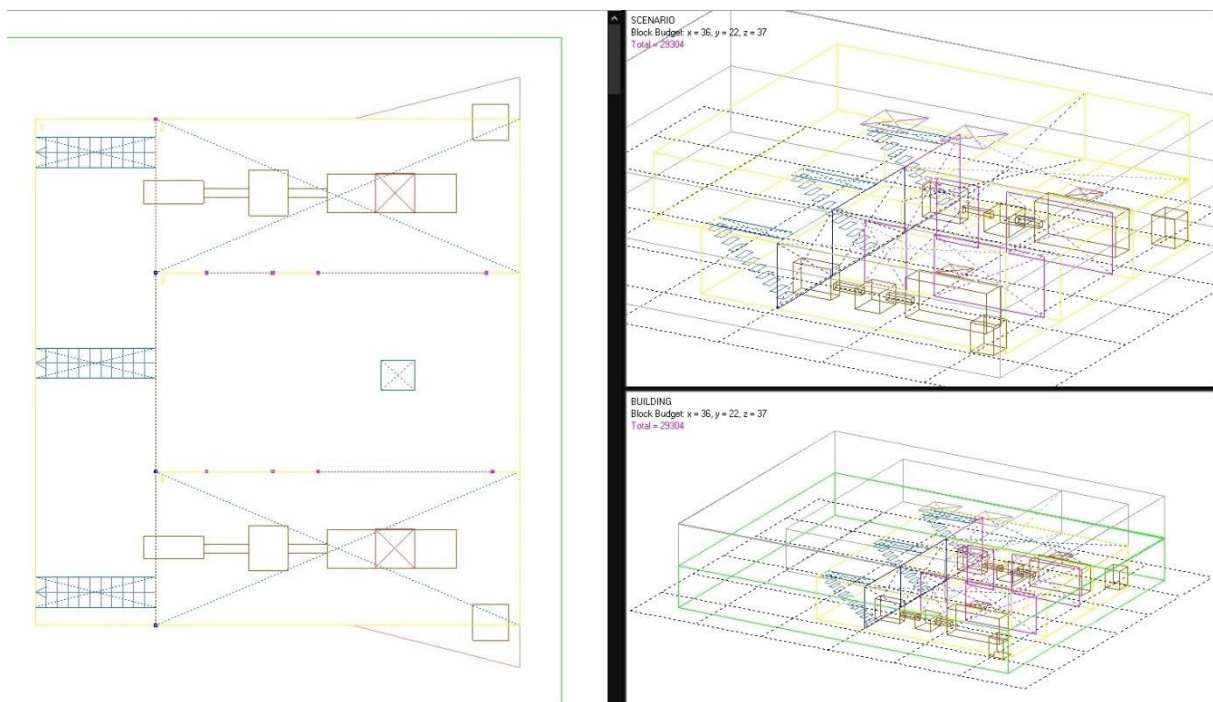


Figure 10 – Completed geometry.

Additionally, the positions of ventilation inlet/outlets are added where expected (near the engine turbo-charger suction, over purifiers, etc.; see Figure 12). Only basic properties of staircases were defined at this point (height, width, material, number of stairs...), as shown in the Figure 11:

Staircase Properties	
Z	-200
Elevation	0
Dimensions	
X Size	4000
Z Size	1000
Height	3210
Stair Width	364
Stair Height	268
General	
Number of Stairs	12
Orientation	X (-)
Style	Open
Attach To Floor	Yes
Attach To Ceiling	Yes
Material	Steel
Units: mm	
Close	

Figure 11 – Staircase properties.

6.4. Case specification using the case specification tool

The configuration of cases is accomplished through the Case Specification Environment, which features an intuitive user interface (UI). Currently, the Case Specification Environment has a limitation wherein it can define a singular computational region of arbitrary dimensions, typically resembling a box. This region may represent diverse spaces such as a single room, a building section, an entire building, or an expansive external volume encompassing multiple buildings. This geometric region allows subdivision into numerous internal compartments or rooms.

Users can also determine the physics handling for all objects, automatically or manually mesh the geometry using a specified cell budget, and initiate the Computational Fluid Dynamics (CFD) engine with the ongoing simulation case.

General arrangement inside case specification is shown in Figure 12:

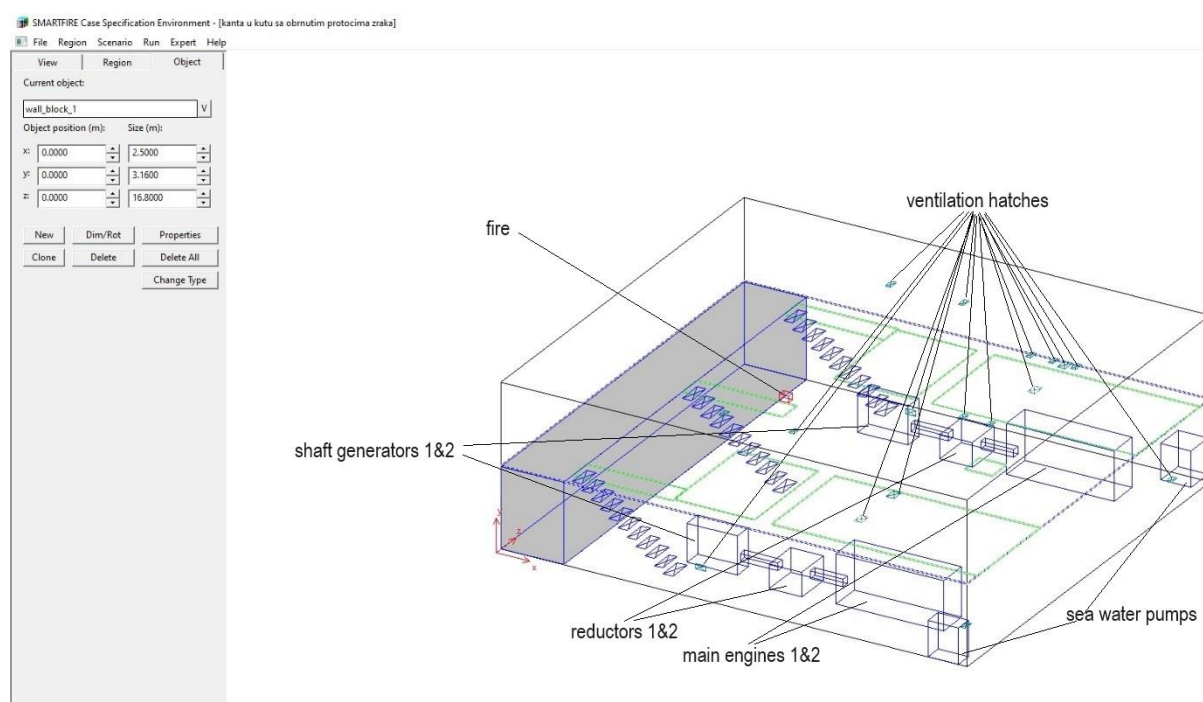


Figure 12 – Case specification, general arrangement.

When defining the geometry, the user initially outlines the maximum modeled dimensions of the computational region. Vents, inlets, and outlets can be positioned on each surface wall within this region. In this context, the term "vent" encompasses any opening to the exterior, such as a door or window. All vents are assumed to be naturally ventilated, meaning they do not have forced airflow. The user must specify the location, size, and, if different from the default behavior, the wall material of each vent. The material type is described using user-friendly terms like "brick" or "insulator", and the translation of these terms into actual material properties and boundary conditions is handled internally by the automated meshing tool.

In the current iteration of SMARTFIRE, the fire is defined as a volumetric heat source or a volumetric source of combustible gas (when utilizing the simple gaseous combustion model). The user needs to specify the physical size, location, and heat output or the mass release rate of gaseous fuel production for the fire source. The numerical "release rate" can be defined as a constant rate, an equation based on coefficients related to simulation time, or extracted from an external data file containing tabular data of release rates at specific times.

Once the user has comprehensively defined the geometry and physics for the simulation, the automated meshing tool can be initiated. This tool allows the user to select cell budgets, controlling the number of computational cells dedicated to the problem. The specified number of cells represents a trade-off between achieving accurate simulation with more fine cells and obtaining results more quickly with a lower budget of coarser cells.

After the automated meshing tool has analyzed the case and generated an acceptable mesh specification, the user can activate the Computational Fluid Dynamics (CFD) component to simulate the scenario.

6.4.1. Defining common properties for the created case

By default, all simulation issues are presumed to be steady-state (non-transient), non-turbulent flow problems devoid of heat transfer. The problem define section specifies the type of simulation to be executed, as seen in Figure 13. Simulation of interest is a transient problem, so the option is selected.

The screenshot shows the 'Problem type options' dialog box with the following settings:

- Module activation:**
 - ☒ Flow model
 - ☒ Radiation model (Setup...)
 - ☒ Smoke model (Setup...)
 - ☐ Sprinkler model (Setup...)
 - ☐ Enhanced Body Force (for fans)
 - ☒ Heat transfer
 - ☒ Combustion model (Setup...)
 - ☐ (Fire) Toxicity model (Setup...)
 - ☐ (Fire) HCl model (Setup...)
 - ☐ Gas Species Release (Setup...)
- Solution Control:**
 - Problem type: ☐ Steady state, ☒ Transient
 - Time step size (s): 0.2
 - Number of steps: 3000
 - Convergence tolerance: 1e-07
 - Sweeps per time step: 25
 - Total sim. time (s): 600.000
 - Total sim. time (h:m:s): 0h 10m 00.00s
- Default physical properties:**
 - Default wall thickness (m): 0.10000
 - Ambient temperature (K): 313.000
 - External pressure (Pa): 101325
 - Initial temperature (K): 313.000
 - Material inside the region: Standard_Air (View...)

Figure 13 – Problem define section.

It is essential to include the heat transfer option for problems related to enthalpy, such as this one. The variable solved for is enthalpy, with temperature also derived from specific heat capacity and enthalpy values. If the simulation incorporates activated flow, buoyancy values will also be computed and integrated into the flow equation. It's important to highlight that the initial enthalpy value is determined based on initial temperatures. Achieving solution convergence is significantly enhanced when realistic values are assigned to the initial temperatures of the solution domain. For instance, in enclosed spaces simulation, providing the initial mean temperature of the space is advisable.

The selection of the “no-flow” option deactivates the flow-related solvers. This implies that variables for pressure and velocity components will not undergo solving, and the turbulence solvers will be turned off as well. The default setting is to enable flow, which was suitable for the observed simulation. Several fire-related phenomena rely on active flow modeling, as it serves as the convective medium and facilitates momentum transport, driving the dissemination of various fire products.

The thermal radiation transfer model would be deactivated if the radiation option is not selected. Conversely, when the radiation option is present, it prompts the parser to examine a distinctly defined radiation section containing exhaustive information on the parameters and configuration of the radiation model intended for thermal radiation modeling. In scenarios involving heat transfer and elevated temperatures (exceeding approximately 100 degrees Celsius or 373 Kelvin), thermal radiation may become a substantial or even predominant mode of thermal energy transfer. This is attributed to radiant energy transfer, which is contingent on the temperature raised to the power of 4.

The presence of the “no combustion” keyword results in the deactivation of combustion. Smartfire currently offers two forms of a simple gaseous combustion model. The diffusion-controlled simple gaseous combustion model is engaged by using the keywords “combustion diffusion controlled”. At the same time, the eddy dissipation-controlled simple gaseous combustion model is initiated with the keywords “combustion eddy dissipation controlled3131”. Activating combustion necessitates the user to specify additional parameters throughout the case specification script, detailing the precise nature and behavior of the gaseous combustion. In this case, an eddy dissipation-controlled simple gaseous combustion model is selected, Figure 14.

Combustion model options

Combustion model

☒ Eddy Dissipation Controlled ☐ Diffusion Controlled

Molecular ratios in combustion equation

1.00000 Fuel + 2.00000 O₂ => 2.00000 H₂O + 1.00000 CO₂

Mass fractions in inlets

Fuel m-fraction: 1.00000 Oxygen m-fraction: 0.23000

Molecular weights (kg kmol⁻¹)

Fuel m-w: 16.00000 Dilutant m-w: 28.00000

Miscellaneous

Heat of Combustion (J/kg): 4e+07

Combustion efficiency: 1.00000

Eddy Dissipation constant: 4.00000

Smoke to Fuel Ratio (kg/kg): 0.015

Combustion Oxidant Limit: 0.00000

OK Defaults Cancel

Figure 14 – Combustion model options.

Since smoke is of great interest for this research and is active in the presence of active combustion, it is selected to be included in the calculation. The smoke source is derived from the combustion rate.

The extinguishing of fire, toxicity, and other options are of no interest to this research, so they are left unselected.

Considering that results will be presented in the VR simulation, a time step size of 0.2 seconds is defined. It would be of better quality to define even smaller time step, but it would also make the results too large for the existing VR hardware to handle it properly. The overall simulation time is set to 600 seconds or 10 minutes. The engine

room's enclosed space is filled with smoke at that time. With these parameters, no additional calculation is necessary. Changing the parameters of fire would call for different simulation time.

Knowing that Smartfire software may not be capable of registering objects less than 10 centimeters thick, the wall thickness of the engine room is set to 10 centimeters. The initial temperature is set to 313 K, and pressure to 101325 Pa.

The specific radiation transfer model can be selected within the define radiation section to simulate thermal radiation effects. Additionally, various radiation parameters are available to delineate the characteristics of the radiation models, such as absorption, which depends on fire type, and can be seen in Figure 15.

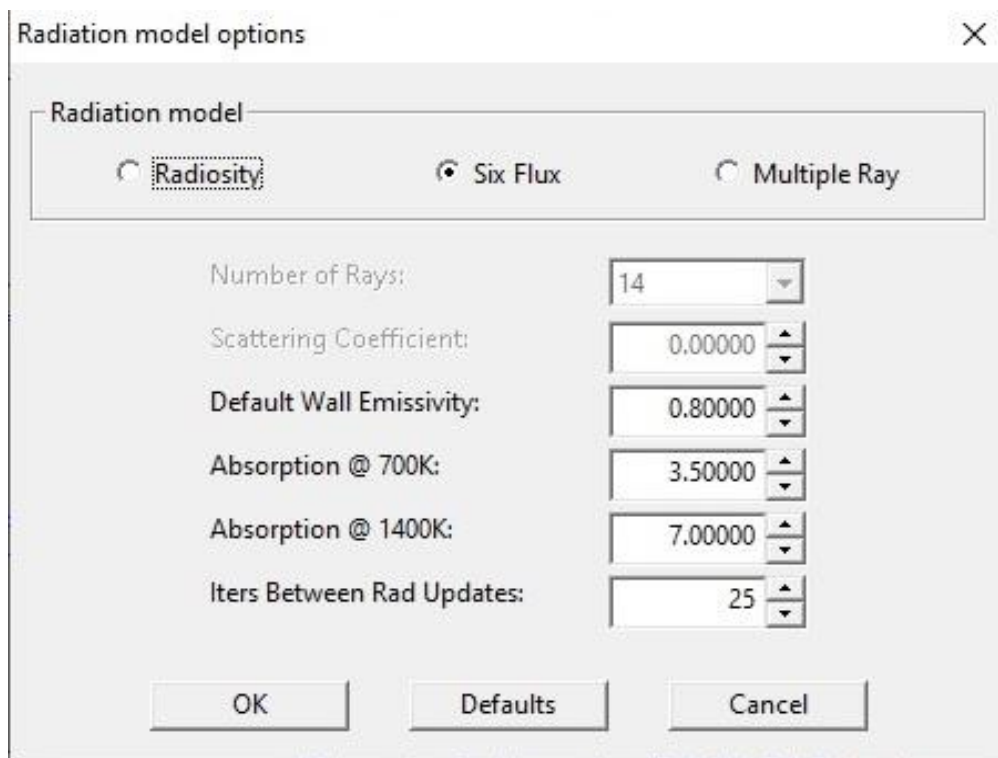


Figure 15 – Radiation model properties.

The available radiation models include “Radiosity”, “Six Flux” Radiation (enhanced), and the “Multiple Ray” Radiation model. In the “Radiosity” model, a single solved variable represents radiant intensity within each control volume. The “Six Flux” Radiation model performs radiative ray calculations in positive and negative coordinate

directions. The “Multiple Ray” Radiation model permits the user to choose any number of ray directions, and the radiation model tracks thermal radiation along each selected ray direction. Additionally, the “HR Smart” and “HR Muscl” are variations of the “Multiple Ray” Radiation model, incorporating high-resolution upwind schemes to enhance solution accuracy.

The “Six Flux” Radiation model is selected, with the note that it is exclusively compatible with fully Cartesian meshes. In contrast, Radiosity and Multiple Ray Radiation models can be applied to meshes of any type.

Except for the boundary conditions defined through geometry, a number of inlet ports is defined, as in Figure 16. At an inlet boundary condition, the assumption is that mass is either entering or potentially leaving the solution domain. The values of flow and turbulence-related variables, except for pressure, are recognized at such a boundary.

Inlet properties window (Fixed-flow surface)

Object name: INLET

Object user name: inlet_3

Physical properties at Inlet

Temperature: (K)	288.15	<input type="checkbox"/> Use Table	Load...	Editor
Flow Rate: (m^3/s)	-2	<input type="checkbox"/> Use Table	Load...	Editor
U-Velocity: (m/s)	0	<input type="checkbox"/> Use Table	Load...	Editor
V-Velocity: (m/s)	-13.3333	<input type="checkbox"/> Use Table	Load...	Editor
W-Velocity: (m/s)	0	<input type="checkbox"/> Use Table	Load...	Editor
Kinetic Energy: (m^2 / s^2)	0.355556	<input type="checkbox"/> Use Table	Load...	Editor
Dissipation Rate: (m^2 / s^3)	1.8845	<input type="checkbox"/> Use Table	Load...	Editor
Fuel fraction in inlet:	0	<input type="checkbox"/> Use Table	Load...	Editor
Mixture fraction in inlet:	0	<input type="checkbox"/> Use Table	Load...	Editor
Gas Species fraction in inlet:	0	<input type="checkbox"/> Use Table	Load...	Editor

☐ Auto-Calculate Turbulence ☐ Internal Inlet ☐ PV Inlet ☐ Neg Direction

Object Activation

Object is: Always Active (default behaviour)

Start Time: 0 End Time: 1e+20

Triggered by: ☒ Critical Change

OK Cancel

Figure 16 – Inlet properties

Air flow is determined in a way that 50% of all air conducted in the engine room is divided between two ports over the turbochargers of two main engines. All other airflow is equally divided between all other inlet ports. There are four inlet ports near both turbochargers, each facing different direction (to keep the mesh coarse, the turbocharger must be simplified), and they are taking away air, simulating air consumption of the engines. The flow rate of inlet ports over the engines is $2\text{m}^3/\text{s}$, and all other ports have a flow rate of $0.5\text{m}^3/\text{s}$.

Another important thing to choose is the solver type for different variables. There are a number of solvers (Figure 17) to choose from:

SOR	Successive Over Relaxation
JOR	Jacobi Over Relaxation
CGM	Conjugate Gradient Method
WHOLE FIELD	matrix solver
RESIDUAL SOR	Residual version of SOR
RESIDUAL JOR	Residual version of JOR
LINE SOR	Pseudo Line-by-line SOR
TDMA	Tri Diagonal Matrix Algorithm for unstructured mesh
BICG	BI Conjugate Gradient solver
CGM	Conjugate Gradient Method solver
MULTI GRID	Algebraic Multi-Grid solver (only for Pressure)
PCG MULTI GRID	Preconditioned Conjugate Gradient Algebraic Multi-Grid solver (Research solver only for Pressure)

Figure 17 – List of solvers.

The pressure variable Residual version of the Successive Over Relaxation solver is chosen (Figure 18) for its potential to efficiently solve large, sparse systems of equations [79]. For the momentum variable, the Jacobi Over Relaxation (JOR) solver is chosen for its simplicity, parallelizability, and effectiveness [80]. For all the other variables (turbulence, enthalpy, radiations, and other variables), the Successive Over Relaxation (SOR) solver is chosen for improved efficiency and convergence rate.

The image shows a 'Solver Controls' dialog box with a table of settings for different variables. The variables are Pressure, Momentum, Turbulence, Enthalpy, Radiation, and Other Vars. For each variable, there are three settings: Iterations, Solver Type, and Difference Scheme. The Solver Type for Pressure is 'RESIDUAL SOR', while for all other variables it is 'SOR'. The Difference Scheme for all variables is 'HYBRID'. The Solver convergence is set to '1e-08'. There are 'OK' and 'Cancel' buttons at the bottom.

Variable	Iterations	Solver Type	Difference Scheme
Pressure:	60	RESIDUAL SOR	HYBRID
Momentum:	5	JOR	HYBRID
Turbulence:	5	SOR	HYBRID
Enthalpy:	5	SOR	HYBRID
Radiation:	5	SOR	HYBRID
Other Vars:	5	SOR	HYBRID

Solver convergence: 1e-08

OK Cancel

Figure 18 – Solver types by variables.

6.5. Fire modelling

Last of properties to define, before creation of mesh, is fire itself. Since the ship's engine room is the environment with a high potential for fire appearance, it is decided to create three scenarios: the fire of the main engine fuel oil pipeline, fire of the purifier, and fire of residual oily rags in a bucket somewhere in the engine room because these are amongst the most common sources of fires inside the engine room [81].

6.5.1. Fire of the main engine fuel oil pipeline

Physically, fire is placed by one of two main engines (software limitations prevent fire from being located inside the object that represents the engine), with dimensions of 0.76 x 0.75 x 0.33 meters, as shown in Figure 19.

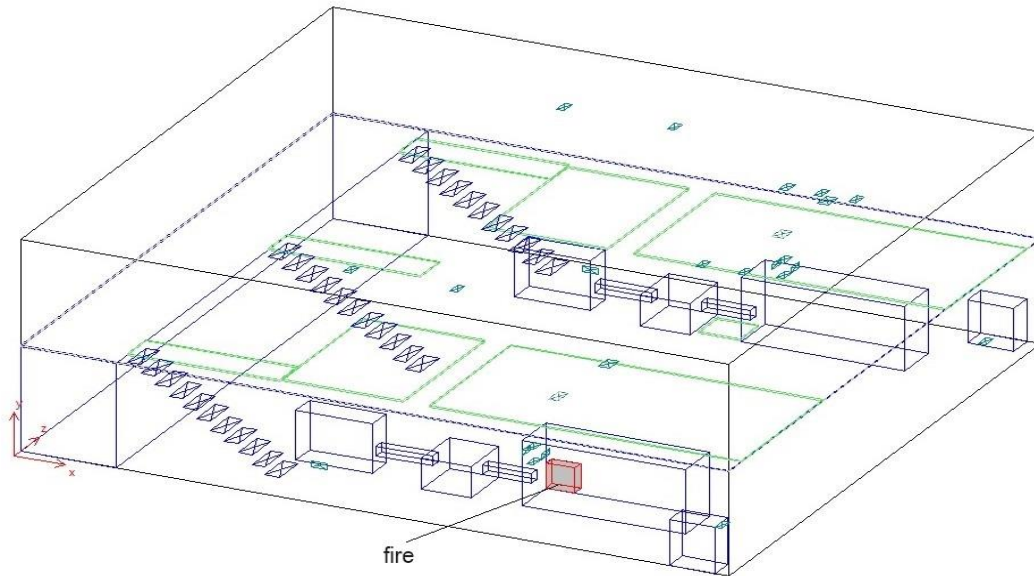


Figure 19 – Fire of main engine – fire location.

Fire is defined as simple fire (fire that can be defined as a function of simulation time squared, Figure 20), and fuel generation is calculated by the expression:

$$P = Ct^2$$

where C is the constant of $6e^{-7}$

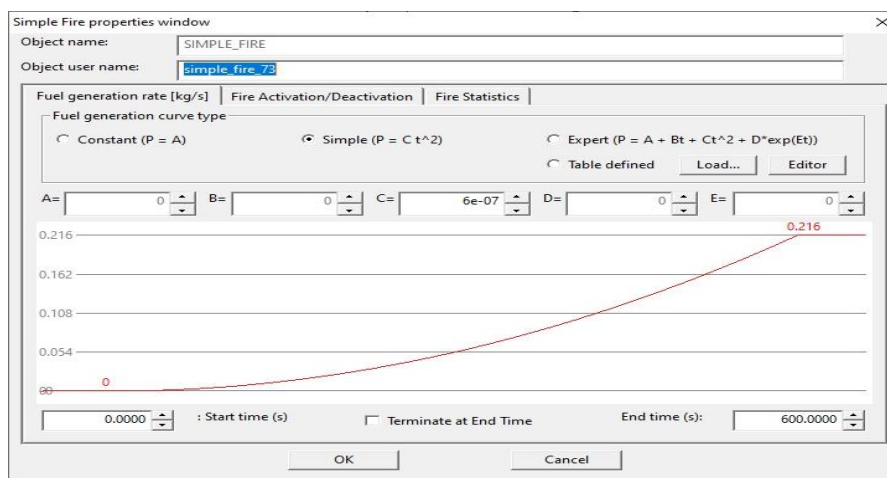


Figure 20 – Fire of the main engine fuel oil pipeline - defining fire properties.

which results in a peak fuel output of 0.215568 kg/s and a total fuel that enters the reaction of 43.1352 kg. That quantity of fuel, with fuel heat combustion of 40 MJ/kg, generates a peak heat output of 8622.73 kW, and total heat output of 1725.41 MJ, as shown in Figure 21.

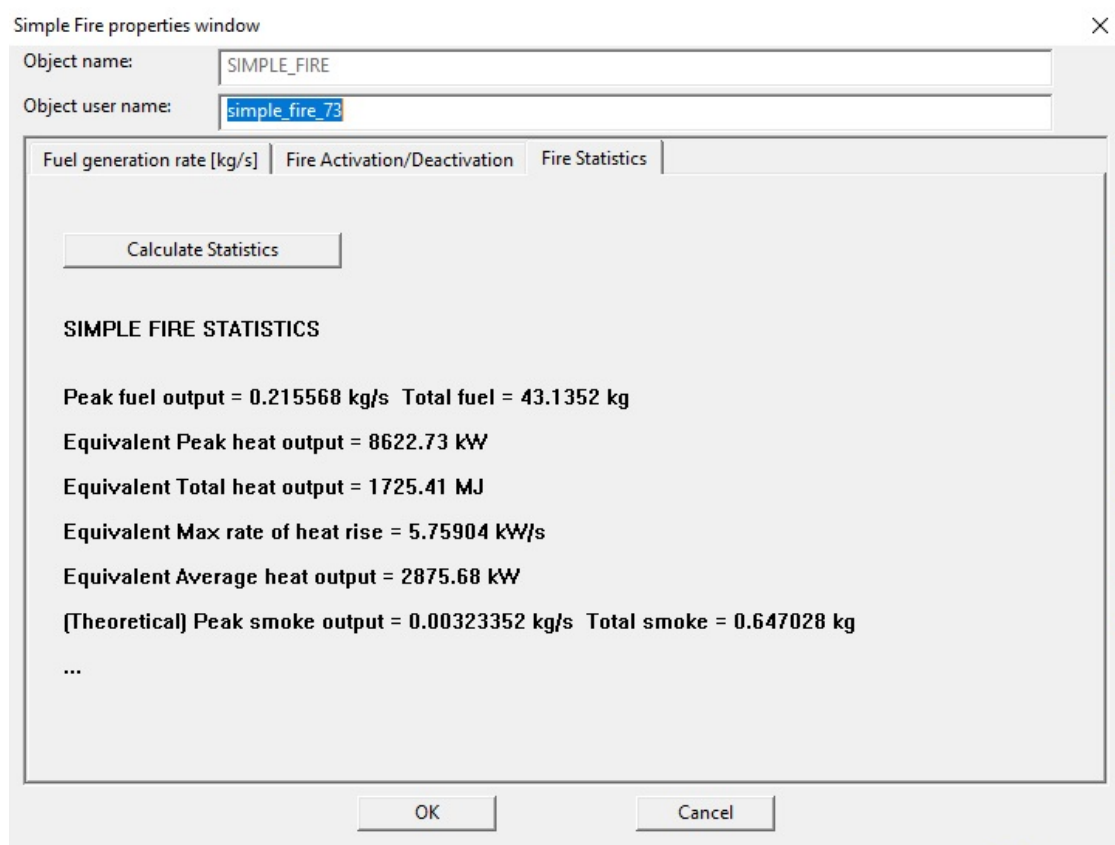


Figure 21 – Fire of main engine – Fire statistics.

6.5.2. Fire of the fuel oil purifier

Since purifiers are located on the upper deck of the engine room, fire is located on a deck above the main engines and under the purifier ventilation, as seen in Figure 22:

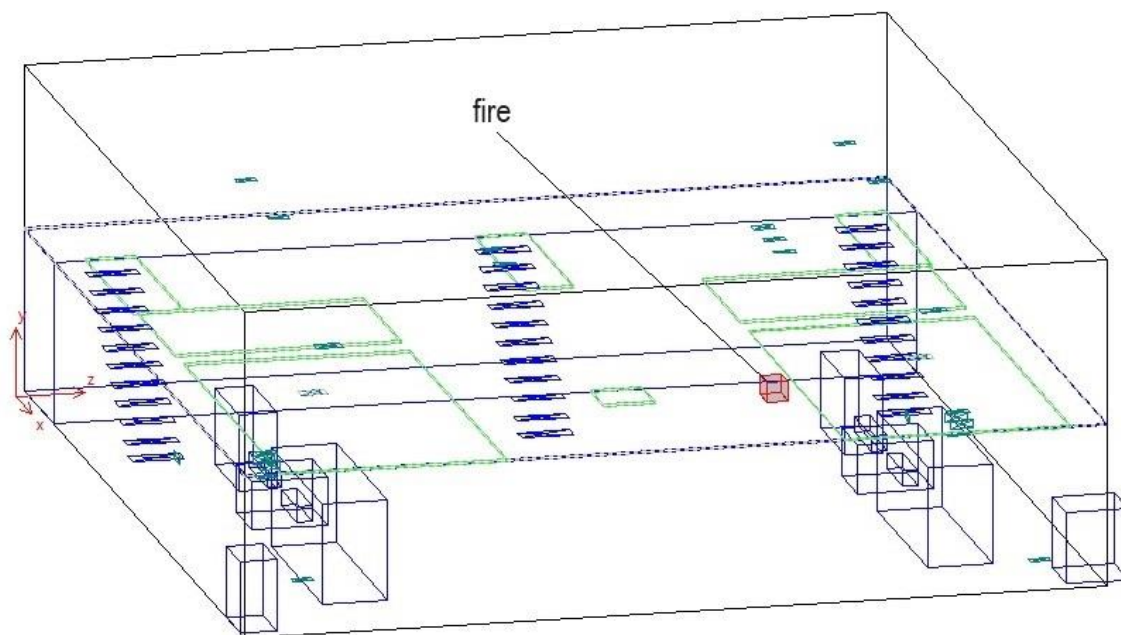


Figure 22 – Fire of the purifier – Fire location.

The fire dimension for this scenario is 0.56 x 0.37 x 0.43 meters.

Fuel properties for this scenario are set to constant ($P = A$) quantity over time, with a generation rate (A) of 0.2 kg/s, as shown in Figure 23.

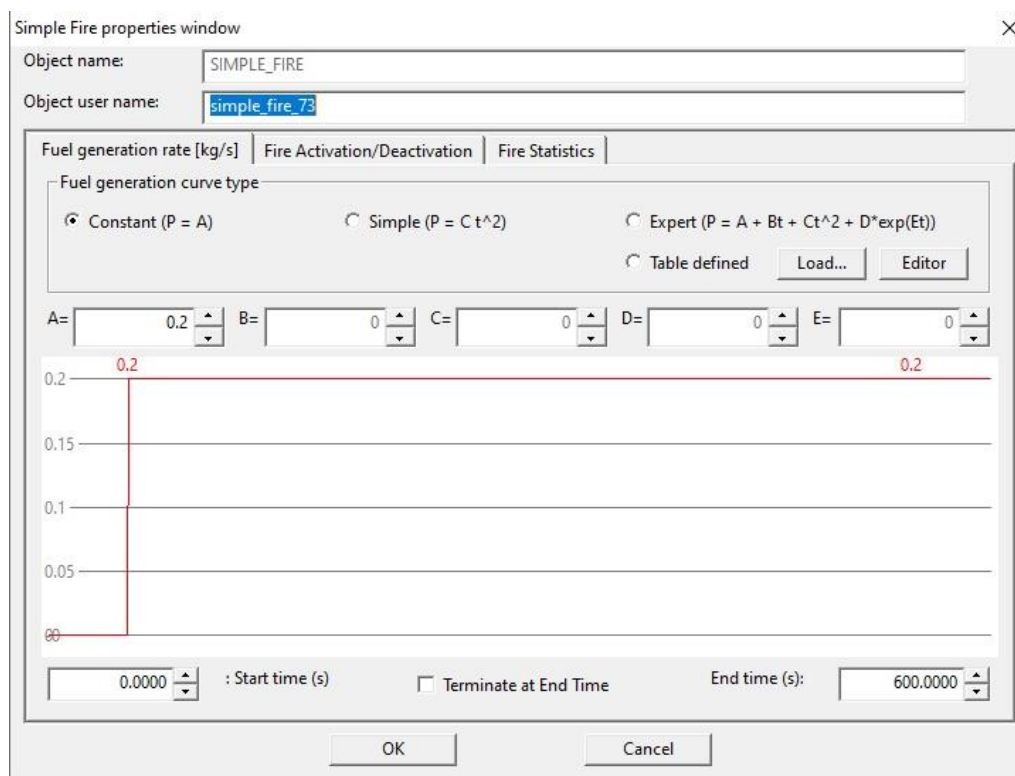


Figure 23 – Fire of the purifier – Fire properties.

That results in a peak fuel output of 0.2 kg/s and a total fuel that enters the reaction of 120 kg. That quantity of fuel generates a peak heat output of 8000 kW, and total heat output of 4800 MJ (Figure 24).

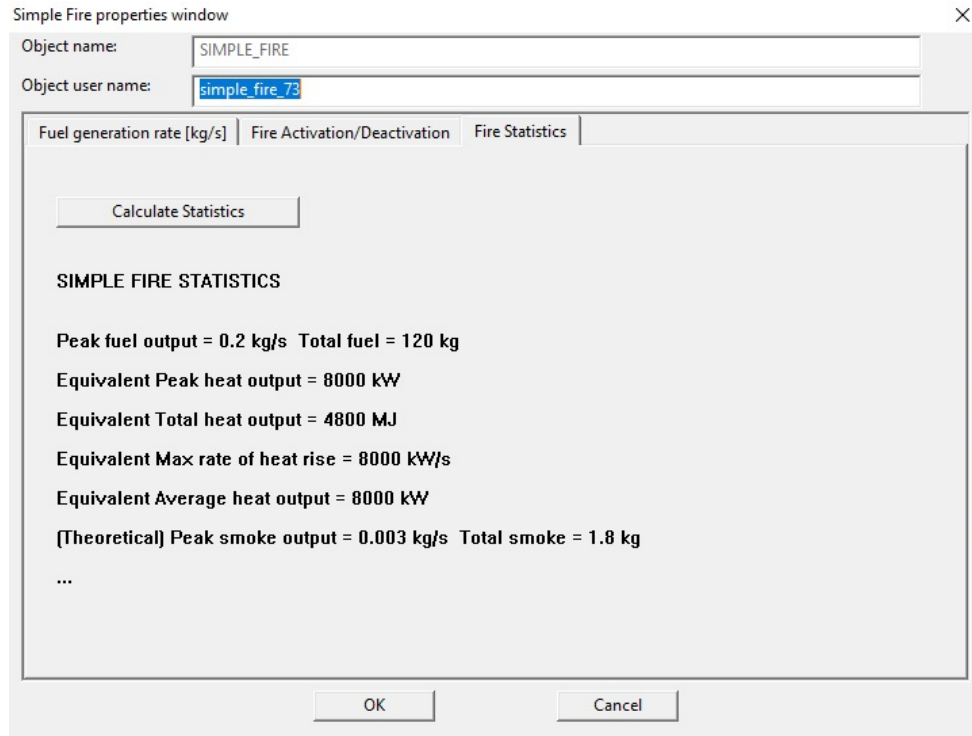


Figure 24 – Fire of the purifier – Fire statistics.

6.5.3. Fire of oily rags left in a bucket

There is always the possibility of some leftover oily rags catching on fire, so it is modelled as a “bucket of fuel” randomly located in the engine room.

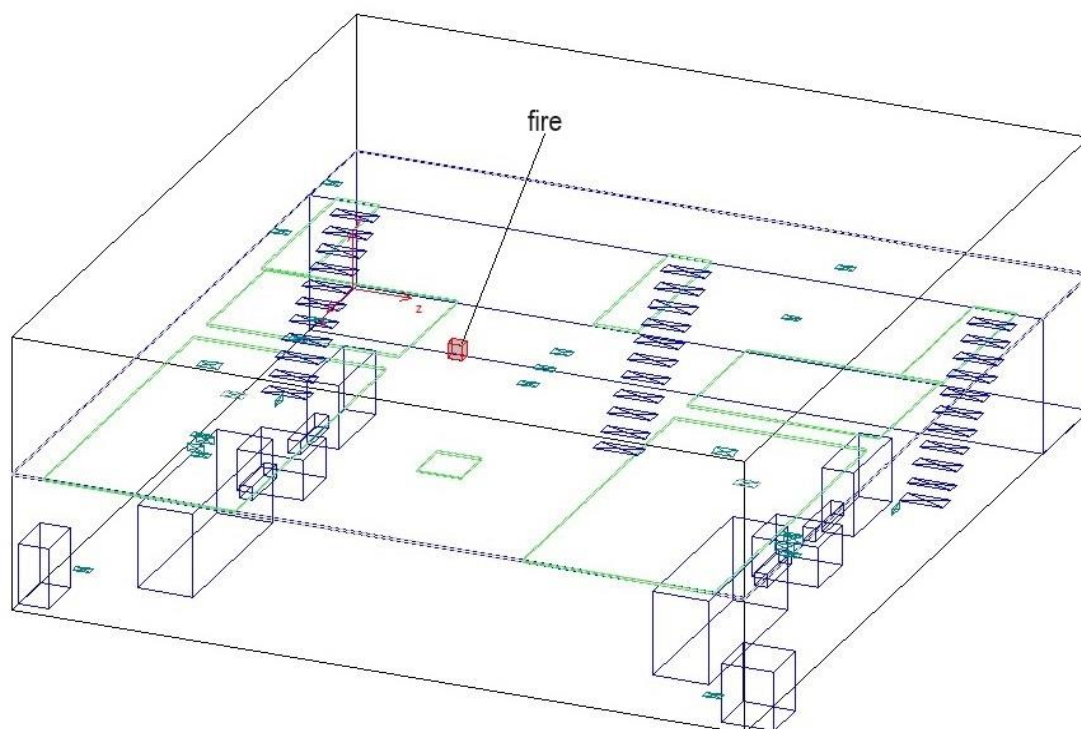


Figure 25 – Fire of oily rags left in a bucket – fire location.

Fuel entering the reaction is defined as constant (Figure 26), once again ($P = A$), and set to 0.05 kg/s, and fire dimensions are set to 0.26 x 0.35 x 0.3 meters (Figure 25).

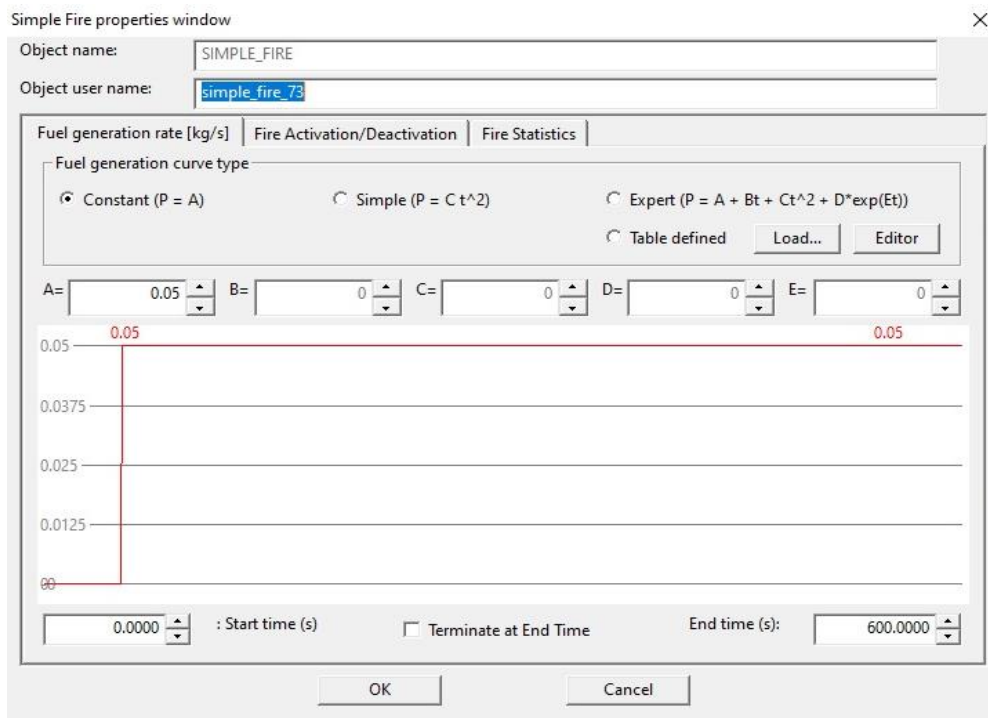


Figure 26 – Fire of oily rags left in a bucket – fire properties.

That fuel generation leads to a peak fuel output of 0.05 kg/s and total fuel that enters the reaction of 30 kg shown in Figure 27. That quantity of fuel generates a peak heat output of 2000 kW, and total heat output of 1200 MJ.

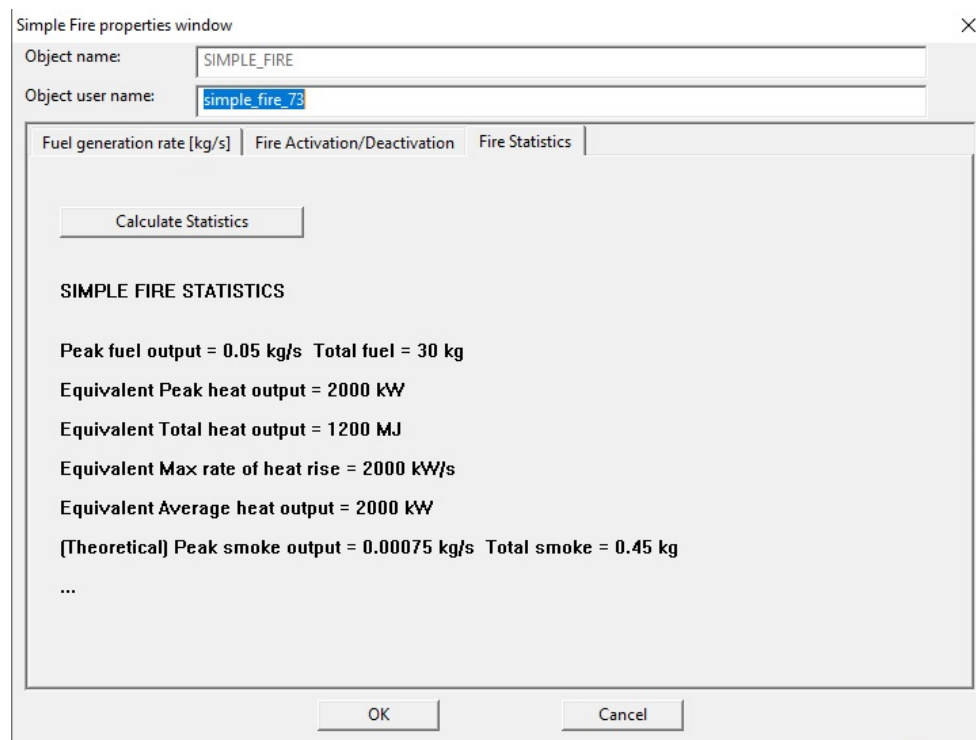


Figure 27 – Fire of oily rags left in a bucket – fire statistics.

6.6. Mesh generation

Since the scenario is going to be calculated on the multi-core machine (Intel i9-10980XE @ 3.00GHz, 128Gb of RAM), the mesh is prepared for parallel computing using Smartfire interactive meshing system.

6.6.1. Smartfire interactive meshing system

The Interactive Meshing System for Smartfire is an automated tool for mesh specification, featuring an integrated manual mesh editing capability. Its automated routines are designed to generate meshing solutions for various simulation scenarios. Positioned within the Smartfire Case Specification Environment, the tool is accessed after completely defining geometry and problem type.

Utilizing a combination of meshing rules tailored to fire simulation cases and parameters from a meshing library, the automated meshing routines determine the optimal way to mesh a given fire modeling scenario. The automated meshing tool can be initiated following the complete specification of geometry and physics for simulation.

The meshing tool initially analyzes the current scenario, offering the user a selection of cell budgets. This grants the user control over the number of computational cells dedicated to simulating the problem. The specified number of cells represents a trade-off between accurate simulation (with a high number of fine cells) and quicker results acquisition (with a lower budget of coarser cells). For beginners, the automated meshing system always suggests the [Recommended] strategy for a cell budget. Alternatively, the user can decide to use other budget types, such as “Unit blocks” with only one cell per slice (useful for manual mesh specification) or “Coarse mesh”, which minimizes cells without violating meshing rules. The user can also override the recommended budget by adjusting the directional cell budgets spin boxes for [X dir], [Y dir], and [Z dir] numbers of cells. However, the selected directional cell budgets may not precisely match the generated ones due to the automated meshing system adhering to meshing rules.

The meshing system incorporates "smart" components that conduct checks on the case specification, identifying inconsistencies or errors in the setup. These checking tools also look for potential issues from not employing techniques superior to the default behavior.

Additionally, the meshing system features a mesh-checking facility that examines mesh cells for problematic aspect ratios between edges within a cell and among adjacent cells. These checks assist the user in evaluating mesh quality.

Once the automated meshing tool has analyzed the case and generated an acceptable mesh specification, it must be accepted to create the case specification files for the CFD Engine to process.

6.6.2. The meshing of the main engine fuel oil pipeline fire scenario

Except for parallel meshing, there are some options left the same for all three scenarios:

The geometry type is set to "Multiple storey building geometry", as the engine room is created as a two-storey building (Figure 28).

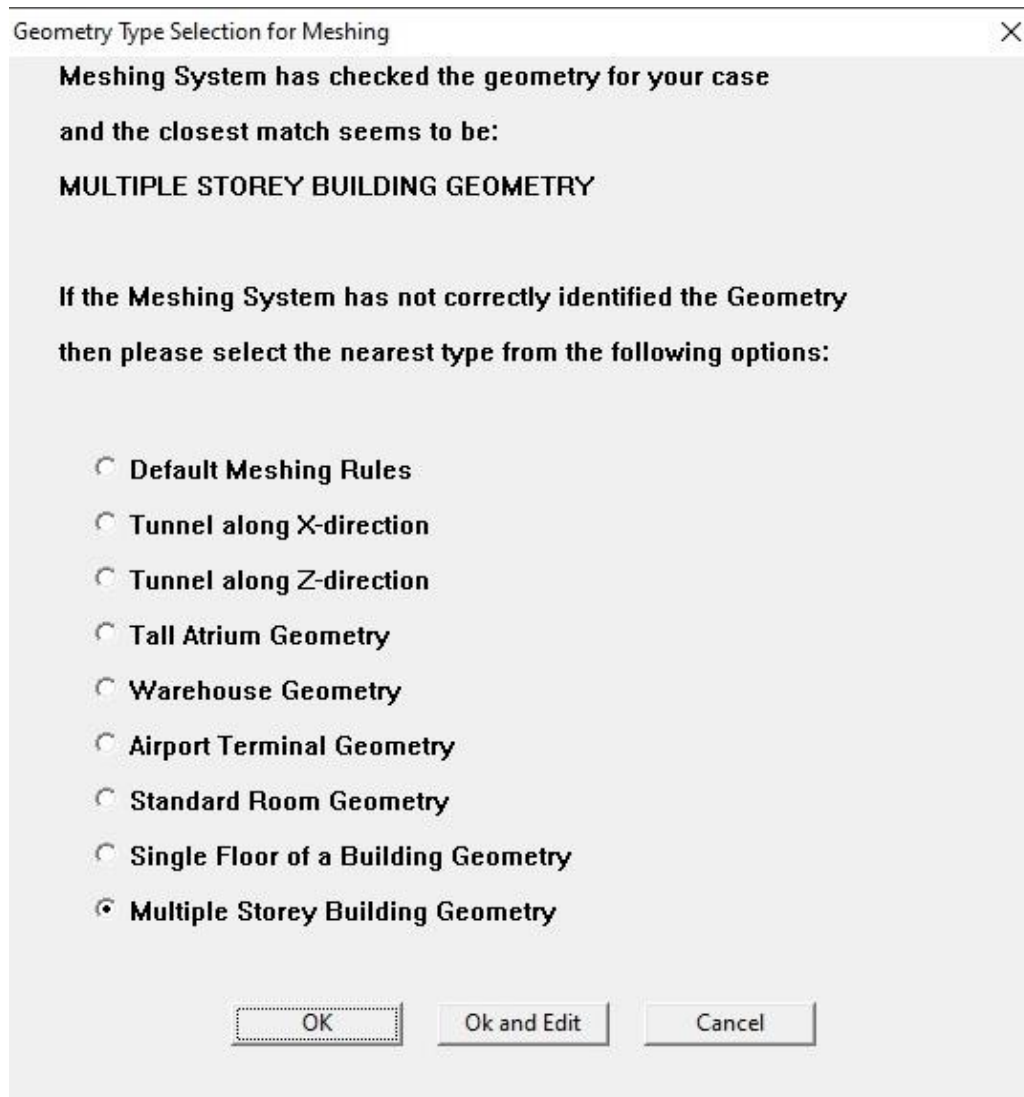


Figure 28 – Geometry type.

The mesh type is set to “recommended”, but since the six-flux radiation model requires it to run, the mesh is coarse, and the refinement option is selected, so the mesh is of better quality.

Cell budget for meshing

Please select **Meshing Strategy**, **Cell Budget** and **Refinement Strategy**.

The meshing system has evaluated your geometry and advises that the minimum recommended cell budget is:

284544 cells in total, with NX = 96 NY = 38 NZ = 78

Strategy to use for initial meshing

☐ Unit blocks
 ☐ Coarse mesh
 ☒ **Recommended**

You can over-ride the directional cell budgets using the spin boxes below. The meshing system will then attempt to distribute these cells.

Directional cell budgets to be used for meshing

X dir:
 Y dir:
 Z dir:

Total default cell budget: 284544 (before refinement)

The mesh specified above can be adjusted using iterative refinement.

Two sets of quality metrics are available for refinement, however the use of refinement may result in a significantly increased cell budget.

Strategy for iterative improvement of the Mesh Quality

☐ No Refinement
 ☒ **Refine**
☐ Refine+

Ok Cancel

Figure 29 – Mesh properties.

The interactive meshing system created a mesh (Figure 29) of 607.600 cells (after refinement) of the “brick” type for solving. During problem modeling, meshes of more than 2.000.000 cells were solved, but it was impractical in several ways. The time to solve a simulation with such a high cell number is significant despite the high-power processor and all 36 threads employed on the problem. Results that were given by high cell number have not generated better results, just slightly better-presented results, because of denser mesh, so it was decided to use coarser mesh shown in Figure 30.

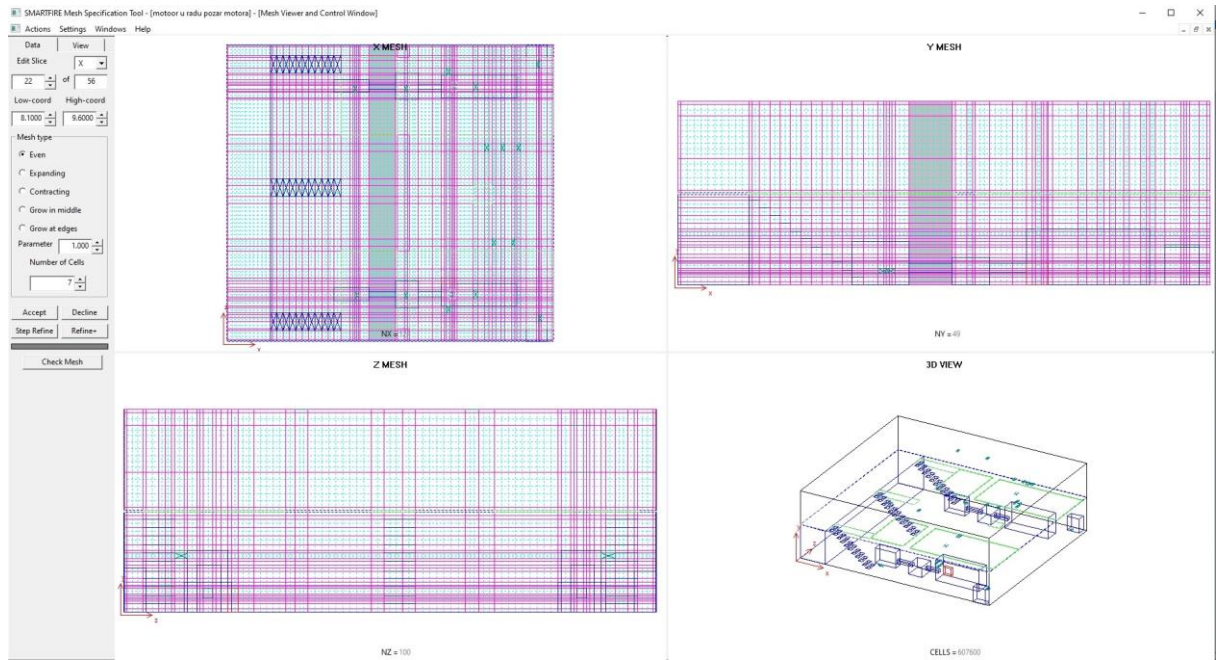


Figure 30 – Mesh for the main engine fuel oil pipeline fire scenario.

6.6.3. Meshing of the purifier fire scenario

Similar to the main engine fuel oil pipeline fire scenario, it has 631.904 cells, and the only difference is the fire's location and size, as seen in Figure 31.

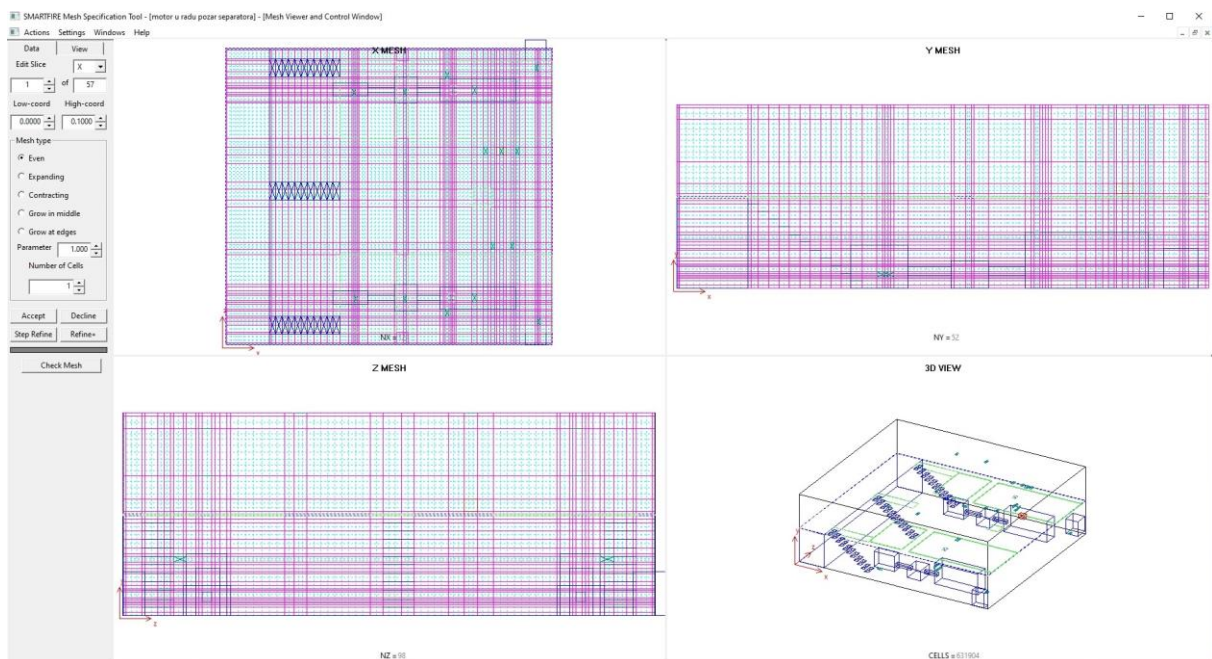


Figure 31 – Mesh for the purifier fire scenario.

6.6.4. The meshing of the oily rags left in a bucket scenario

Once again, the only difference is fire size and location; this mesh consists of 625.000 cells (Figure 32).

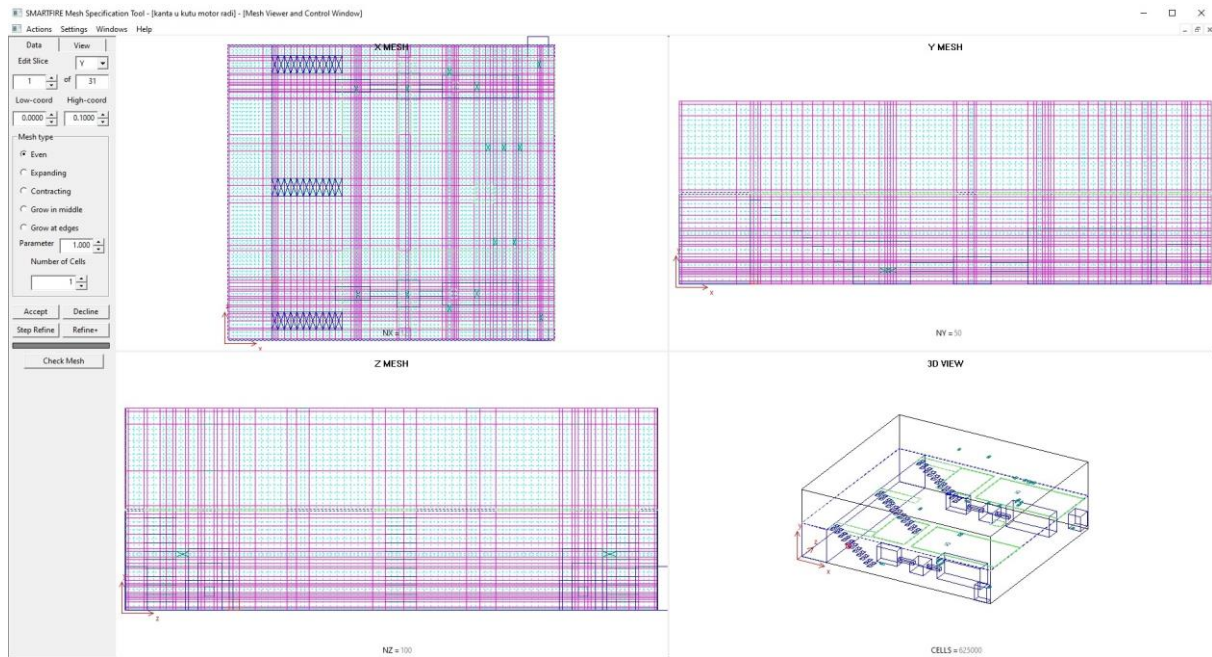


Figure 32 – Mesh for the oily rags left in a bucket scenario.

6.7. Running scenarios

The CFD code utilizes the SIMPLE pressure correction algorithm. It can solve coupled turbulent (two-equation k-epsilon closure with buoyancy modification) or laminar flow problems under transient or steady-state conditions. It also supports sub-models for calculating thermal radiation (via Radiosity, Six Flux, or Multiple Ray radiation models), simple gaseous combustion using the Eddy Dissipation model, and smoke spread.

The current version of Smartfire simulates the fire source by using a user-defined volumetric heat release rate or as a mass source of combustible gas.

Unlike traditional fire field models, the Smartfire user interface enables interaction with the solution by observing its development and adjusting to control parameters while the code is in operation. This dynamic user control is feasible due to the open architecture, integral GUI, and data monitoring and examination tools. Expert users can achieve significant savings in computational time by adjusting parameters dynamically. The solution data is displayed graphically, allowing users to see the effects of their changes easily.

Further exploration of this approach involves developing automated dynamic solution control by creating an expert system to control parameters when certain conditions are detected. This aids novice users and may include controlling parameters such as time steps, iterations per time step, and relaxation values. Early indications for dynamic solution control have shown up to 50% run-time savings.

The object-oriented data structures on which the CFD code is based enable the development of a novel solution technique known as the unstructured "group solver." This solver, still undergoing development and testing, allows the solution domain to be split into a number of geometric or solution-based groups, each controlled independently. Group solvers, coupled with expert system dynamic control, offer the potential for reducing overall computational time while preserving the accuracy of the CFD techniques.

All the scenarios were successfully solved within 27-30 hours each (Figure 33). Other iterations included meshes of approximately 200.000 cells and over 2.000.000 cells, which took around 6 and 50 hours to solve.

Better quality meshes produce "heavy" results for the hardware to run once it is transferred into VR, and coarser meshes produce "grainy" images once imported into VR.

The selected mesh quality is optimal from the hardware capabilities and image quality perspective.

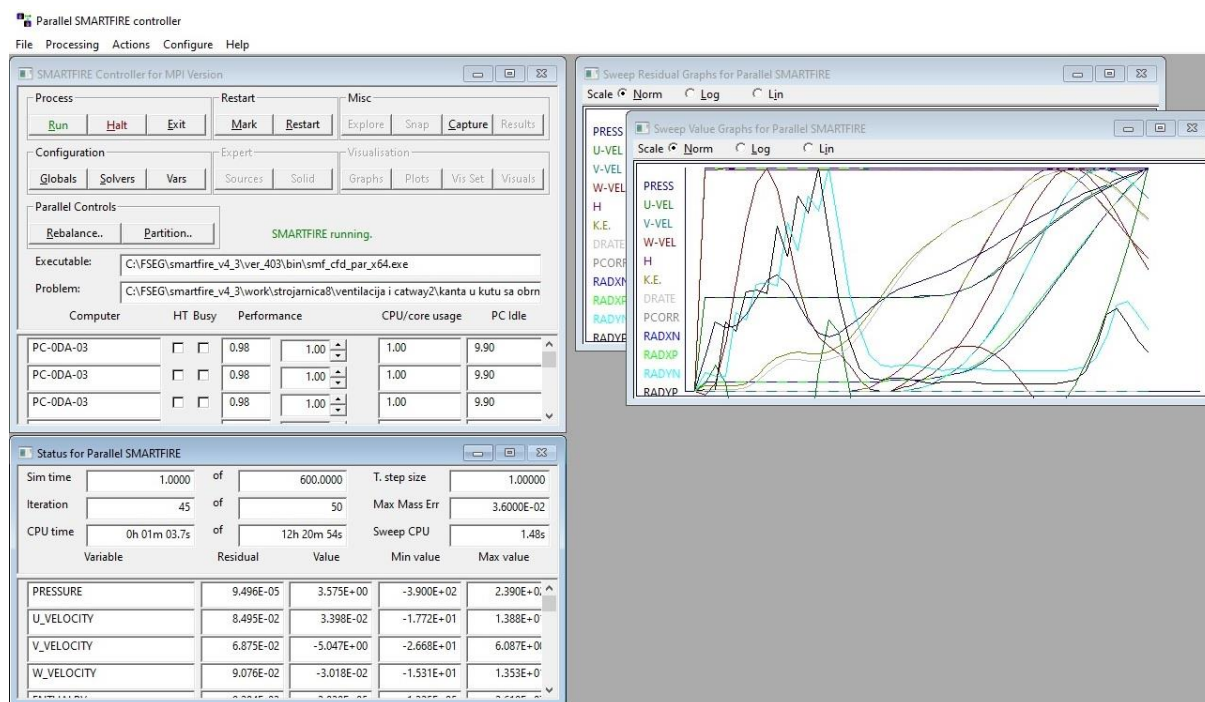


Figure 33 – Solving of scenarios.

6.8. Results of CFD analysis

All the requested data is saved each time step in its own VTK and VTU file. Since there are 3000 time steps for each simulation, data quantity is significant (around 60Gb of data per scenario).

Inspection of that amount of data is no easy task. The Smartfire DataView (Figure 34) serves precisely that purpose; as a post-processing visualization and animation tool. It offers convenient access to commonly needed visualization features, including isosurfaces, contour fill cut planes, velocity vectors, and volumetric smoke visualization. Additionally, the DataView tool includes a user-friendly animation function, particularly when results data (in VTK or VTU format) is available for each simulation time step, which is the case.

Additionally, for this research, the processing of particular data is not of interest; the whole package of results needs to be presented in the VR engine room.

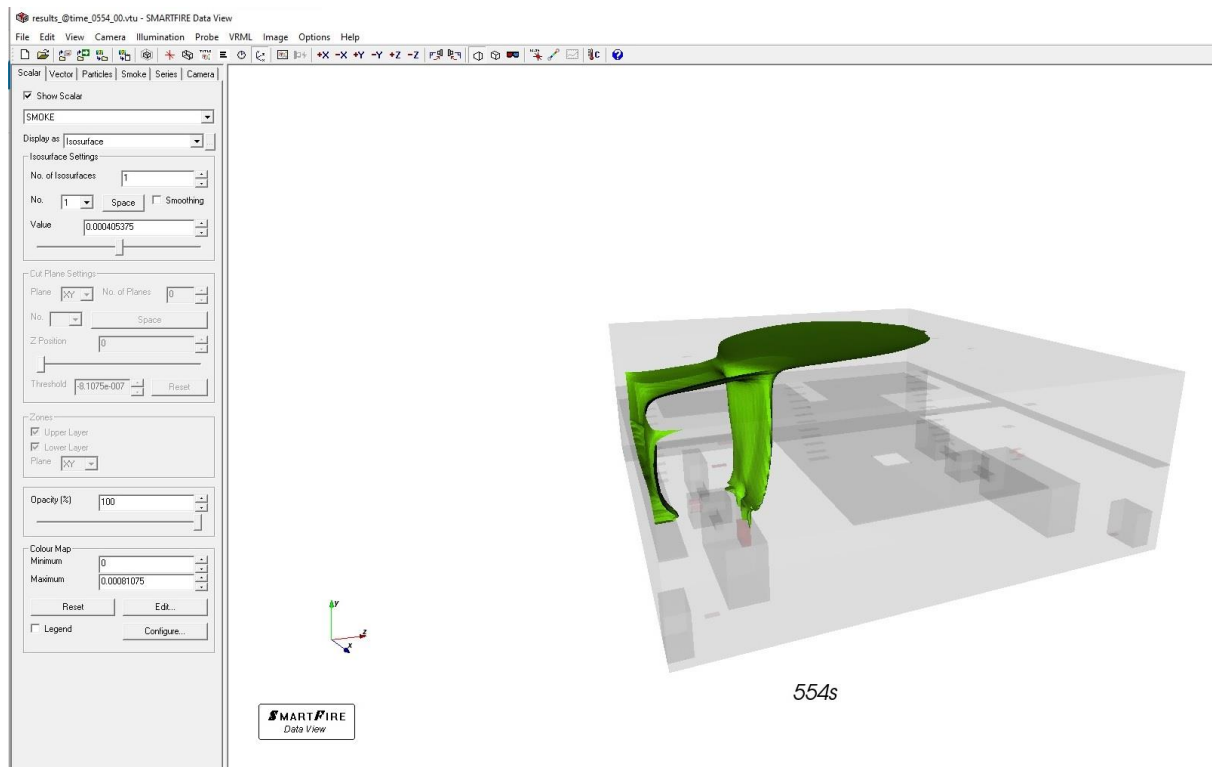


Figure 34 – DataView software.

6.8.1. Results of the main engine fuel oil pipeline fire scenario

The best way to present results for all three scenarios is the animation of different isosurfaces throughout time. Still, due to limitations of printed format, only screenshots of the last time step for smoke (Figure 35) and temperature (Figure 36) isosurfaces are presented.

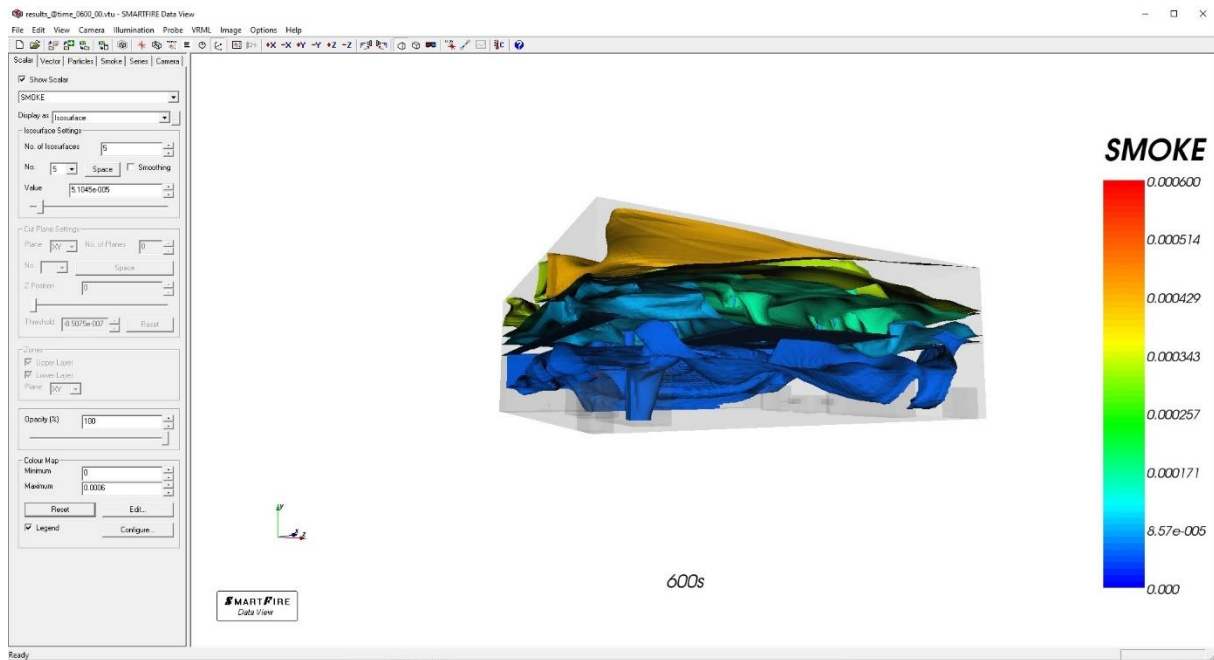


Figure 35 – Main engine fuel oil pipeline fire scenario – smoke density isosurfaces at $t = 600s$.

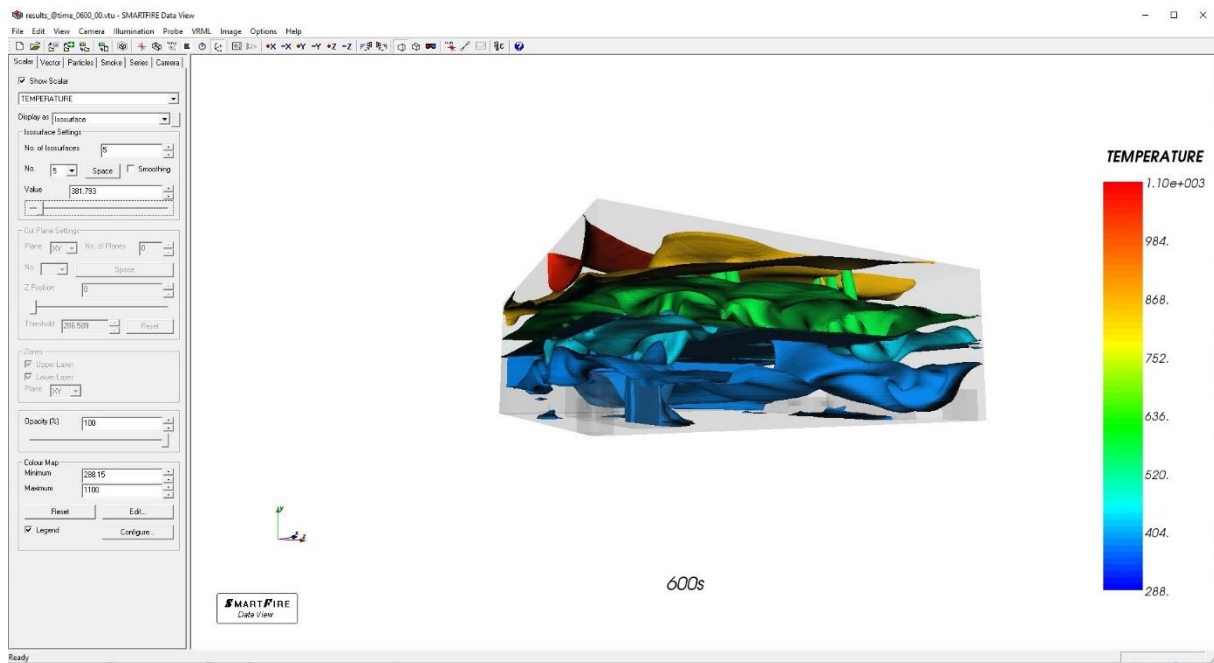


Figure 36 – Main engine fuel oil pipeline fire scenario – temperature isosurfaces at $t = 600s$.

6.8.2. Results of the fuel oil purifier fire scenario

As for the fire of the main engine fuel oil pipeline scenario, only screenshots of isosurfaces of smoke density and temperatures are provided in Figure 37 and Figure 38.

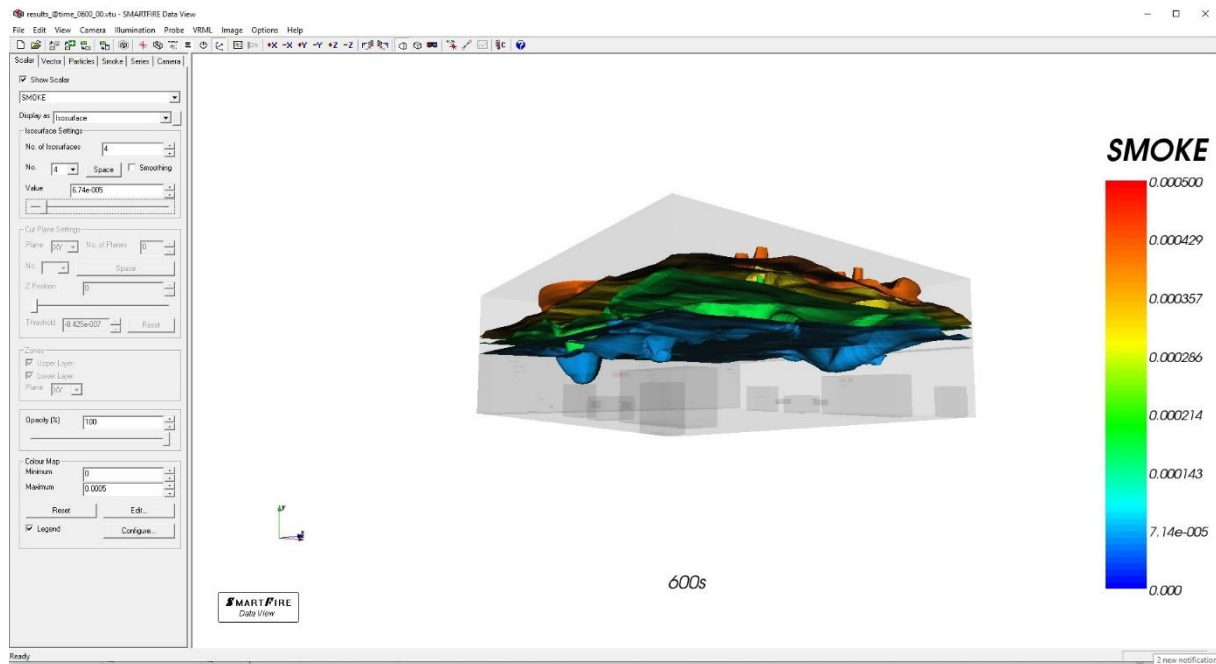


Figure 37 – Fuel oil purifier fire scenario – smoke density isosurfaces at $t = 600s$.

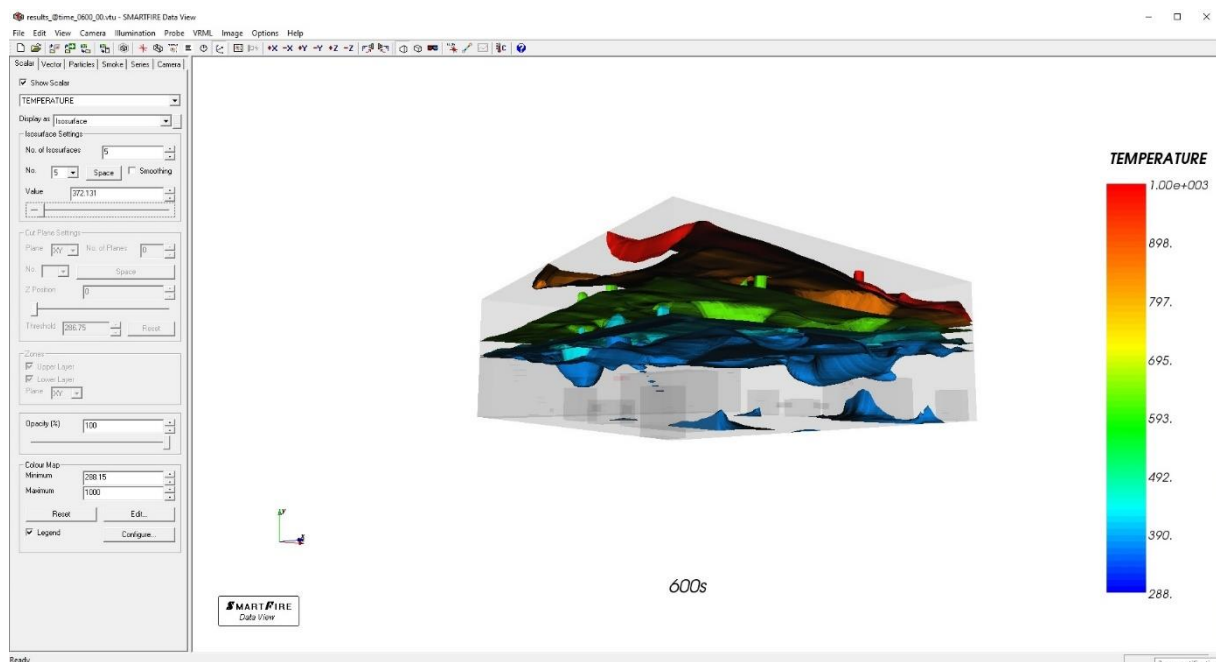


Figure 38 – Fuel oil purifier fire scenario – temperature isosurfaces at $t = 600s$.

6.8.3. Oily rags left in a bucket fire scenario results

Once again, isosurfaces of smoke density and temperatures are given in Figure 39 and Figure 40.

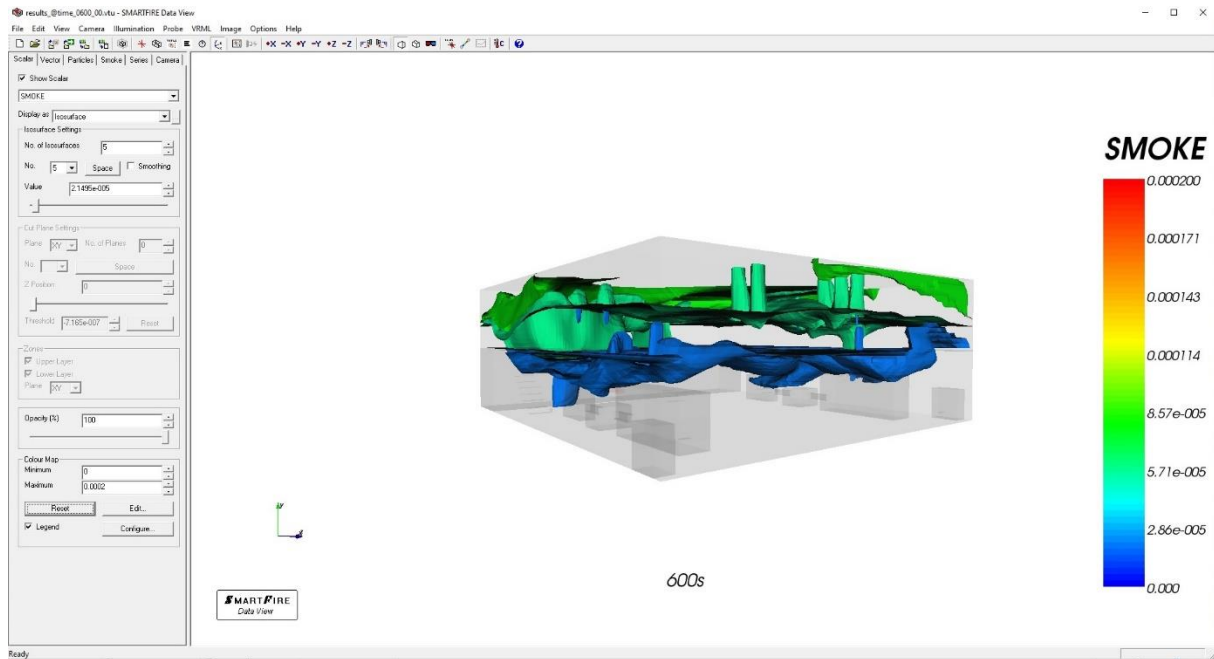


Figure 39 – Oily rags left in a bucket fire scenario – smoke density isosurfaces at $t = 600s$.

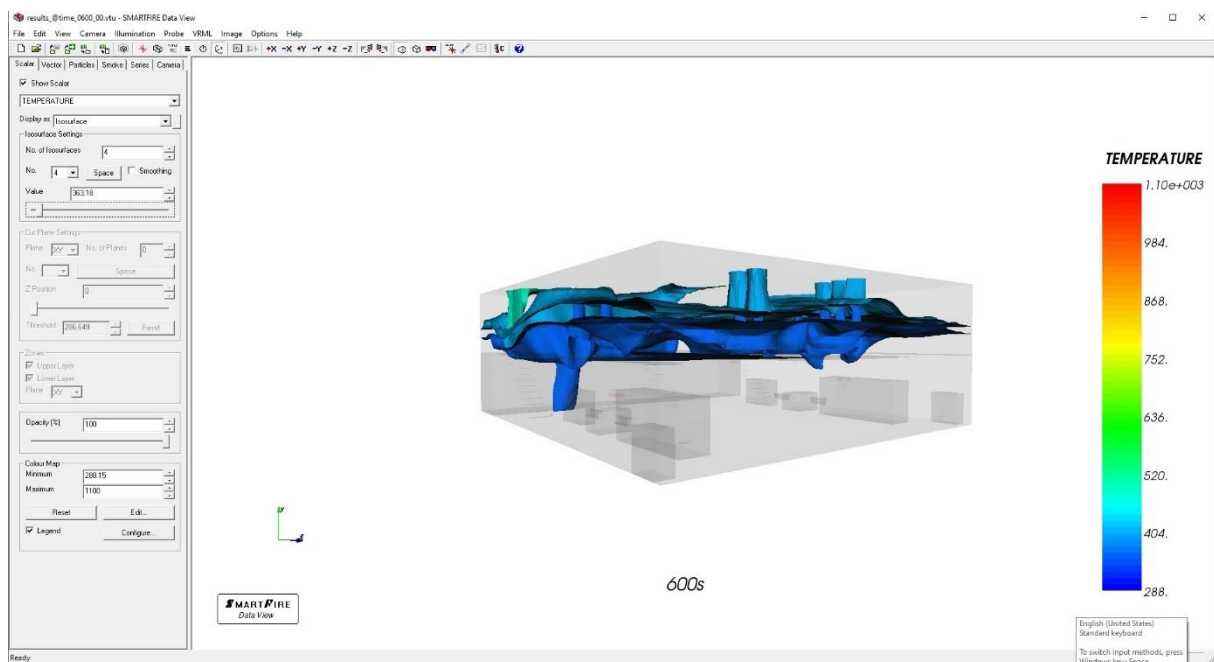


Figure 40 – Oily rags left in a bucket fire scenario – temperature isosurfaces at $t = 600s$.

6.9. Research validation

Experimental evaluation of simulated results would require an actual engine room to be set on fire. It would be unreasonable to make that kind of evaluation even if that particular engine room existed. Since the engine room that is the subject of this research does not exist in physical form but in virtual reality, it is impossible to validate the results experimentally.

Since it is impossible to validate results, it was decided to validate the method itself. Steckler, Quintiere, and Rinkinen experimented in 1982 and described it in a paper labeled “Flow induced by fire in a compartment” [82]. Their experiment will be recreated using Smartfire software, and the results of given temperatures will be compared.

Additional validation is obtained by alternating scenario inputs while observing the change of temperature and smoke density.

6.10. Steckler’s experiment

Fifty-five comprehensive steady-state experiments were conducted to investigate the flow generated by a simulated pool fire within a compartment, reflecting conditions typical of the fire's initial development phase. The mass flow rate through door or window openings and limitations on the fire plume entrainment rate are presented in relation to opening geometry, fire intensity, and fire location.

The observed characteristics of opening flow rates are elucidated through a straightforward hydrostatic model based on temperature distribution. The study demonstrates a strong correlation between the measured outcomes and idealized flows, accounting for the entire temperature distribution.

Entrainment outcomes for fires near walls align reasonably well with results obtained from free-standing plume models. Except for the smallest openings, fires in other locations exhibit an entrainment rate two to three times higher than predicted by

these models. This phenomenon is attributed to disturbances within the room induced by the opening flow, resembling the behavior of a fire plume in a crosswind [82].

The whole experiment is well documented with all required data for simulation creation (Figure 41).

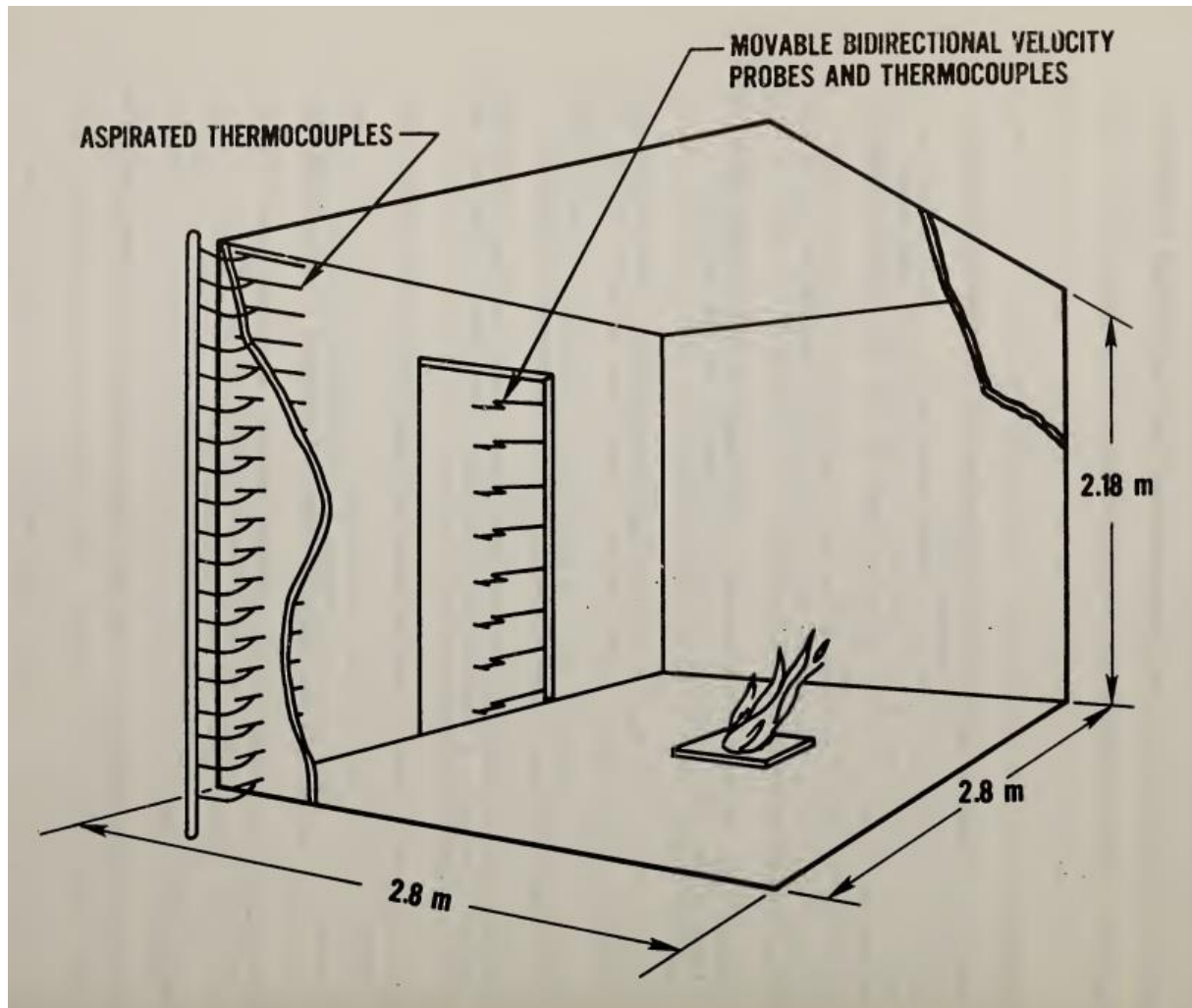


Figure 41 – Steckler's experiment arrangement [82].

Of all the variations of the experiment that Steckler conducted, the one with fire placed centrally and a door with a width of 0.74m and a height of 1.83m was selected for simulation. Fire releases energy of 62.9kW [82].

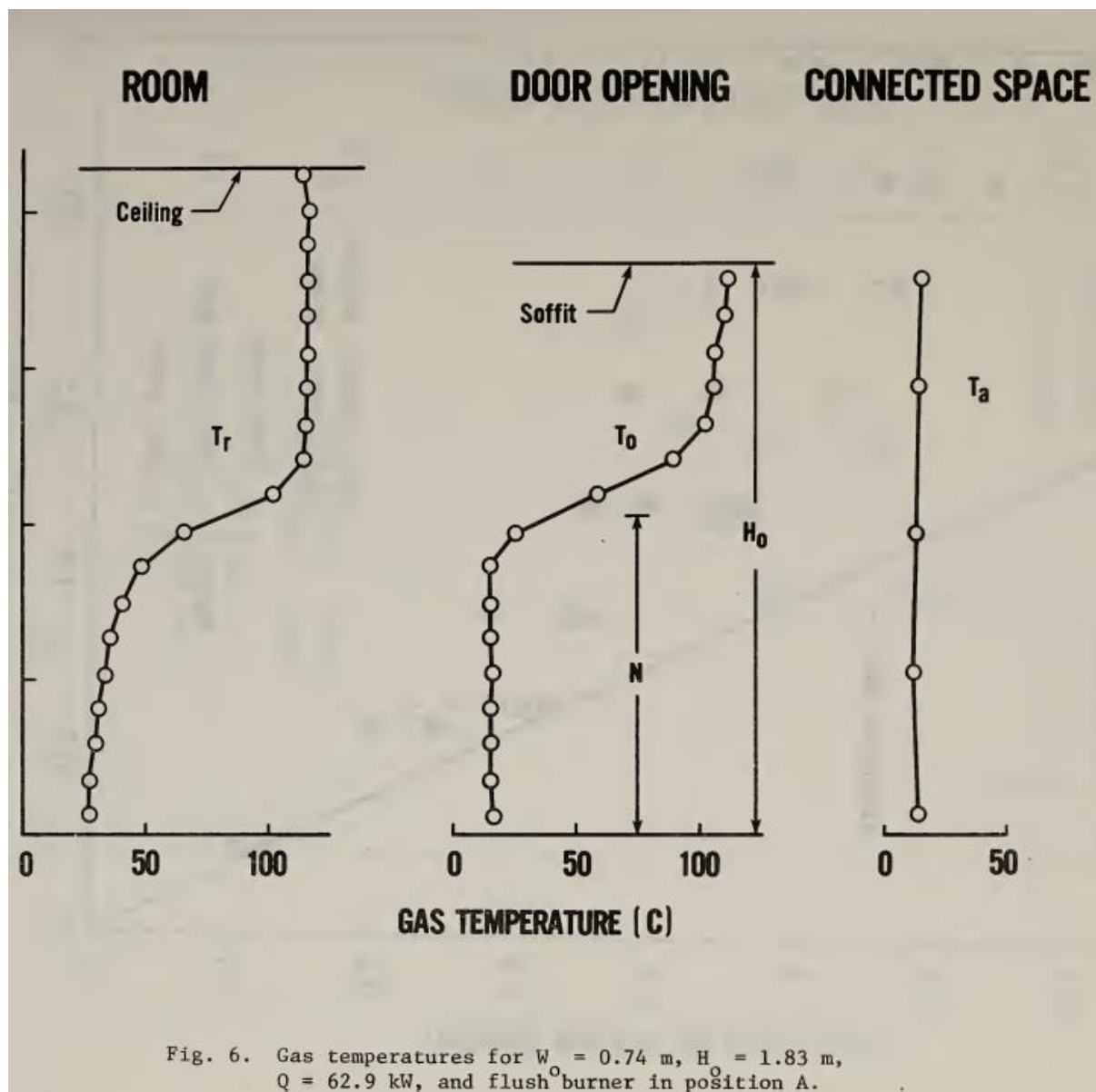


Figure 42 – Steckler's experiment results graph [82].

The selected variation results are derived from the graph, Figure 42, and given in Table 2 and Table 3.

Table 2 – Steckler's results for the corner sensors.

Temperature [°C]	Height [m]
25,86	0,05
25,90	0,16
28,42	0,28

29,68	0,40
32,31	0,51
34,48	0,63
39,39	0,74
46,97	0,86
64,54	0,97
99,85	1,10
112,32	1,21
113,42	1,32
113,79	1,45
114,25	1,55
114,14	1,68
114,31	1,79
114,22	1,92
114,96	2,02
112,44	2,14

Table 3 - Steckler's results for the door sensors.

Temperature [°C]	Height [m]
15,60	0,05
14,48	0,16
14,63	0,29
14,45	0,40
15,36	0,52
14,58	0,63
14,40	0,74
14,28	0,87
24,84	0,97
57,40	1,10
88,45	1,22
101,59	1,33
105,19	1,45

105,80	1,56
109,86	1,68
110,96	1,80

6.11. Simulation of the Steckler's experiment

Since the experiment is well documented and its compartment is quite simple, there was no need to simplify geometry or fire properties.

A simple room with dimensions 2.8mx2.8m is created. The only opening is for the door, which has a dimension of 0.74mx1.83m and is located in the center of one of the walls, as shown in Figure 43.

Five sensor stacks are in each room corner, and the fifth is in the middle of the door opening. Sensor stacks are chosen for their ability to record designated parameters in the range stack height, similar to sensor placement in the original experiment.

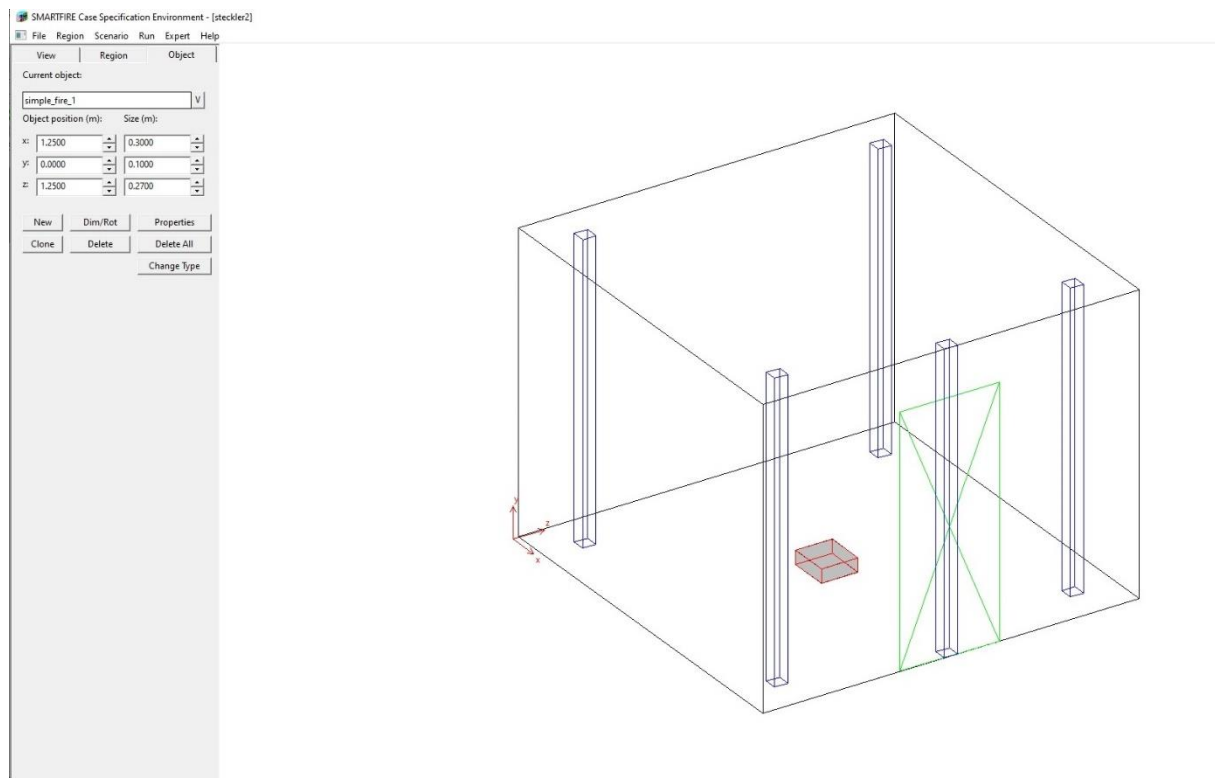


Figure 43 – Geometry and fire and sensor locations for Steckler's simulation.

Flow, combustion, heat transfer, radiation, and smoke models are selected for the simulation. The radiation model is to be multiple-ray, with 24 rays. The ambient temperature is set to 13°C, and the timestep is 2 s. A total of 500 timesteps are to be solved to determine when the simulation will reach equilibrium, and the results will be used at that point.

Fire is set to constant fuel generation with an output of 0.001258 kg/s, equivalent to a 62.9 kW peak heat output, Figure 44.

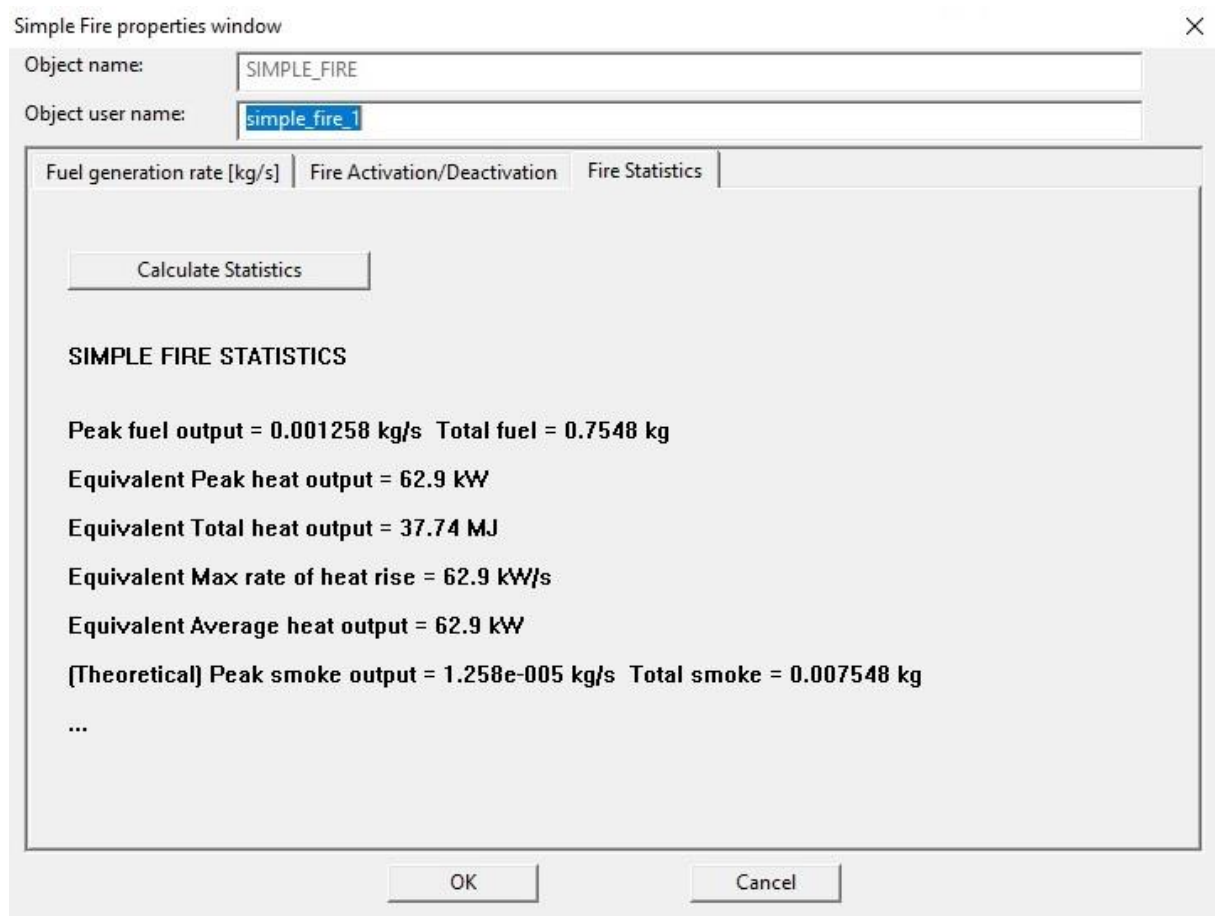


Figure 44 – Fire statistics for Steckler's simulation.

Meshing was made automatic for parallel computing with the “refine+” option enabled, resulting in a quite large mesh of 202.176 cells, Figure 45.

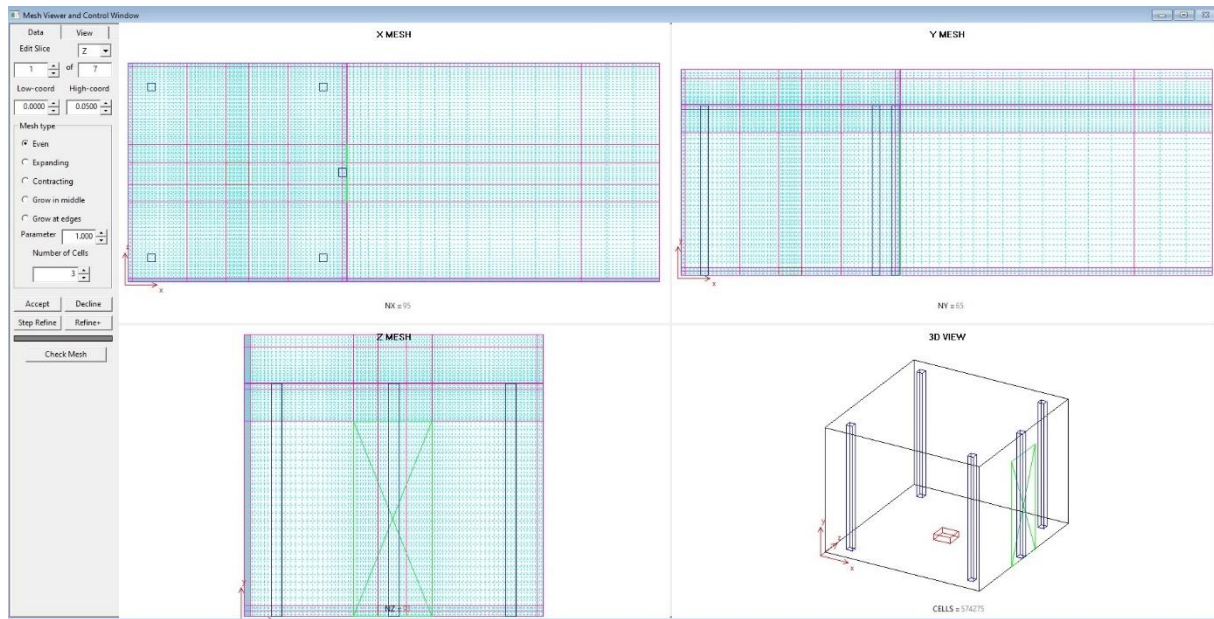


Figure 45 – Mesh for the Steckler's simulation.

6.12. Comparison of results for Steckler's experiment and simulation

In the original experiment, sensors were located in the corner of a wall with the door opening and the middle of the door opening itself. To be as close as possible to experiment conditions, results from sensor stack number 3 (one to the left from the door) and sensor number 2 (the one in the door opening) will be compared with the original results. Comparison is shown in Figure 46, Figure 47, Figure 48, and Figure 49.

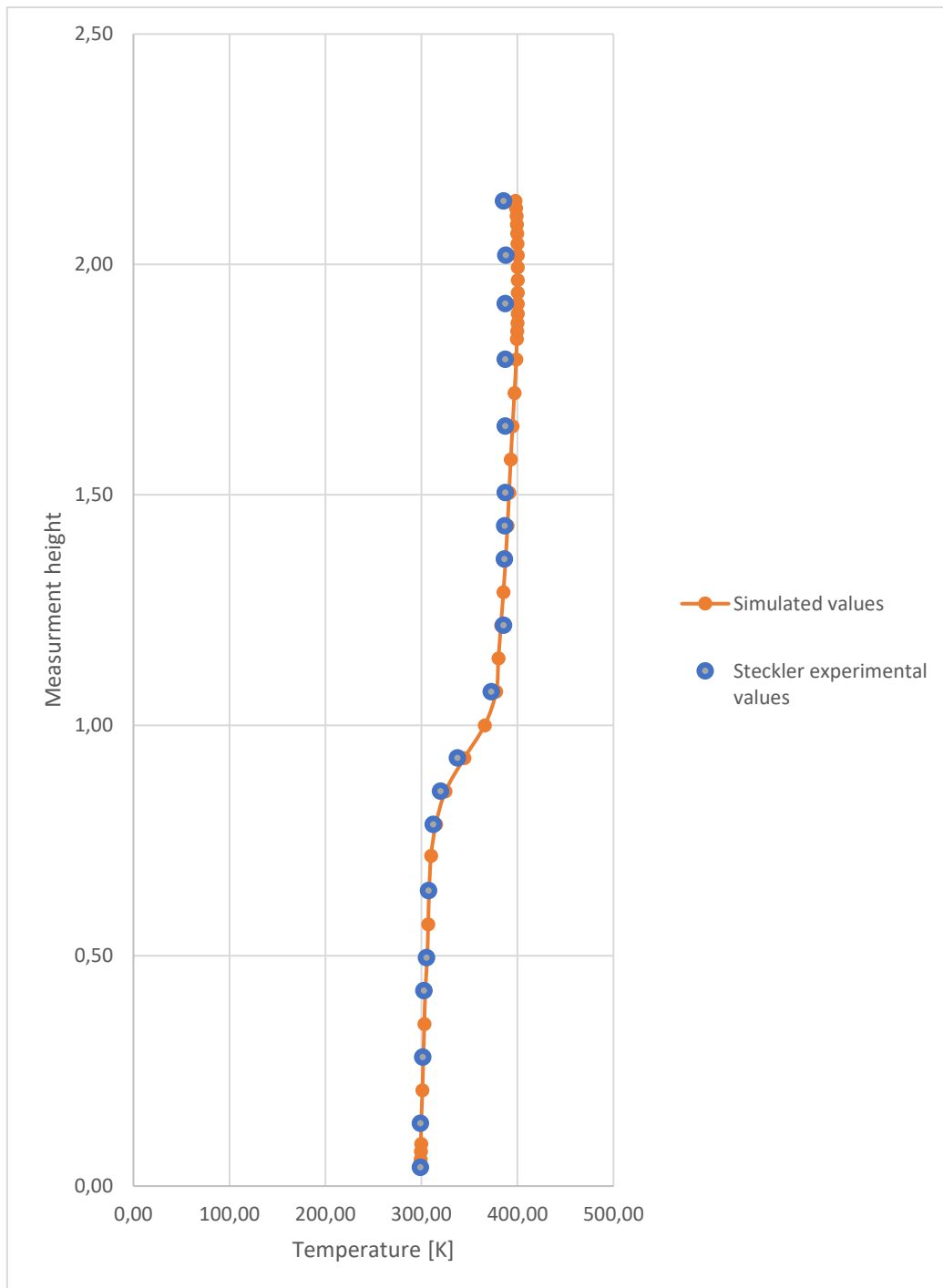


Figure 46 - Comparison of the corner sensors for the Steckler's experiment and simulation.

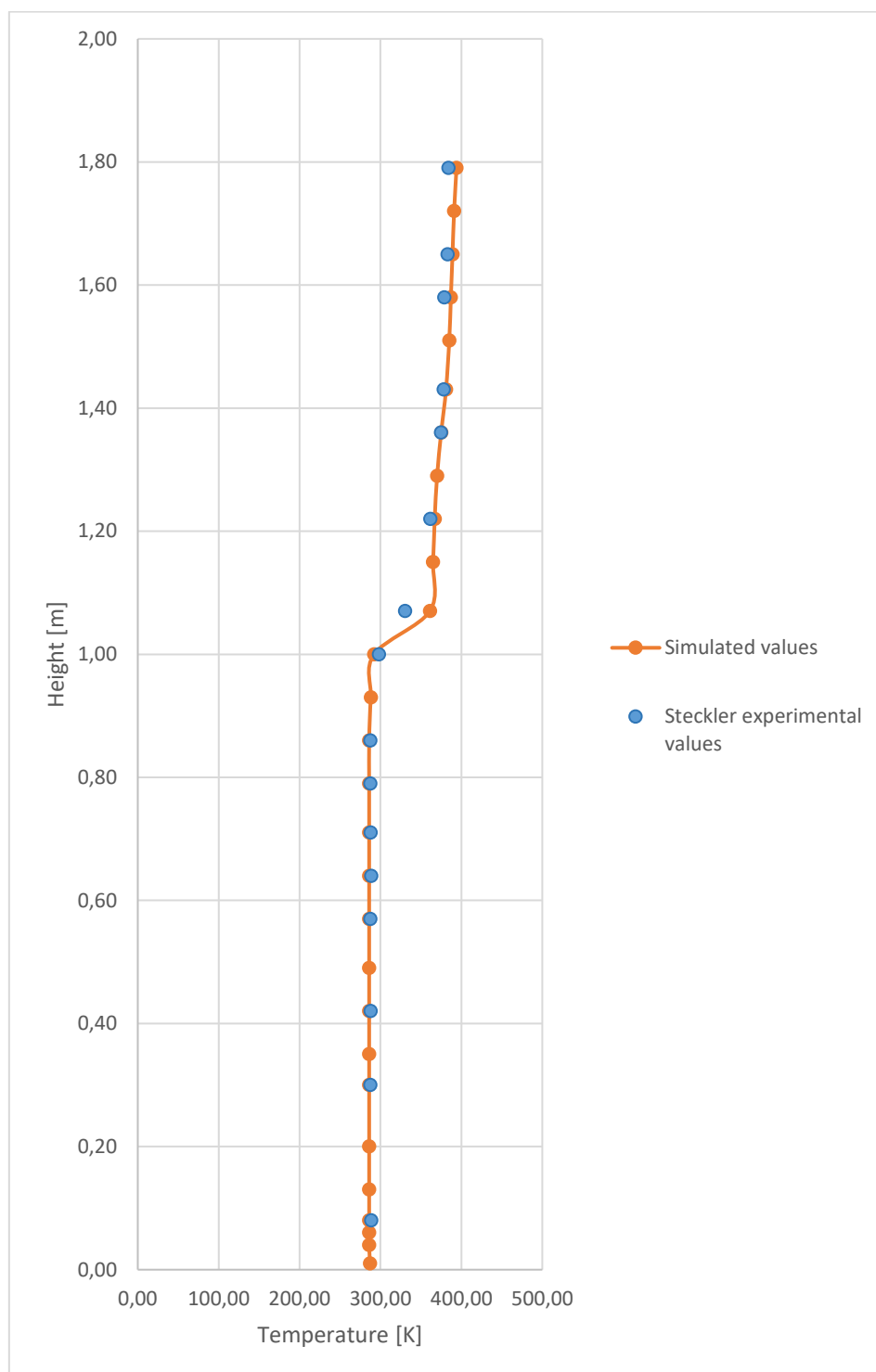


Figure 47 – Comparison of the door sensors for the Steckler's experiment and simulation.

The deviation of results measured during the experiment and results gained from the simulation is calculated and shown as percentages on the graphs.

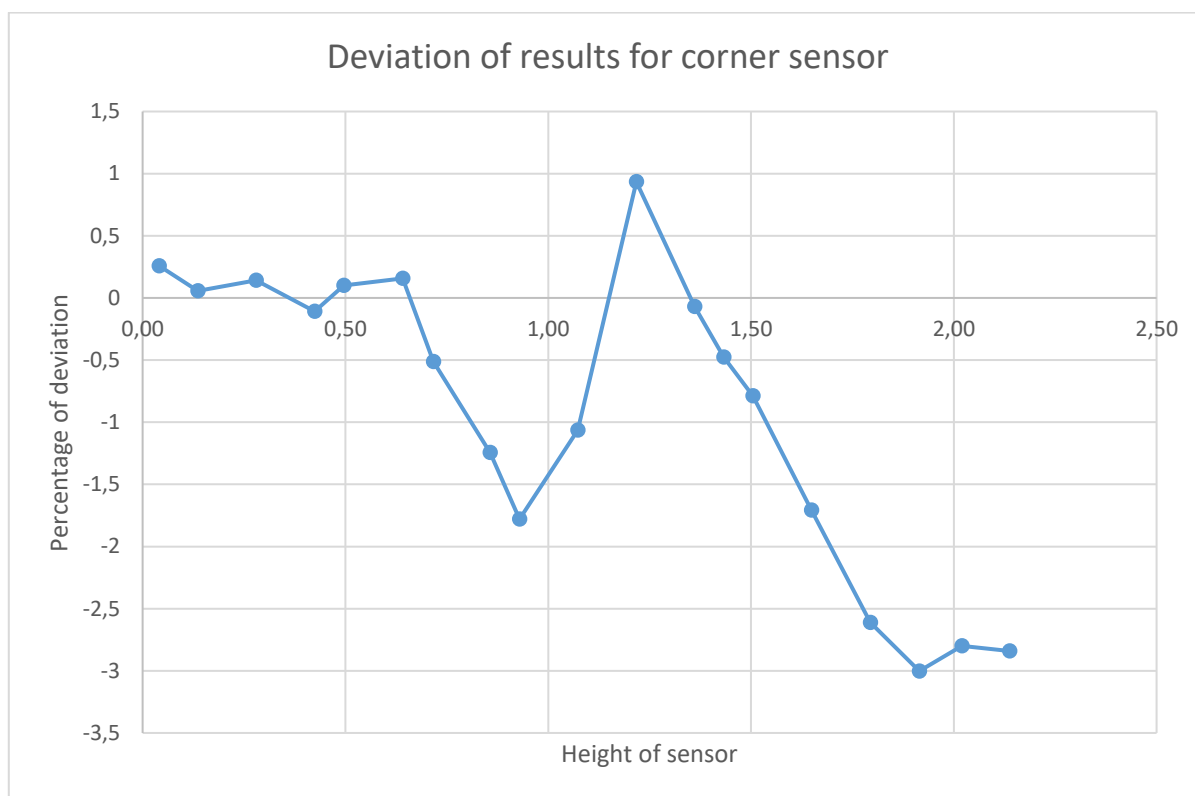


Figure 48 – Deviation of results for corner sensor.

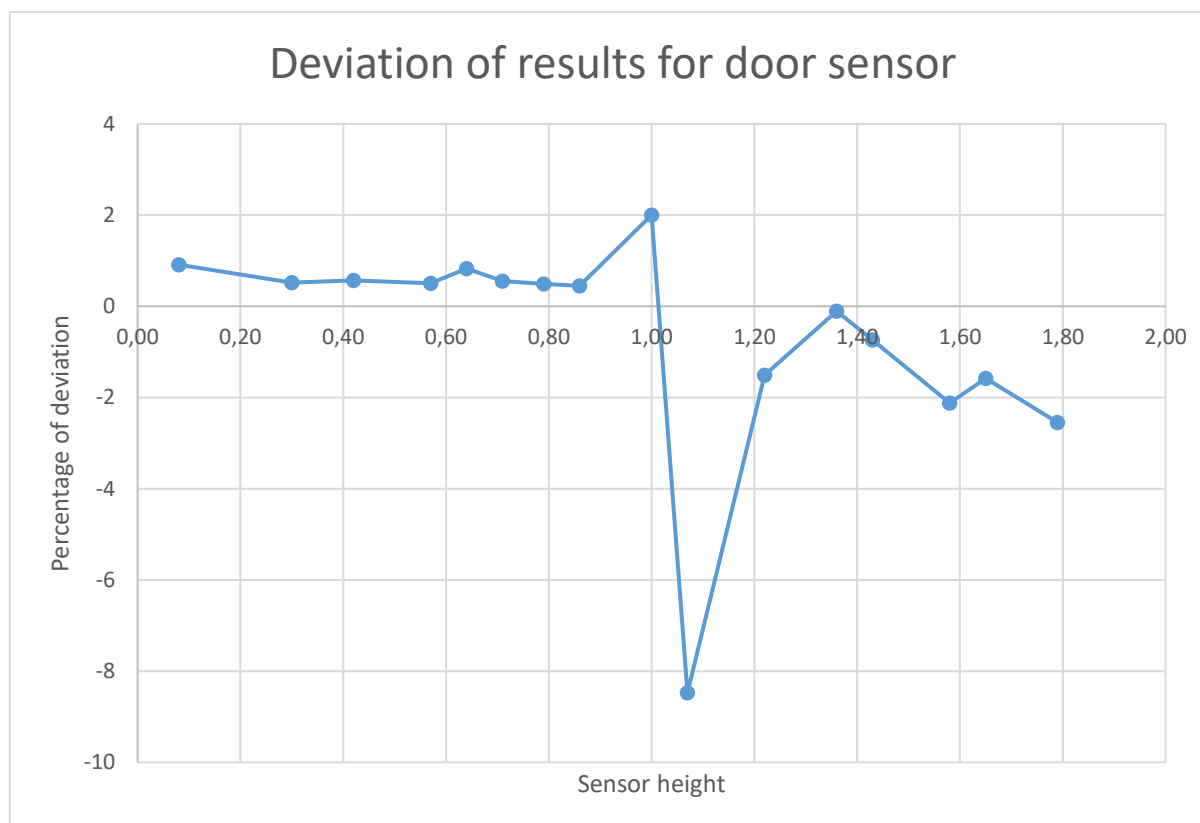


Figure 49 – Deviation of results for the door sensor.

As shown, the results of the corner sensor are highly similar between experiment and simulation. Deviance is no larger than 3%.

Door sensor deviance is not so simple. Although on the results graph, similarity is shown as very high, the deviance graph shows discrepancies of more than 8%. The discrepancy shows on the point in the air layer where there is a transient layer between two temperatures – low and high. The trend is good on the results graph, but that particular point deviates more than 8% of the experiment results. Since experimental results are gathered from the original graph, which was not in digital form, so some deviances could occur when converting data because of the thick graph line, and since those points are in the transient layer, which is thin, even minor error in sensor position height generates significant deviation of temperature. According to Ed Galea, another possible reason for such deviance is that door sensors were not completely still during the original experiment, thus affecting measurement accuracy. All other result points are within a deviation of 3%, which is acceptable.

6.13. Validation by scenario input change

Some confidence in the effectiveness of the developed numerical model can be gained by varying the input parameters and monitoring the resulting variations in temperature and smoke density. In this case, the amount of fuel was varied to examine its impact on the temperature and smoke density distribution throughout the engine room's height (Figure 50 and Figure 51).

It is observed that increasing the fuel quantity in the engine to twice its original amount results in a rise in temperature. Also, reducing the fuel quantity to half of what is utilized in the standard simulation also leads to a decrease in temperatures. The temperature distribution profile across the engine room's height reveals higher temperatures near the ceiling on the upper deck, Figure 50. A similar pattern is observed with smoke density throughout the height of the room, Figure 51.

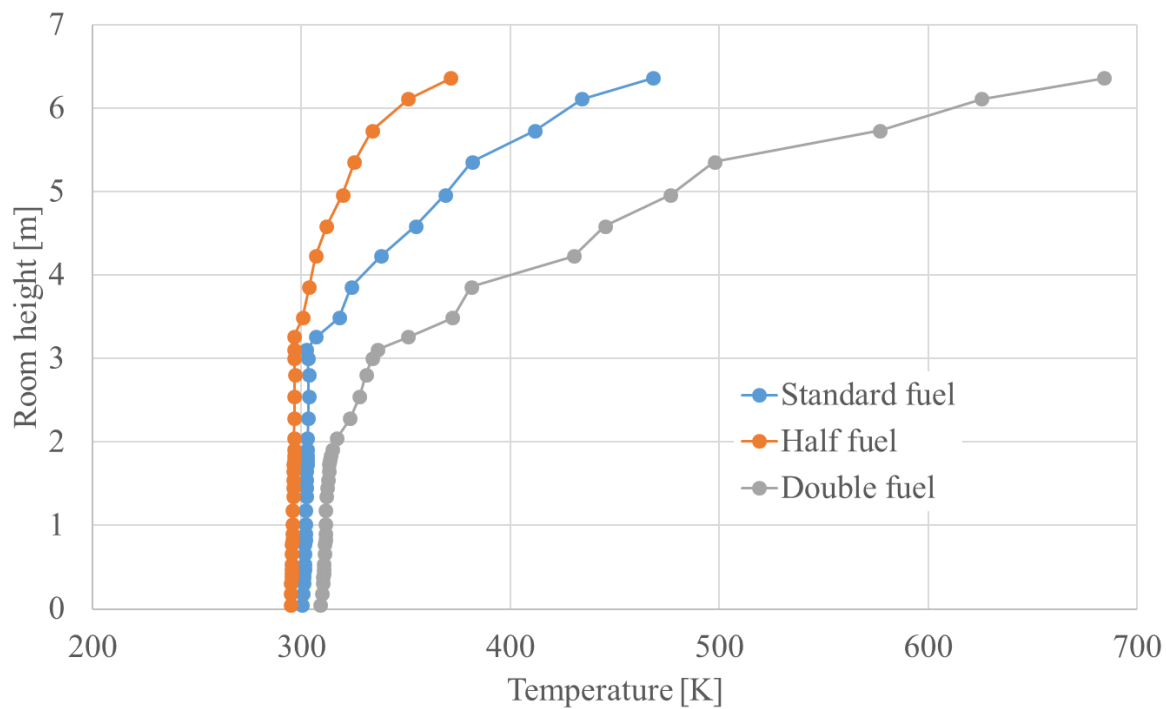


Figure 50 - Comparison of the effect of fuel quantity on the distribution of temperature in timestep $T = 300$.

300.

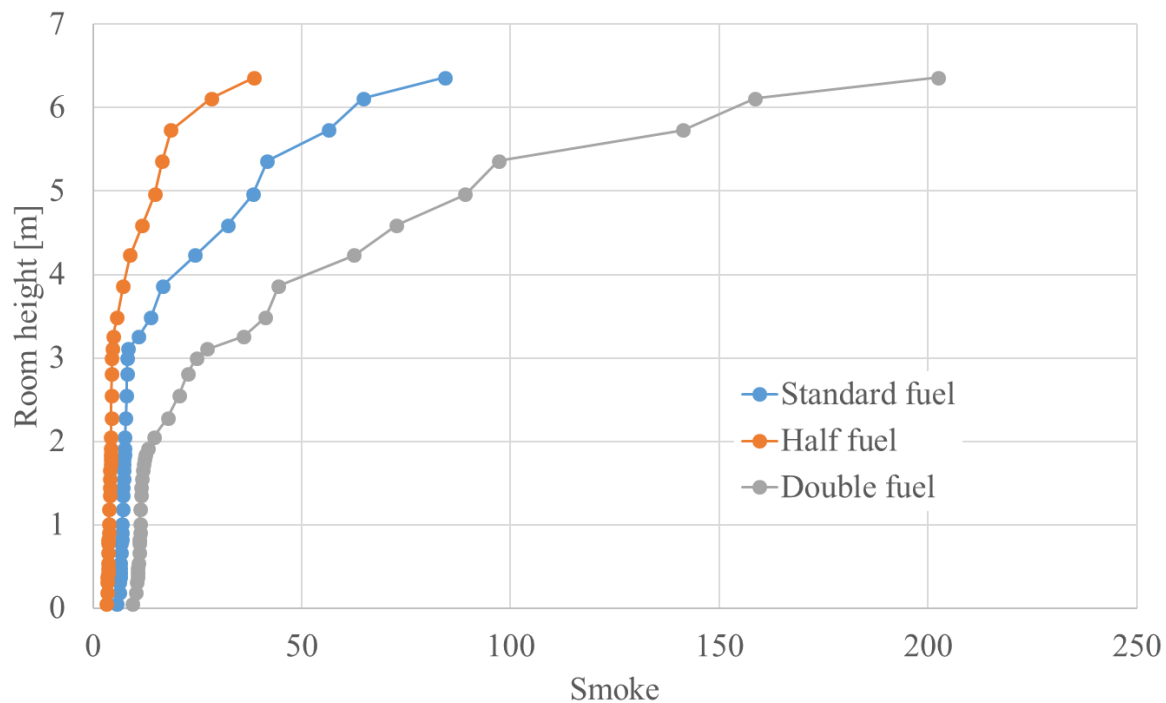


Figure 51 - Comparison of the effect of fuel quantity on the distribution of smoke in timestep $T = 300$.

In addition to this, for standard fuel quantity, distribution of temperature and smoke density was recorded for four different time steps in the simulation (180 s, 300 s, 420 s, 600 s), Figure 52 and Figure 53.

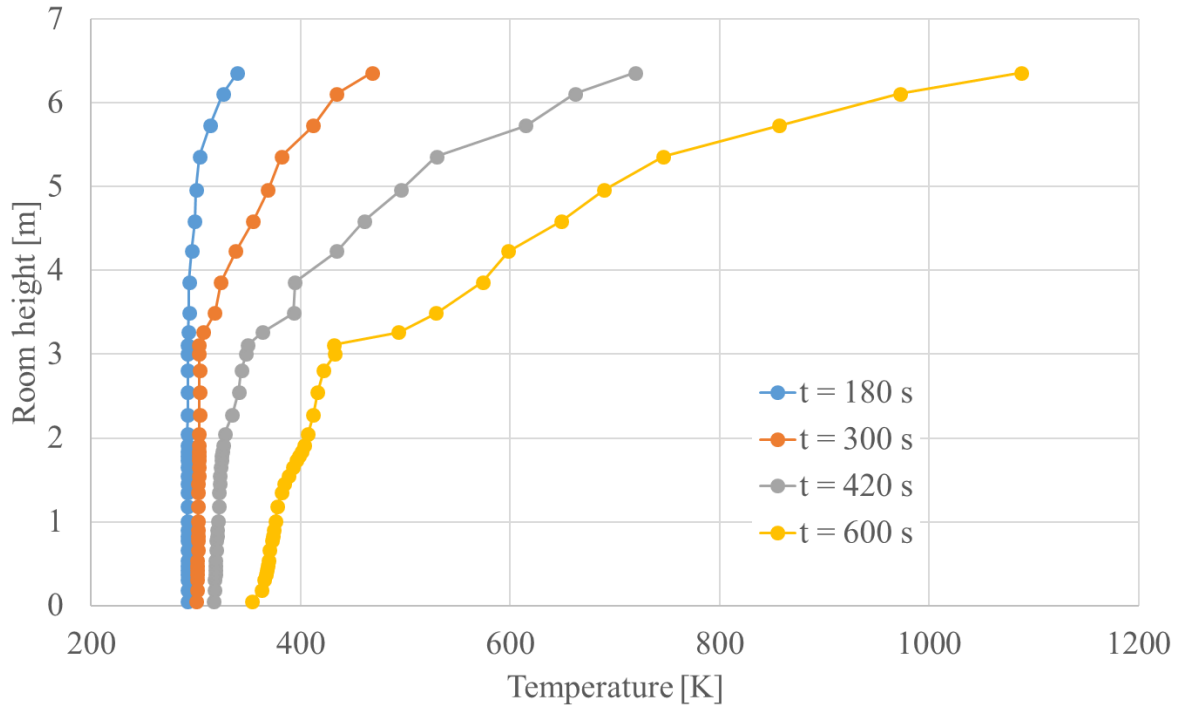


Figure 52 - Comparison of distribution of temperature in four different time steps and for a standard quantity of fuel.

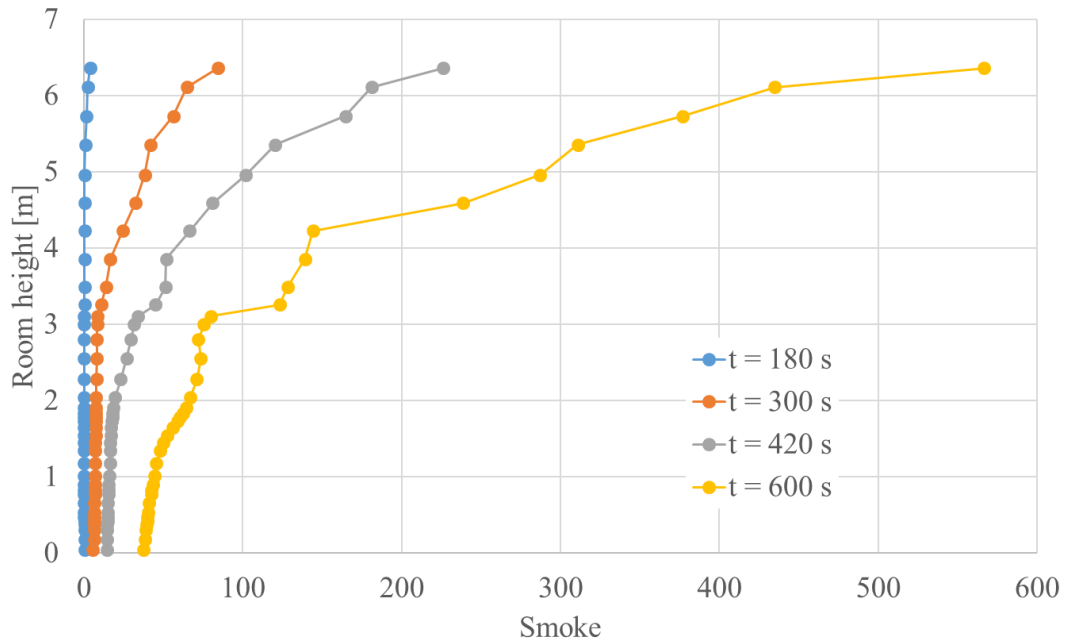


Figure 53 - Comparison of distribution of smoke in four different time steps and for a standard quantity of fuel.

As the simulation advances through the four recorded time steps, a progressive increase in the temperature profile can be observed, with temperatures rising over time, Figure 52. Similar behaviour can be noticed also for smoke distribution, Figure 53.

7. COUPLING CFD AND VR

Smartfire simulation results are saved as .vtu or .vtk files, and the original virtual engine room is made by Unreal Engine, as stated before. Smartfire saves files, and the Unreal Engine uses voxels, so representing one using the other seems like a minor technical task.

7.1. Voxels

Voxels, derived from "volume" and "pixel," represent a crucial concept in computer graphics, medical imaging, and scientific simulations. Unlike pixels confined to 2D spaces, voxels extend into three dimensions, serving as discrete elements in a three-dimensional grid that encapsulates information about specific points in space. This exploration of voxels delves into their applications, attributes, and the pivotal roles they play across diverse fields.

Essentially, a voxel is a distinct element within a three-dimensional grid, capturing information about a particular point's properties, such as color or density. Voxels are fundamental for constructing 3D representations, offering a foundation for developing intricate and lifelike 3D graphics in computer graphics applications.

The significance of voxels extends to medical imaging, where they are crucial for representing three-dimensional information about the human body. Imaging techniques like CT scans and MRI generate volumetric data by partitioning scanned regions into a grid of voxels, with each voxel containing valuable details about tissue density or intensity, contributing to the diagnosis and understanding of various health conditions [83].

Scientific simulations, particularly those in fluid dynamics, weather modeling, and material science, leverage voxels to discretize continuous spaces under study. Voxels effectively represent and analyze complex three-dimensional data in simulations dealing with intricate interactions and behaviors.

An integral aspect of voxels is their resolution, influencing the level of detail in three-dimensional representations. Higher resolution translates to smaller voxels, offering greater precision but demanding increased computational resources. Conversely, lower resolution leads to larger voxels, sacrificing detail for computational efficiency. Striking the right balance between resolution and performance is crucial in various applications.

Voxel-based rendering, distinct from traditional polygon-based rendering, involves visualizing 3D scenes using information stored in voxels. Widely adopted in applications like video games and virtual environments, voxel-based rendering enables unique visual styles and dynamic, destructible environments.

In scientific research, voxels play a crucial role in exploring complex datasets. Whether unraveling geological structures, studying microscopic materials, or simulating fluid flows, researchers harness the power of voxels to represent and comprehend intricate 3D phenomena.

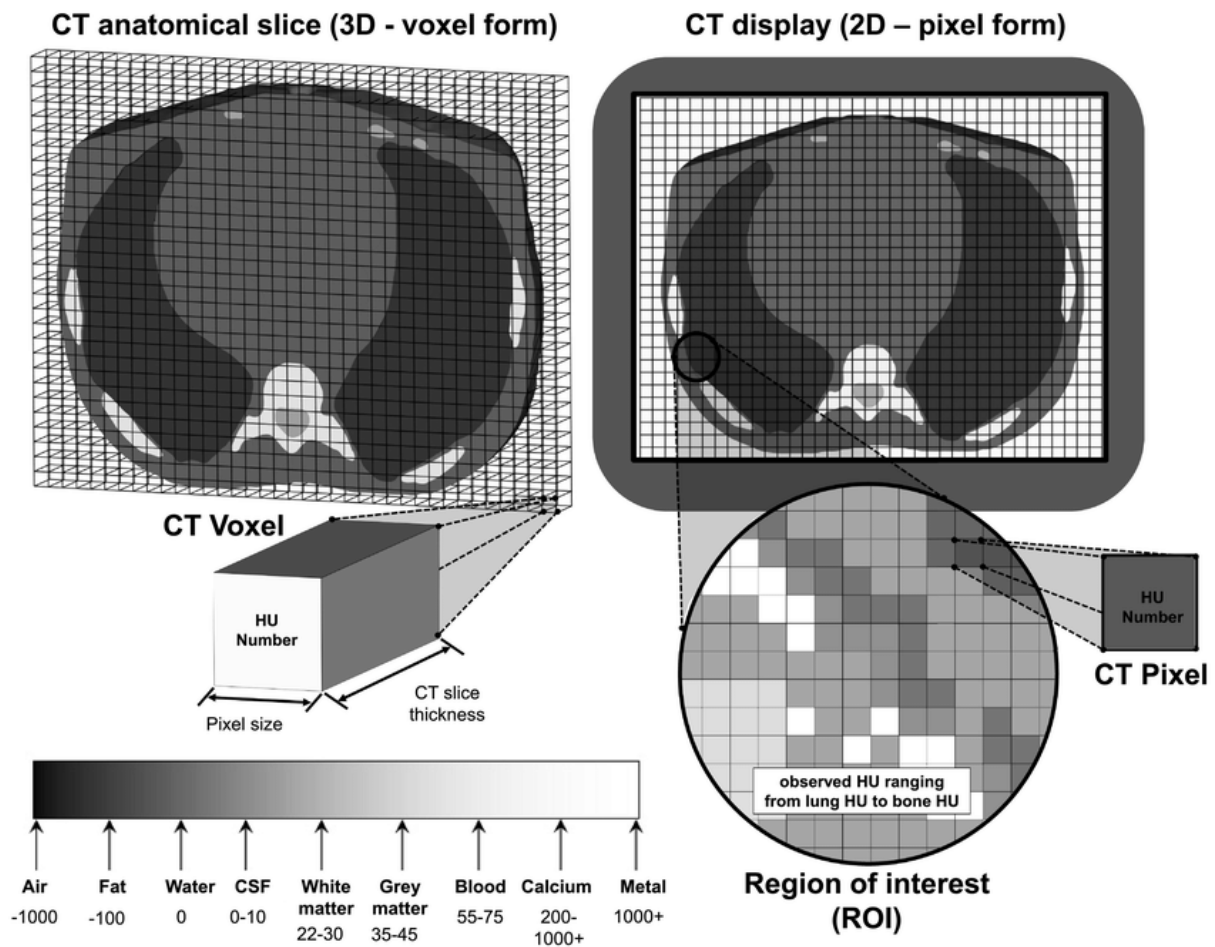


Figure 54 – Graphical voxel representation, compared to pixel [84].

Technological advancements, including faster computation and improved graphics hardware, have expanded the capabilities of working with voxels. Real-time applications, such as virtual reality experiences and interactive simulations, benefit from efficiently manipulating and rendering voxel-based data.

Despite their widespread use, challenges persist in optimizing voxel-based workflows. Managing large datasets, enhancing rendering efficiency, and addressing memory constraints remain ongoing considerations. Researchers and developers continually explore innovative solutions to unlock the full potential of voxels in creating immersive experiences and advancing our understanding of the three-dimensional world.

Voxels are a cornerstone of three-dimensional representation, finding applications in computer graphics, medical imaging, scientific simulations, and beyond. Their ability to encapsulate spatial information within a 3D grid allows realistic visualizations, accurate medical diagnostics, and in-depth scientific explorations. As

technology evolves, so does the potential for harnessing the versatility of voxels in creating immersive experiences and advancing our understanding of the three-dimensional world.

7.2. .VTU filetype

The .vtu file extension is linked with the Visualization Toolkit (VTK), a widely used open-source software system catering to 3D computer graphics, image processing, and visualization needs. Specifically, .vtu files form part of VTK's file format designed for unstructured grid data, commonly utilized for modeling intricate 3D geometries in diverse scientific and engineering simulations.

Within .vtu files, unstructured grid data is stored, depicting a spatial domain fragmented into irregularly shaped and sized elements. Unlike regular grids, this approach offers superior accuracy in modeling complex geometries. The .vtu format is Extensible Markup Language (XML) based, employing XML to articulate the data structure in human-readable and machine-readable formats [85].

These files encapsulate information about the mesh geometry and its associated data attributes. Mesh details comprise points, cells, and connectivity specifications, forming the foundation for constructing the 3D geometry. Data attributes represent diverse physical quantities, such as temperature, pressure, or velocity, linked to each point or cell in the mesh.

Originating from the VTK library, renowned for scientific visualization and computer graphics, .vtu files enjoy compatibility with this library. This integration facilitates seamless reading, writing, and visualization of unstructured grid data. Notably, .vtu files are frequently employed in simulations spanning fluid dynamics, structural mechanics, and other scientific domains, enabling the representation and analysis of intricate physical phenomena within irregular geometries [86].

The .vtu format's open nature and XML foundation encourage interoperability across different software tools supporting the VTK library. This interoperability proves valuable for exchanging simulation data and results across diverse scientific computing platforms. Visualization tools compatible with VTK can interpret .vtu files to generate

detailed visual representations of simulation results. Post-processing tasks, such as contour plot generation or animations, also leverage the data stored in .vtu files.

An essential feature of the .vtu format is its support for parallel processing. This capability allows simulations to efficiently harness multiple processors or computing nodes, enhancing computational speed.

The .vtu file format stands as a critical element within the VTK library, providing a standardized and versatile mechanism for storing unstructured grid data. Its prevalence in scientific simulations and visualization workflows underscores its significance in portraying and analyzing intricate 3D geometries and their associated physical attributes.

7.3. .VTK filetype

The .vtk file extension is also linked with the Visualization Toolkit (VTK). In the realm of .vtk files, structured grid data is encompassed, portraying a spatial domain systematically divided into regular grid elements. This grid structure, comprising points, cells, and associated attributes, establishes a systematic approach to representing 3D geometries in a precise and organized manner. The .vtk format utilizes both ASCII and binary encoding, offering flexibility regarding human readability and file size optimization.

Within a .vtk file, points delineate the coordinates in space, cells establish connectivity between points to create geometric shapes, and attributes associate data values with points or cells. These attributes can represent diverse physical quantities, such as temperature, pressure, or scalar fields, comprehensively representing simulated or measured phenomena [87].

The versatility and compatibility of the .vtk file format with various visualization tools and software packages are noteworthy. Its open nature allows seamless integration with different scientific computing platforms, enabling data exchange across diverse environments. Furthermore, the ASCII format of .vtk files facilitates manual inspection and editing, enhancing accessibility for researchers and engineers.

The .vtk file format finds applications in medical imaging, computational fluid dynamics, structural mechanics, and scientific visualization. Visualization tools compatible with VTK effortlessly interpret .vtk files, generating visual representations of the underlying data. This capability is invaluable for researchers, engineers, and scientists involved in analyzing and interpreting complex 3D geometries and associated attributes.

Additionally, the .vtk file format also supports parallel processing.

In short, the .vtk file format is a standardized and versatile medium within the VTK library, facilitating the storage and exchange of structured grid data for diverse scientific and engineering applications. Its widespread adoption in the visualization and simulation community underscores its significance in representing and analyzing 3D geometries and associated physical phenomena.

7.4. Unreal Engine

Unreal Engine, developed by Epic Games, stands as a pinnacle in the realm of game development frameworks. Since its inception in 1998, this engine has undergone significant evolution, emerging as a powerhouse in the industry known for its capacity to craft high-quality, visually stunning games and applications across diverse platforms. In this exploration, we delve into Unreal Engine's essential features, its profound impact on the gaming industry, and its expansive applications, mainly focusing on its transformative role in virtual reality (VR).

At its core, Unreal Engine offers game developers a comprehensive suite of tools. This includes Blueprints, a visual scripting system, a robust rendering engine, and advanced physics simulations. A defining feature is the engine's emphasis on real-time rendering, allowing developers to witness instant changes as they manipulate assets and code. This real-time feedback accelerates the development process, fostering creativity and iterative refinement.

Unreal Engine's prowess in graphics rendering is a standout feature. Employing a physically-based rendering (PBR) system, it authentically simulates how light interacts with materials, resulting in remarkably realistic visuals. This capability makes

Unreal Engine a preferred choice for developers aiming to create immersive and visually stunning environments.

A distinctive aspect of Unreal Engine is the Blueprints system, which offers a node-based visual scripting interface. Even those without extensive coding experience can use this system to create game logic, democratizing game development and encouraging innovation.

The engine's cross-platform compatibility is another strength, supporting various platforms, including PC, consoles, mobile devices, and virtual reality (VR). The engine's scalability ensures projects can be optimized for varying hardware specifications, from high-end gaming PCs to more modest mobile devices.

Unreal Engine has played a pivotal role in shaping the gaming industry. Titles developed using the engine consistently push the boundaries of visual fidelity, offering cinematic experiences. Games like the "Gears of War" and "Fortnite" series highlight the engine's adaptability to various genres, from intense action shooters to expansive open-world adventures.

Looking specifically at virtual reality (VR), Unreal Engine has become a driving force. Its support for VR platforms and its intuitive development environment have contributed significantly to the growth of VR gaming and applications. Developers leverage Unreal Engine to create immersive VR experiences, pushing the boundaries of what is possible in virtual realms [88].

Beyond gaming, Unreal Engine has found applications in industries such as architecture, automotive design, film and television production, and virtual production. Architectural visualization enables the creation of interactive virtual walkthroughs. In the automotive industry, it facilitates realistic simulations for testing and showcasing vehicles. Additionally, filmmakers use the engine for virtual production, previsualization, and creating entire digital sets.

In essence, Unreal Engine is a game-changer in modern game development, setting visual fidelity standards and offering creators versatile tools. Its influence extends beyond gaming, significantly impacting various industries, especially in the transformative realm of virtual reality. As technology advances, Unreal Engine remains

at the forefront, empowering creators to bring their imaginative visions to life, whether in the fantastical realms of gaming or the immersive landscapes of virtual reality.

7.5. Compatibility issues within Unreal engine and .vtk and .vtu files

Although both Unreal engine and .vtk and .vtu files use voxels, it is observed that there is no native support for these file types within Unreal engine, leaving the only options to use a third-party solution or to consider a custom solution.

After an extensive and, unfortunately, futile search for a third-party solution, it is decided to go with a custom solution.

The primary save option for Smartfire is .vtu, where .vtk is optional; it is decided to convert results as simply as possible, meaning a native .vtu file will be used.

All relevant data, smoke, temperature, fuel, pressures, and velocities relevant to this research are stored in .vtu files and must be custom-imported to the Unreal Engine.

After investigation, it is noted that the Unreal engine can import data from the .vdb file type, which is relatively similar to the .vtu file type, meaning that conversion should be possible.

7.5.1. .VDB file type

The .vdb file type is employed for storing volumetric data and is widely utilized in computer graphics, particularly in applications related to visual effects, simulation, and rendering. This file format is commonly associated with tasks like fluid dynamics simulations, where it efficiently represents three-dimensional (3D) voxel-based datasets.

Fundamentally, a .vdb file stores volumetric data in a manner that effectively captures 3D structures, often organized in a grid of voxels. Voxels, serving as

volumetric pixels, extend into the third dimension, enabling the representation of intricate 3D scenes such as smoke, fire, or volumetric effects.

Renowned for its efficiency, the .vdb format excels in handling large datasets, making it well-suited for scenarios requiring precise storage and processing of volumetric information. This is particularly crucial in applications that demand realistic simulations of phenomena like fluid behavior or complex particle systems.

In a .vdb file, information about voxels' density, color, temperature, or other physical attributes within the 3D grid is typically stored. These attributes define the characteristics of the volumetric data being represented. For instance, in a fluid simulation, density might represent fluid concentration at each voxel, while in a cloud simulation, it could signify the density of water vapor.

A notable feature of the .vdb file format is its support for sparse data structures, allowing it to efficiently represent volumetric data where much of the space is empty. This optimization aids in conserving storage and processing resources. The sparse nature of .vdb files enables the representation of intricate details in a scene without requiring excessive storage space.

Applications in the visual effects and computer graphics industries, such as Autodesk's Houdini and SideFX Houdini FX, commonly leverage the .vdb format for storing and exchanging volumetric data. The versatility and efficiency of .vdb files make them invaluable for diverse tasks, from creating realistic visual effects in films to conducting scientific simulations involving volumetric data representation.

7.5.2. Conversion of .VTU to .VDB

Research of the topic returned with the free-to-use software Paraview, which converts .vtu to .vdb.

7.5.2.1. ParaView software

ParaView is an open-source software solution designed for visualizing and analyzing intricate scientific, engineering, and academic datasets. Developed by Kitware, Inc., it has gained widespread acceptance due to its capabilities in handling large datasets derived from simulations, experiments, or measurements.

The software accommodates various data formats commonly used in scientific and engineering simulations. Its capacity to work with datasets from computational fluid dynamics (CFD), finite element analysis (FEA), medical imaging, and other scientific domains makes it a versatile choice.

A notable feature of ParaView lies in its ability to create interactive 3D visualizations of complex datasets. Users can actively explore and manipulate visual representations to gain insights into the underlying data.

ParaView has parallel processing capabilities, enabling users to harness high-performance computing resources for managing substantial datasets. This proves particularly beneficial for simulations generating massive amounts of data.

Flexibility is a crucial attribute of ParaView. Users can customize and extend the software by creating custom scripts, plugins, or workflows, tailoring it to specific requirements across various applications.

Cross-platform compatibility enhances accessibility, with ParaView supporting various operating systems, including Windows, macOS, and Linux. This adaptability accommodates users in diverse computing environments.

The software provides tools for performing quantitative analysis on datasets, allowing users to extract information, compute statistics, and visualize different aspects of the data for a comprehensive understanding.

Built on the Visualization Toolkit (VTK), ParaView leverages this robust open-source library for 3D computer graphics, image processing, and visualization. Integration with VTK enhances ParaView's capabilities and ensures efficient handling of visualization tasks.

The open-source nature of ParaView fosters a collaborative community of developers and users. This environment encourages continuous improvement, updates, and the sharing of knowledge and resources.

Scientific visualization is a primary application of ParaView across various disciplines, including physics, engineering, geoscience, and medical imaging. It assists researchers and scientists in gaining insights from complex datasets generated by simulations or experiments.

ParaView offers a user-friendly graphical interface for interactive exploration and a scripting interface for batch processing and automation. This dual-mode approach caters to diverse user workflows.

7.5.2.2. ParaView conversion

A simple setup was needed to convert the .vtu to .vdb file using ParaView. Data from one time-step is loaded into Paraview and visualized to check for data consistency (Figure 55).

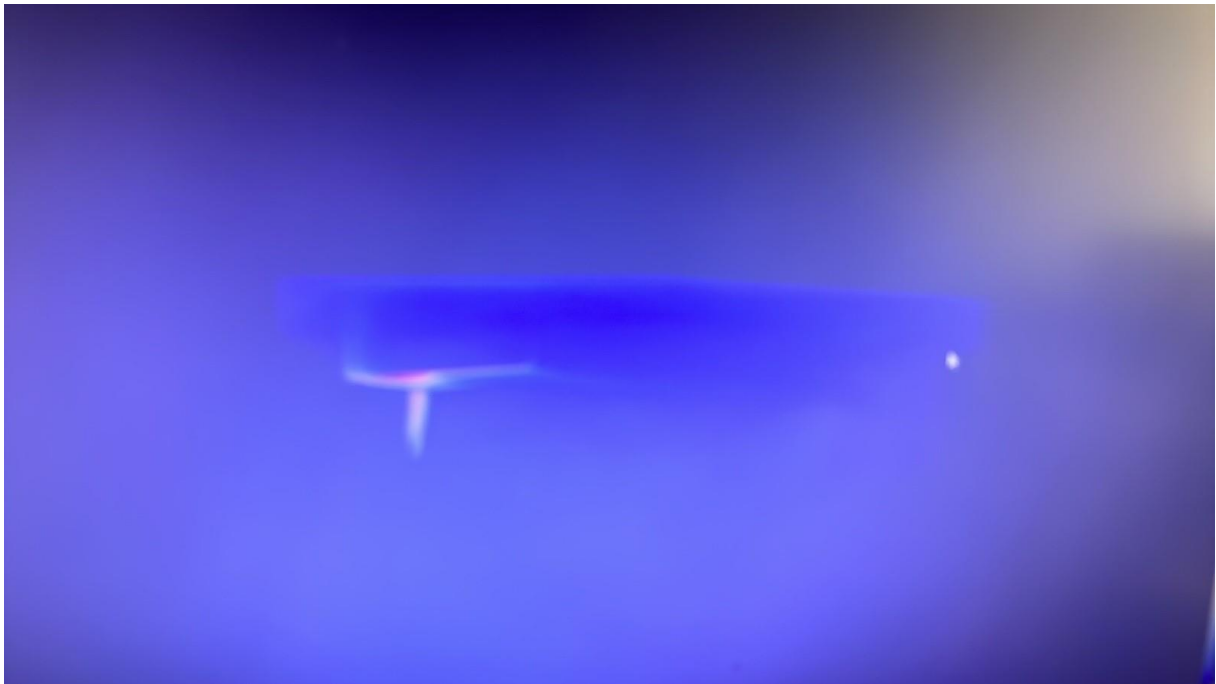


Figure 55 – ParaView preview of smoke data from .vtu file.

Simple “Save as” the option was selected, with the checked option “Choose arrays to write”. Only the smoke array was selected and saved at this point. The output file of the .vdb type was created successfully.

7.5.2.3. Import of .VDB file with smoke data to Unreal engine

At the time of first .vdb import Unreal engine 5.2. was the latest version and did not offer native .vdb import support. It was necessary to import it using several plugins and with the help of SideFX’s Houdini program.

7.5.2.4. SideFX’s Houdini

Houdini stands out as a robust 3D animation and visual effects (VFX) software developed by SideFX. Widely embraced in the entertainment industry, particularly in film, television, and video game production, Houdini is renowned for its capacity to craft impressive visual effects, animations, and simulations. Key aspects of Houdini include its procedural, node-based workflow, proficiency in dynamic simulations encompassing fluid dynamics, smoke, fire, cloth, and particles, and its excellence in character animation, rigging, and character effects.

What distinguishes Houdini is its procedural approach, where artists construct networks of nodes defining relationships and interactions between elements, allowing for heightened flexibility and control. The software excels in dynamic simulations, offering advanced particle systems for intricate effects and supporting character animation and rigging. Houdini is also a go-to choice for visual effects creation, providing tools for compositing 3D elements seamlessly into live-action footage [89].

Houdini FX, a specialized version for visual effects work, comes with additional features tailored to the needs of VFX artists. The software's simulation networks empower artists to create complex effects and animations by defining behaviors through rules and parameters. Due to its capabilities, Houdini has become an industry

standard, extensively used by film, television, and gaming professionals for a diverse range of projects demanding advanced simulations and visual effects.

The software's strength lies in its flexibility, enabling artists to experiment and iterate, ultimately achieving detailed and realistic results. Its procedural nature makes it adept at handling complex scenes and effects. It is a valuable tool for artists and studios involved in cutting-edge and demanding projects within the ever-evolving landscape of animation and visual effects.

7.5.2.5. Import of data into Houdini software

Houdini software recognized the .vdb file type and opened it natively. Immediately after opening the preview on the screen, it was observed that data was not represented correctly. It was represented as point data rather than voxels. After deeper investigation, it became clear that the properties of the .vdb file state that data is of voxel point data rather than voxel volumetric data, Figure 56, which was unacceptable.

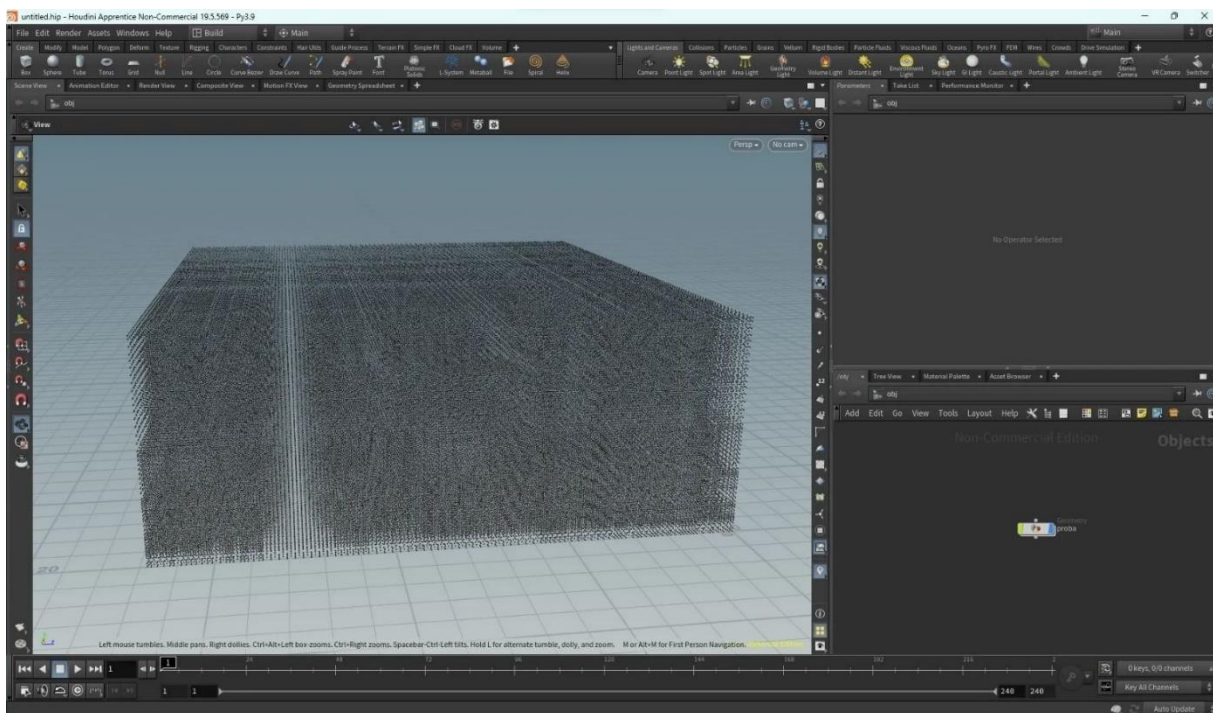


Figure 56 – Data representation in Houdini software.

Several more attempts were made to convert .vtu to .vdb using ParaView and import it into Houdini using different properties and data sets. The new, simple simulation was created with empty and symmetrical geometry but with the same result.

The external .vdb file was imported for testing purposes, and since it was a success, it was concluded that conversion from .vtu to .vdb was useless for the case.

A completely new approach was designed; it is a custom-made C++ script to convert .vtu to .vdb.

7.6. C++ code to convert .vtu data to .vdb

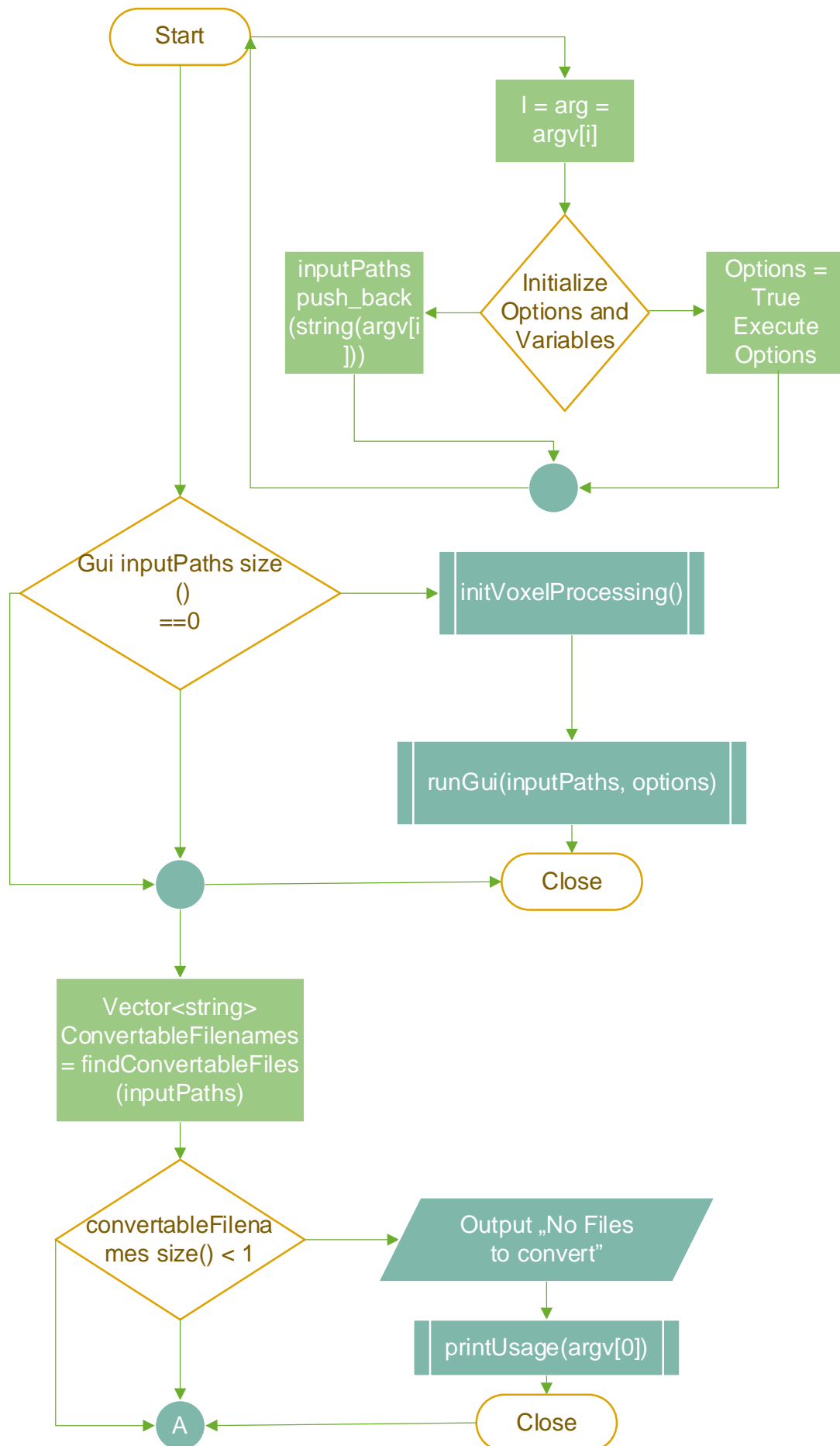
After investigating the .vtu and .vdb file types and their functionality, a code is written using C++ programming language to convert data from .vtu to .vdb file type.

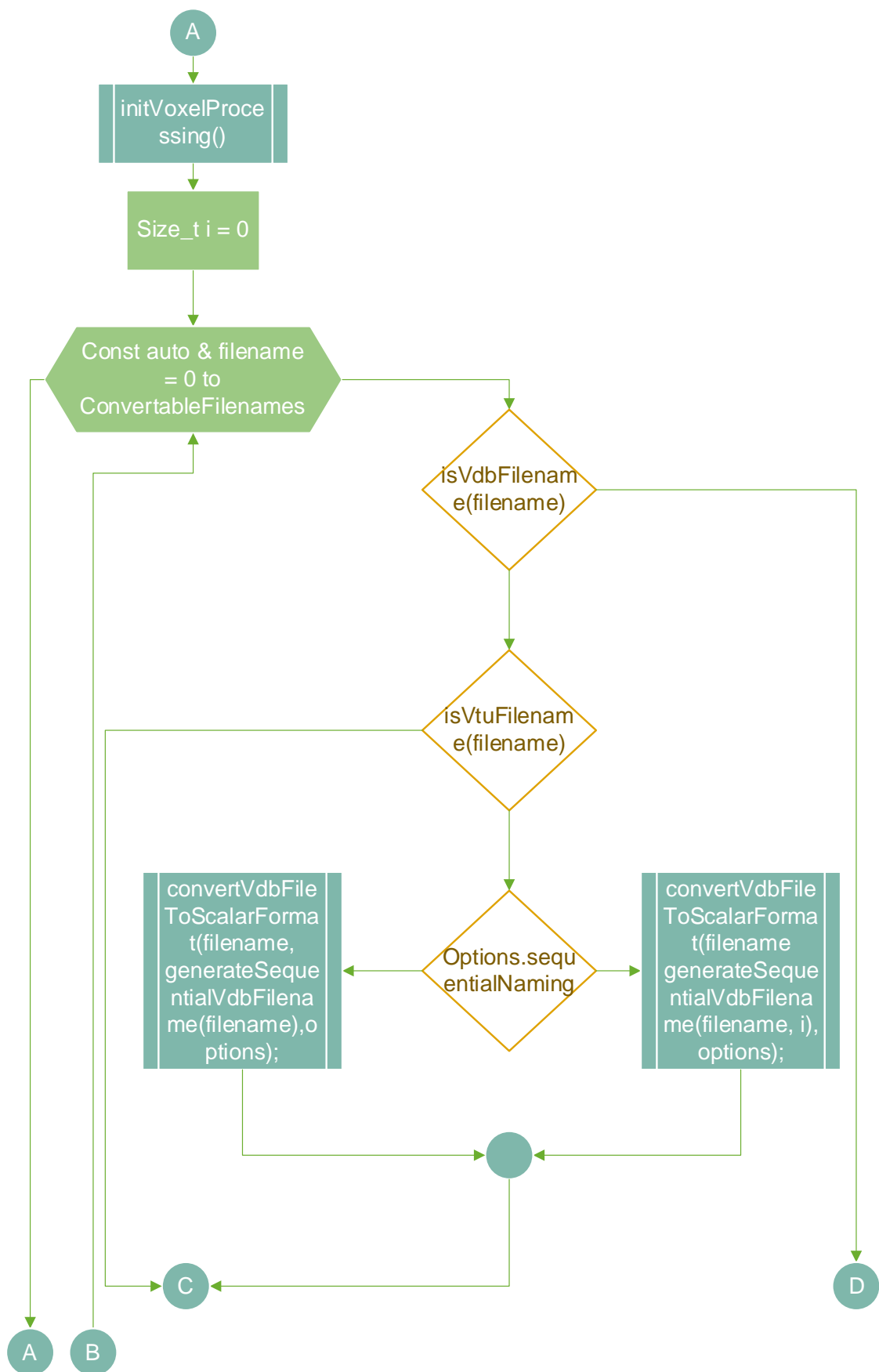
C++ emerged as a versatile programming language, evolving from the foundations of the C programming language. Conceived by Bjarne Stroustrup at Bell Labs in the early 1980s, C++ introduced object-oriented programming elements to C, enabling the utilization of procedural and object-oriented programming paradigms.

The code contains several files: gui.cpp, gui.h, main.cpp, options.h, utils.cpp, utils.h, voxels.cpp and voxels.h. Additional external sources are loaded: vdb_converter library, VTK, OpenVDB, raylib, openg132, gdi32, and winmm libraries.

7.7. Description of converter files

A short description of every file that makes a converter is to be presented with each file's essential functionalities and contents. Additionally, a simplified block diagram representing the basic functionalities of converter is shown in Figure 57.





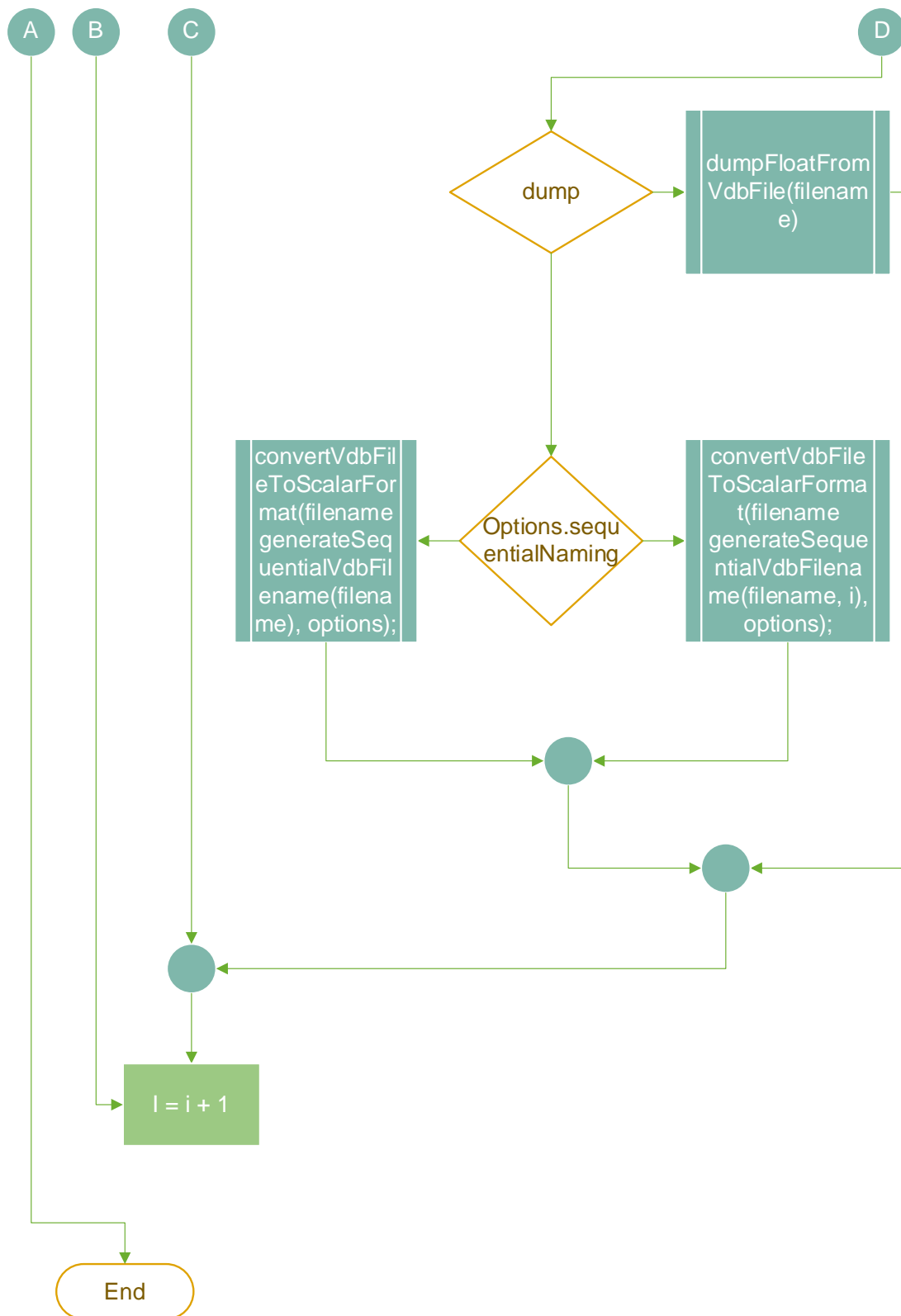


Figure 57 – Simplified block diagram of basic converter functionalities.

- **Main.cpp:**

This C++ code is a utility for handling and converting voxel data files through the command line. It includes headers for options, utility functions, voxel processing, and graphical user interface (GUI). The program parses command-line arguments, initializes options, and processes various options the user specifies. It also features a main function determining whether to launch the GUI or convert files in the command-line mode. The command-line arguments are iterated through, and corresponding options are set based on the user input. The program checks if the GUI mode is specified or there are no input files; in such cases, it initializes voxel processing and runs the GUI. Otherwise, it proceeds with file conversion. The program identifies its type (VDB or VTU) for each convertible file and performs the conversion accordingly, considering options like dumping voxel context, sequential naming, and field inclusion. This C++ utility offers a versatile solution for processing voxel data through both command-line and GUI modes, supporting customizable options and file formats.

- **Gui.cpp:**

This C++ code is a program segment designed as a GUI for a VDB/VTU file converter. The primary purpose of this program is to enable users to convert voxel data files between different formats through an intuitive graphical interface. The following provides an overview of the significant components and functionalities within the code:

The code includes necessary headers for options, utility functions, and voxel processing. Additionally, external libraries like Raylib and Raygui are incorporated to facilitate the graphical user interface implementation. Standard C++ libraries such as mutex, string, chrono, thread, iostream, and vector are also integrated.

A shared state is managed using a mutex and shared variables to synchronize information between the worker thread and the GUI. This shared state encompasses a queue of filenames slated for conversion, the current position in the conversion queue, and the total number of files converted.

The worker thread function, named `workerThreadRunnable`, is responsible for processing the queue of files designated for conversion. It continuously checks for files

in the queue, converts them based on their type (VDB or VTU), and updates the conversion statistics.

Utility functions like `DrawTextCentered` are provided for drawing-centered text on the GUI. The primary function of `runGUI` is running the graphical. It initializes the conversion queue with the initial paths provided and initiates the worker thread. The GUI window is created using `Raylib`, displaying information such as total files converted, main status, and progress updates. Users can drop files onto the GUI for conversion, and the dropped files are subsequently added to the processing queue. The GUI dynamically updates to reflect the ongoing conversion progress, allowing users to close the window once the conversion is complete.

In summary, this code exemplifies the implementation of a GUI for a VDB/VTU file converter, incorporating multi-threading to process files concurrently. The graphical interface offers an accessible means for users to interact with the converter, monitor conversion progress, and initiate file conversions effortlessly by dragging and dropping files onto the application window.

- **Gui.h**

This code segment serves as a C++ header file with essential declarations and includes a graphical user interface (GUI) component within a larger program. It employs a `"#pragma"` once directive to ensure single inclusion during compilation, guarding against potential issues.

The header includes declarations for the `"options.h"` header file, containing elements influencing program behavior. Additionally, standard C++ library headers for string and vector operations are included.

The line using namespace `std` brings the entire `std` namespace into the current scope, simplifying references to standard library components.

The header declares a function named `"runGui"` with two parameters:

`"initialPaths"`: A vector of strings representing initial paths, such as file or directory paths, provided to the GUI for processing.

Options “Options”: An object of type Options containing information or settings pertinent to the GUI's behavior.

In essence, this header file establishes the framework for the runGui function, defines its parameters, and incorporates the necessary dependencies for its implementation, which is expected to be provided in a corresponding source file.

- **Options.h**

Code comprises a C++ header file that introduces a struct named Options. The struct is designed to store various configuration settings for a larger program. The file begins with a #pragma once directive, serving as a header guard to ensure single inclusion during compilation.

Standard C++ library headers for utilizing std::map, std::set, and std::string are included. These containers are anticipated for use within the Options struct.

A macro, DEFAULT_VTU_VOXEL_RESOLUTION_MULTIPLIER, is defined with a value of 1. Macros are employed to define constants or perform text substitutions during preprocessing.

The core of the code is the Options struct, encapsulating different configuration parameters. It includes Boolean flags like verbose for enabling verbose logging and sequentialNaming to indicate whether sequential file naming should be applied. Additionally, it contains containers like std::set<std::string> includedFields to store names of fields to be included, and std::map<std::string, float> normalizedFields associating field names with normalized 1-values.

Numeric parameters, such as resolutionMultiplier and fixedResolution, offer control over voxel resolution when converting from VTU to VDB. This struct provides a structured and organized representation of the program's configuration options.

- **Utils.cpp**

Given C++ code functions as a utility for managing file names and paths related to VDB (Voxel Database) and VTU (VTK Unstructured Grid) files.

The code starts with typical C++ includes, a pragma directive for header protection and using namespace std; statement. It defines global string variables that specify common file suffixes and prefixes.

Subsequently, there's a utility function called endsWith designed to ascertain whether a given string ends with a specified suffix.

Various functions are implemented to handle file naming conventions:

isVdbFilename: Checks if a file name concludes with ".vdb",

isConvertedVdbFilename: Verifies if a file name denotes a converted VDB file,

generateConvertedVdbFilename: Constructs a converted VDB file name from an original file name,

isVtuFilename: Determines if a file name concludes with ".vtu",

generateConvertedVtuFilename: Creates a converted VTU file name from an original file name,

generateSequentialVdbFilename: Constructs a sequential VDB file name with a frame index.

Additionally, the code incorporates a function, findConvertibleFiles, which takes a vector of file paths, filters out files that don't meet specific criteria (e.g., existence, type), sorts the remaining paths, and returns them as convertible files.

In essence, this utility code aids in identifying and analyzing VDB and VTU files within a larger program focused on file conversion.

- **Utils.h**

It defines a set of utility functions of file naming and identification within the context of VDB (Voxel Database) and VTU (VTK Unstructured Grid) files. The functions include:

isVdbFilename: Takes a file path and determines if it has a ".vdb" extension, returning a Boolean.

`isConvertedVdbFilename`: Given a file path, checks if it is a converted VDB file by verifying the ".vdb" extension and a specific prefix ("converted_").

`generateConvertedVdbFilename`: Accepts an original file name and produces the converted VDB file name by appending a prefix ("converted_").

`isVtuFilename`: Examines a file path to ascertain if it has a ".vtu" extension, returning a Boolean.

`generateConvertedVtuFilename`: Takes an original file name and generates the converted VTU file name by replacing the ".vtu" extension with ".vdb" and appending a prefix ("converted_").

`generateSequentialVdbFilename`: Given an original file name and a `size_t` index, constructs a sequential VDB file name with a frame index.

`findConvertibleFiles`: Takes a vector of file paths, filters them based on specific criteria such as existence and type (VDB or VTU), sorts the remaining paths, and returns them as a vector.

Collectively, these functions serve as tools for managing file names and identifying files related to VDB and VTU formats.

- **Voxels.cpp**

It is a comprehensive utility for converting VDB (Voxel Database) and VTU (VTK Unstructured Grid) files. It utilizes key header files related to VDB, VTU, and external libraries, such as OpenVDB and VTK (Visualization Toolkit).

The code includes several utility functions to enhance functionality and readability. For example, the `describeGrid` function outputs detailed information about an OpenVDB grid, covering its type, value type, creator, class, world space status, and bounding box. The `FloatGridInfo` struct encapsulates details about a float grid, including its name, attribute index, and a pointer to the actual float grid. Another utility function, `extractFloatGridsFromIndexGrid`, is designed to extract float grids from an index-based VDB grid based on specified options.

Additionally, there's the `initVoxelProcessing` function, which is responsible for initializing voxel processing libraries, particularly OpenVDB.

The code also features conversion functions:

The `convertVdbFileToScalarFormat` function converts a VDB file containing non-float grids into one containing only float grids. It processes each grid in the input file, Extracting and converting relevant information based on specified options.

The `convertVtuFileToScalarVdbFormat` function converts a VTU file to a VDB file with float grids. It involves metadata examination, cell scanning, resampling, and writing the output.

The `dumpFloatVoxelsFromVdbFile` function serves a debugging/verification purpose, providing information about float voxels in a VDB file, including details like voxel coordinates and corresponding scalar values.

The main function demonstrates the usage of these conversion functions. It initializes voxel processing libraries, performs float grid extraction and conversion operations, and handles attributes, resampling data, and writing the results to a new file. The code heavily relies on the capabilities of OpenVDB and VTK libraries for voxel and visualization operations.

In essence, the code represents a robust and versatile tool for handling voxel data, facilitating the transformation between different file formats, and providing insights into the voxelized representation of complex geometrical structures.

- **Voxels.h**

This file includes the necessary header files, particularly focusing on VDB (Voxel Database) and VTU (VTK Unstructured Grid) files. It defines functions for voxel processing and file format conversion. Let's break down the code:

The `#pragma once` directive ensures that the header file is included only once during compilation, preventing multiple inclusion issues.

The `#include "options.h"` line includes the "options.h" header file, likely containing declarations related to voxel processing and file conversion settings.

The `using namespace std;` statement brings the entire `std` namespace into scope, facilitating the use of standard C++ entities.

The `void initVoxelProcessing();` function initializes voxel processing libraries or performs necessary setup for voxel-related operations.

The `void convertVdbFileToScalarFormat(string inFilename, string outFilename, Options options);` function converts a VDB file (likely containing non-float voxel data) to a scalar format. It takes input and output filenames and an `Options` object representing various settings.

The `void dumpFloatVoxelsFromVdbFile(string inFilename);` function is for debugging or verification purposes. It outputs information about float voxels from a VDB file, taking the input filename as a parameter.

The `void convertVtuFileToScalarVdbFormat(string inFilename, string outFilename, Options options);` function converts a VTU file to a VDB file with scalar format. Like the first conversion function, it takes input and output filenames and an `Options` object.

In essence, this code provides functions for initializing voxel processing, converting VDB files to scalar format, dumping information about float voxels in a VDB file, and converting VTU files to VDB format with scalar representation. The actual implementation of these functions is found in another file or files, and the provided declarations serve as a public interface for using these functionalities.

7.8. Import of converted data into Unreal engine

Setting saves of Smartfire simulation to every time step, with 0.2 seconds time step and 600 seconds of total simulation time, resulted in 3000 save files, or .vtu files, with an overall size of 60Gb. During conversion, only smoke data was extracted and saved inside .vdb files to save on the data size needed to import.

The result was 3000 .vdb files with a size of 3 GB. That amount of data is a challenge for hardware to process, so at this moment, increasing data volume is questionable.

During the code creation process for data conversion, a new version of Unreal engine was released, version 5.3, offering native .vdb import support, making plugins and Houdini software used in version 5.2 unnecessary.

Import was done with no issues, but some bugs were observed. Smoke data from the .vdb file was imported and represented correctly, but when the VR version was created, the image was displayed only on the right eye; no smoke data was displayed on the left eye.

This matter was investigated thoroughly, and it was concluded that it is a bug in the newest version and is highly probable to be repaired. A ticket for the issue was sent to the manufacturer.

Still, the displayed data was usable, so research continued.

Figure 58, Figure 59, and Figure 60 show the final result of data imported into the VR engine room in different scenarios.



Figure 58 – CFD imported smoke data of fire of the main engine fuel line scenario to the VR engine room.



Figure 59 – CFD imported smoke data of fire of oily rags left in a bucket scenario to the VR engine room

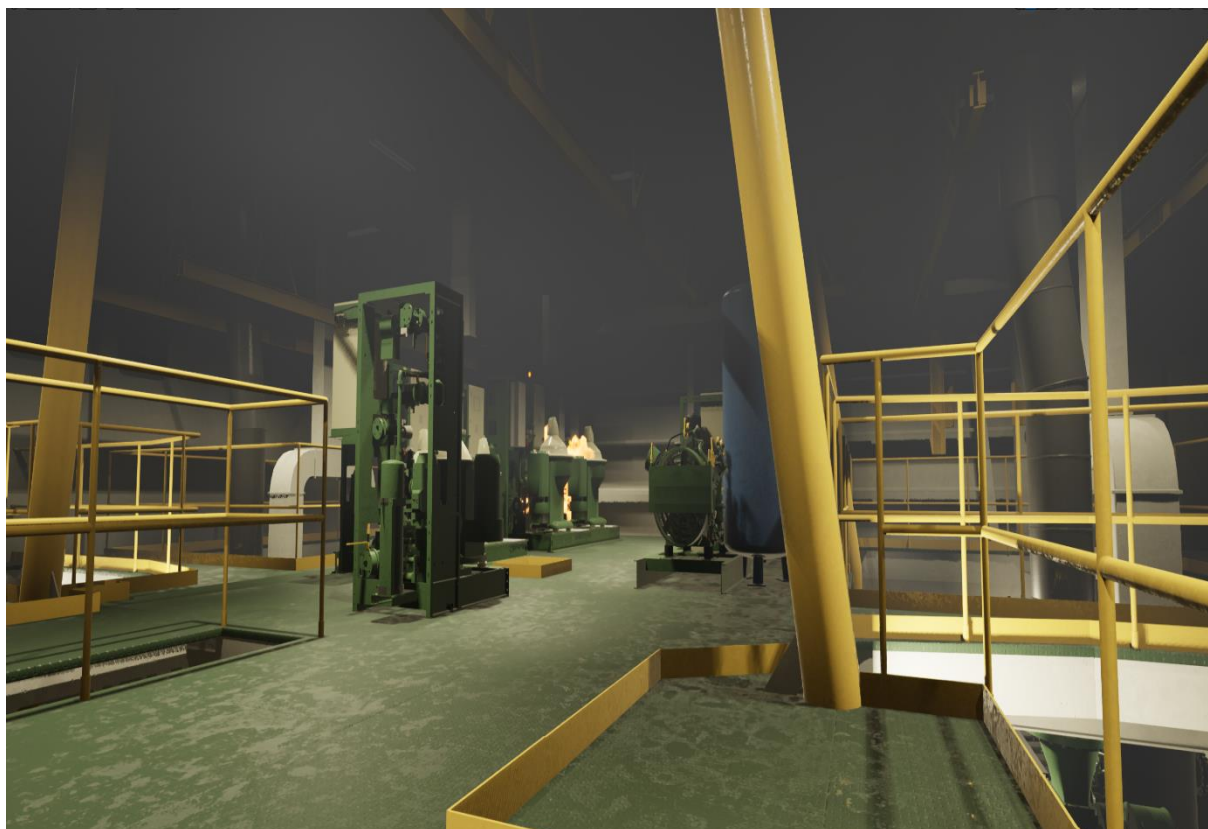


Figure 60 – CFD imported smoke data of fire of the fuel oil purifier scenario to the VR engine room

8. USER EXPERIENCE OF ADVANCED VR FIRE SPREAD MODEL

A survey is a research method used for collecting data from a predefined group of respondents to gain information and insights on various topics of interest. Thus, it is selected as a method for user experience measurement.

For the current software limitations with left eye display, it is decided not to use the VR version of the simulation on test subjects. The full VR experience is not possible in current conditions; it is rather decided to create two videos: the first one to display CFD-based smoke imported to Unreal engine, and the second one with regular game smoke that can be found in the Unreal Engine library, after which several questions will be asked to survey participants.

Both videos will be played to all of the test subjects, with half of the group viewing the first video and answering questions, the second one next, and the other half in the opposite order.

As the target population selection must be as narrow as possible [90], and the theme of the survey is the fire in a ship's engine room, two professions are naturally emerging for survey participation: experienced seamen who serve in the engine room and firefighters, as both groups can come across fire in the engine room and possibly be endangered by it.

Two test subject groups are to be created: one group will be populated with professional firefighters, and the other group will be populated with experienced seamen serving in the engine room.

Because of the nature of their work, experienced seamen are often aboard ships, and researcher-administered questionnaires are challenging. For that reason, the questionnaire is to be written rather than verbal or mixed, and for the seamen, it will be self-administered and delivered using e-mail, and for firefighters, it will be researcher-administered in their workplace, using several laptop computers.

Questions are to be feasible, and respondents will have the right not to participate. Questions will be civil and ethical [90].

The questionnaire is to be a mix of close-ended and open-ended questions.

8.1. Contents of the questionnaires

Questionnaires for both test subject groups are not the same, although differences are minor. For the measuring instruments, there are to be initial questions about gender, age, and years of experience in the firefighting profession, namely experience with indoor ship fires and the use of fire extinguishers. After initial questions, each respondent will be asked seven identical questions following two presented video clips. They were required to respond by indicating to what extent they agreed with each statement, marking numbers from 1 - disagree entirely to 6 - completely agree. The questions were related to the respondent's perception of the realism of the situation, namely the smoke, and the possibility of learning and coping in the depicted situation.

The procedure for each respondent is to show two video clips: 1. A generic representation of smoke spread obtained by game engine; 2. A representation of smoke spread according to the CFD model. The order of the clips was rotated among the respondents according to a random schedule. After each video clip presentation, seven questions were asked about the realism of the situation and learning/coping in the depicted situation.

The questions are shown in Table 4 and Table 5.

Table 4 – First set of questionnaire questions.

Age:	—
Gender (if applicable; otherwise omit)	M/F
Years of experience in firefighting: (for firefighters only)	—
Years of experience working on a ship: (for seafarers only)	—
Have you ever been in a fire situation inside a ship?	YES/NO
If yes, please specify how many times:	—
If yes, did you use a fire extinguisher in that situation?	YES/NO

The second set of questions is based on video-watching experience:

Table 5 – Second set of questionnaire questions.

The situation presented is realistic.	1 2 3 4 5 6
The depicted fire seems too artificial to me.	1 2 3 4 5 6
The way the smoke spreads in the room seems realistic to me.	1 2 3 4 5 6
The depicted smoke seemed too artificial to me.	1 2 3 4 5 6
I have encountered smoke similar to the one shown before.	1 2 3 4 5 6
I believe that the presented situation (for VR: the displayed task) could help me practice dealing with such situations in reality.	1 2 3 4 5 6
Considering my previous experience, I believe I could handle solving the situation shown in the video (VR: in the task).	1 2 3 4 5 6
Please describe in a few words why you stated that the presented situation is realistic or unrealistic.	_____

8.2. Survey results

In the study, 22 firefighters and 33 male sailors participated. The average age of the firefighters was 42.23 (SD = 10.85; min = 22, max = 60). The average age of the sailors was 42.55 (Standard Deviation (SD) = 7.44; min = 30, max = 57). The average number of years of work experience for firefighters was 19.18 (SD = 11.21; min = 0, max = 35). Out of the total 33 sailors, 28 had experienced a fire in the interior of a ship at least once, nine had such experience two or more times, and three had such experience three or more times. Twenty-five of them used a fire extinguisher in the mentioned situation. Out of 33 sailors, 22 provided responses after both video clips.

8.3. Firefighter results

In Table 6, there is descriptive data related to the firefighters' responses to all seven questions after each video clip; a graphic representation can be found in Figure 61. Since the answers to 5 out of 14 questions posed showed significant correlations with the years of professional experience of firefighters (significant correlations ranged from .47 to .56), and considering there is a large variation in the years of professional

experience, the data were further analyzed only among respondents with more than ten years of experience. Additional data are presented, and all analyses were repeated on the 15 more experienced firefighters (average years of experience: 25.40; SD = 7.05; min = 15, max = 35) out of a total of 22 firefighters (average age for the seven omitted = 5.85; SD = 4.45; min = 0, max = 10).

Table 6 - Arithmetic means and standard deviations of responses to 7 questions asked after each of 2 video clips, the first containing generic smoke and the second CFD modeled smoke, for all firefighters (N=22) and the group of experienced firefighters (N=15)

	All firemen AM/(SD)		Experienced firemen AM/(SD)	
	Generic	CFD	Generic	CFD
1. The situation presented is realistic.	3.67 (1.35)	3.64 (1.76)	3.63 (1.28)	4.20 (1.57)
2. The depicted fire seems too artificial to me.	3.77 (1.66)	3.45 (1.68)	3.67 (1.68)	2.93 (1.49)
3. The way the smoke spread in the room seems realistic to me.	3.67 (1.24)	3.52 (1.63)	3.71 (1.07)	4.00 (1.46)
4. The depicted smoke seemed too artificial to me.	3.05 (1.56)	3.32 (1.67)	2.80 (1.42)	2.73 (1.44)
5. I have encountered smoke similar to the one shown before.	3.45 (1.84)	3.67 (1.80)	3.53 (1.68)	4.36 (1.60)
6. I believe that the presented situation (for VR: the displayed task) could help me practice dealing with such situations in reality.	4.05 (1.65)	4.09 (1.48)	3.93 (1.71)	4.40 (1.30)
7. Considering my previous experience, I believe I could handle solving the situation shown in the video (VR: in the task).	5.29 (1.06)	5.00 (1.30)	5.29 (1.07)	5.35 (1.08)

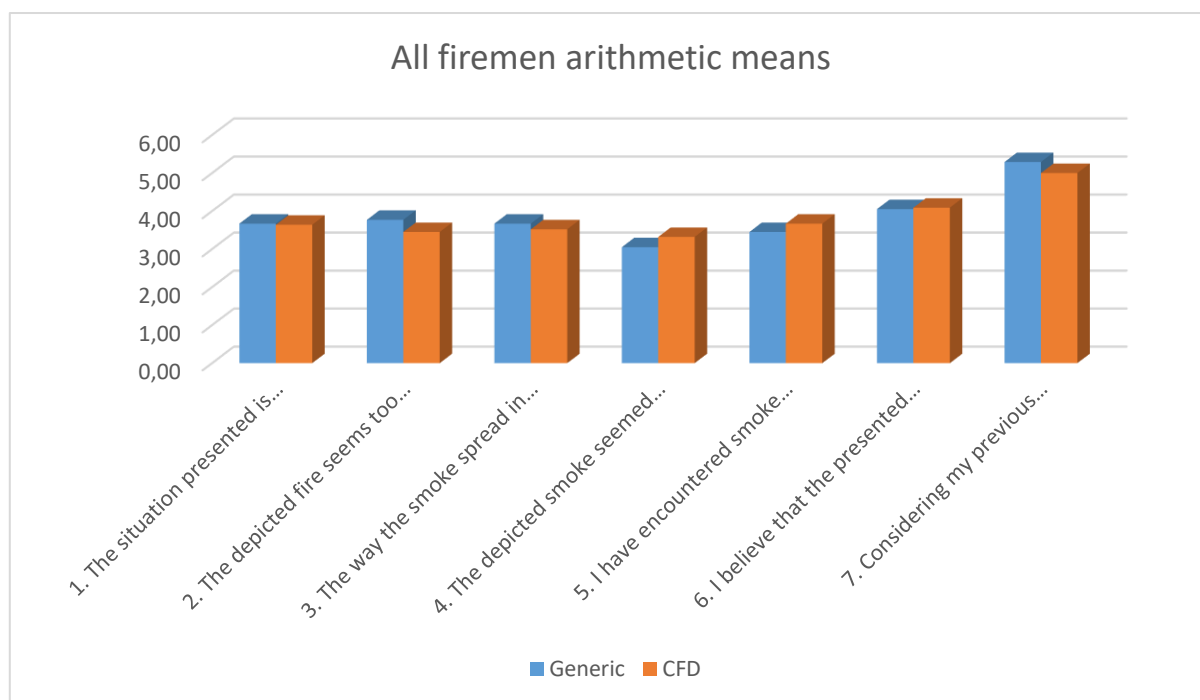


Figure 61 – All firefighters arithmetic means results.

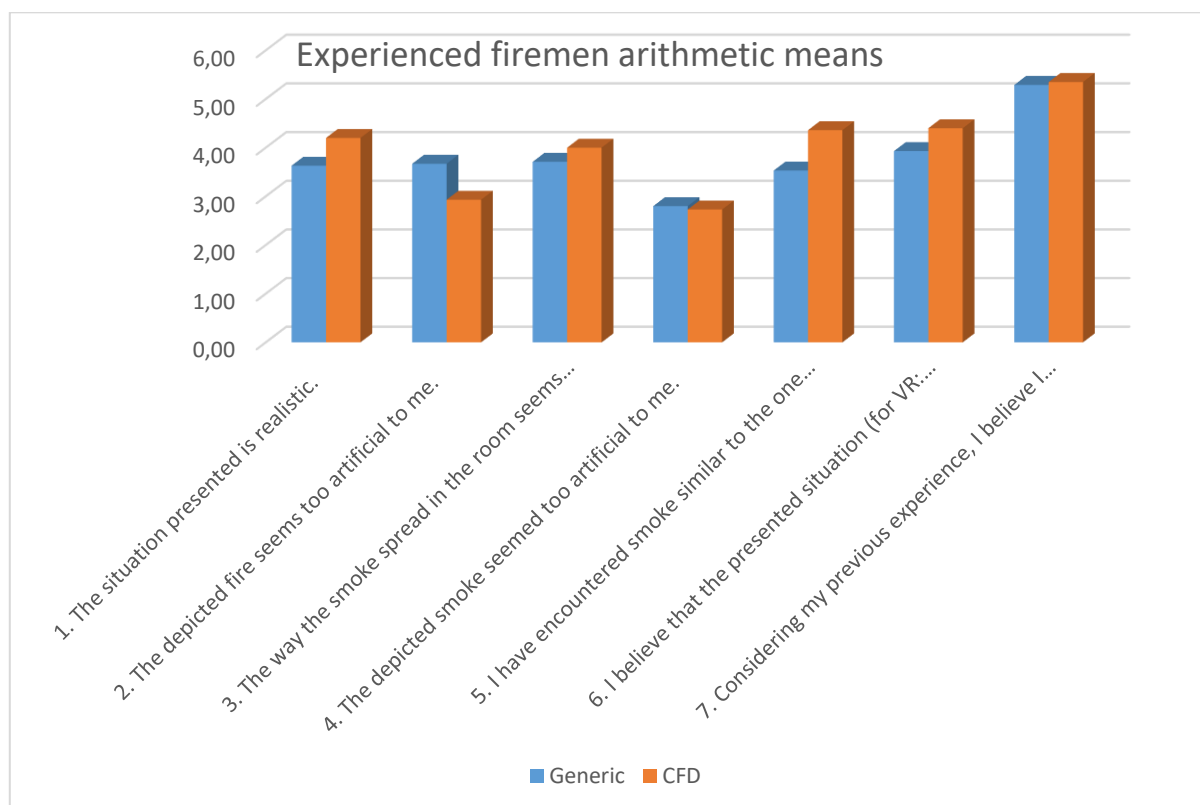


Figure 62 - Experienced firefighters arithmetic means results.

A series of repeated measures analyses of variance (ANOVA) was calculated for each posed question to compare the respondents' answers after the two video clips. The first level of the variable pertains to the video with generic smoke, and the second

to the video with CFD-modeled smoke, with control of the order in which the video clips were presented. When analyses were made on all firefighters, no effect proved statistically significant. When the analyses were done on the more experienced firefighters, the only statistically significant effect was the type of smoke on the responses to question 5 $F(1,12) = 6.35$; $p = .027$; $\eta^2p = 0.35$ (F-statistic, P-value and Analysis of Variance). Firefighters more frequently indicated that they had previously encountered smoke that looked like the smoke just shown when it was CFD smoke as opposed to generic smoke. Although none of the differences in responses to the remaining six questions were statistically significant ($p > .05$), given the small number of respondents, the differences can be commented upon with certain caution at the descriptive level. In this context, it should be emphasized that the differences in arithmetic means of responses to all six questions are in the direction of a greater perception of realism and more appropriate reactions regarding video clips that show CFD smoke.

8.4. Seamen results

In Table 7, there is descriptive data related to the seamen's responses to all seven questions asked after each video clip. Graphical representations can be seen in Figure 63 for all seamen and Figure 64 for seamen with fire experience.

Table 7 - Arithmetic means and standard deviations of responses to 7 questions asked after each of the 2 video clips, the first containing generic smoke and the second CFD modeled smoke, for all sailors (N=28) and for the group of seamen who have experienced fire onboard a ship at least once (N=23).

	All seamen AM/(SD)		With fire experience AM/(SD)	
	Generic	CFD	Generic	CFD
1. The situation presented is realistic.	3.21 (1.60)	3.36 (1.32)	3.12 (1.70)	3.20 (1.40)
2. The depicted fire seems too artificial to me.	3.93 (1.49)	3.92 (1.35)	4.00 (1.60)	4.05 (1.47)

3. The way the smoke spreads in the room seems realistic to me.	3.11 (1.63)	3.03 (1.54)	3.00 (1.68)	2.81 (1.60)
4. The depicted smoke seemed too artificial to me.	3.46 (1.57)	3.54 (1.58)	3.58 (1.64)	3.57 (1.69)
5. I have encountered smoke similar to the one shown before.	2.38 (1.50)	2.61 (1.68)	2.36 (1.47)	2.50 (1.65)
6. I believe that the presented situation (for VR: the displayed task) could help me practice dealing with such situations in reality.	3.48 (1.74)	3.27 (1.66)	3.26 (1.79)	2.90 (1.64)
7. Considering my previous experience, I believe I could handle solving the situation shown in the video (VR: in the task).	5.12 (1.24)	5.26 (1.08)	5.15 (1.25)	5.35 (1.18)

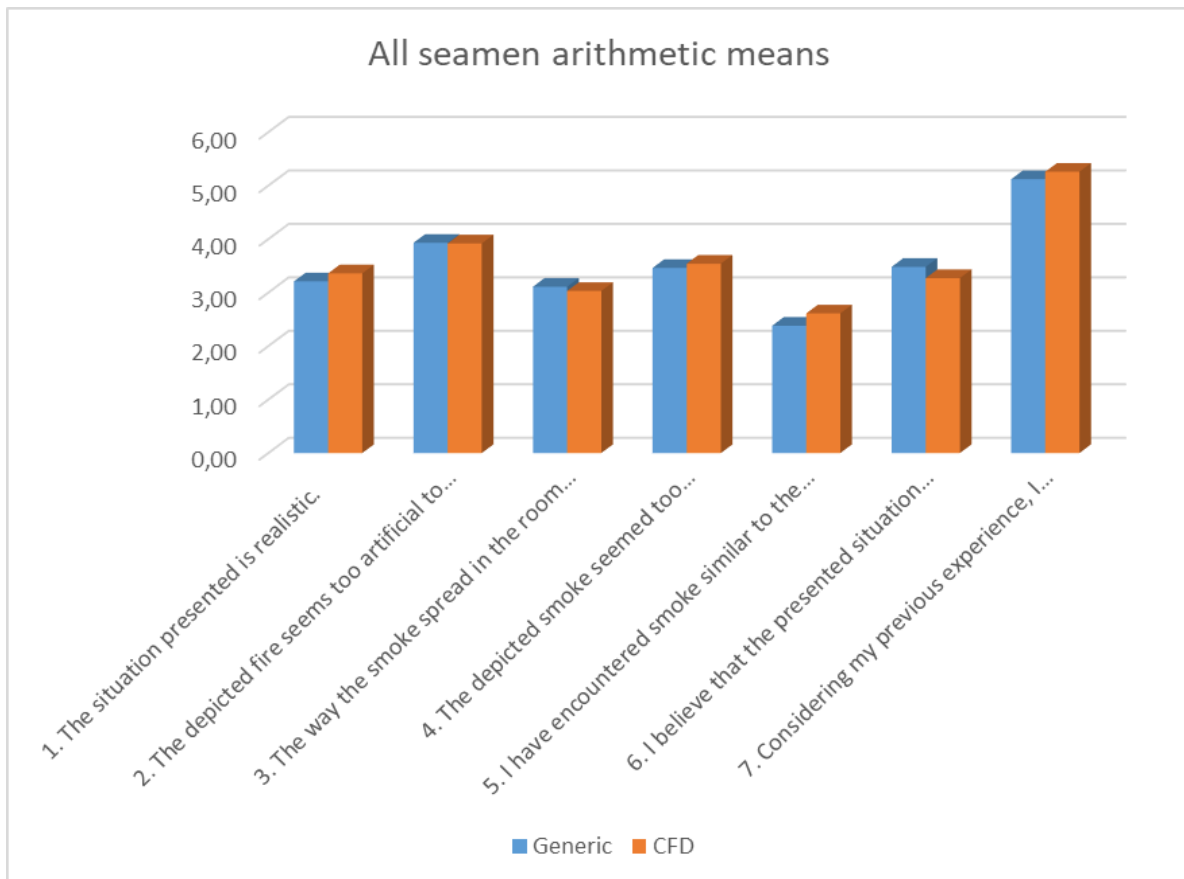


Figure 63 - All seamen arithmetic means results.

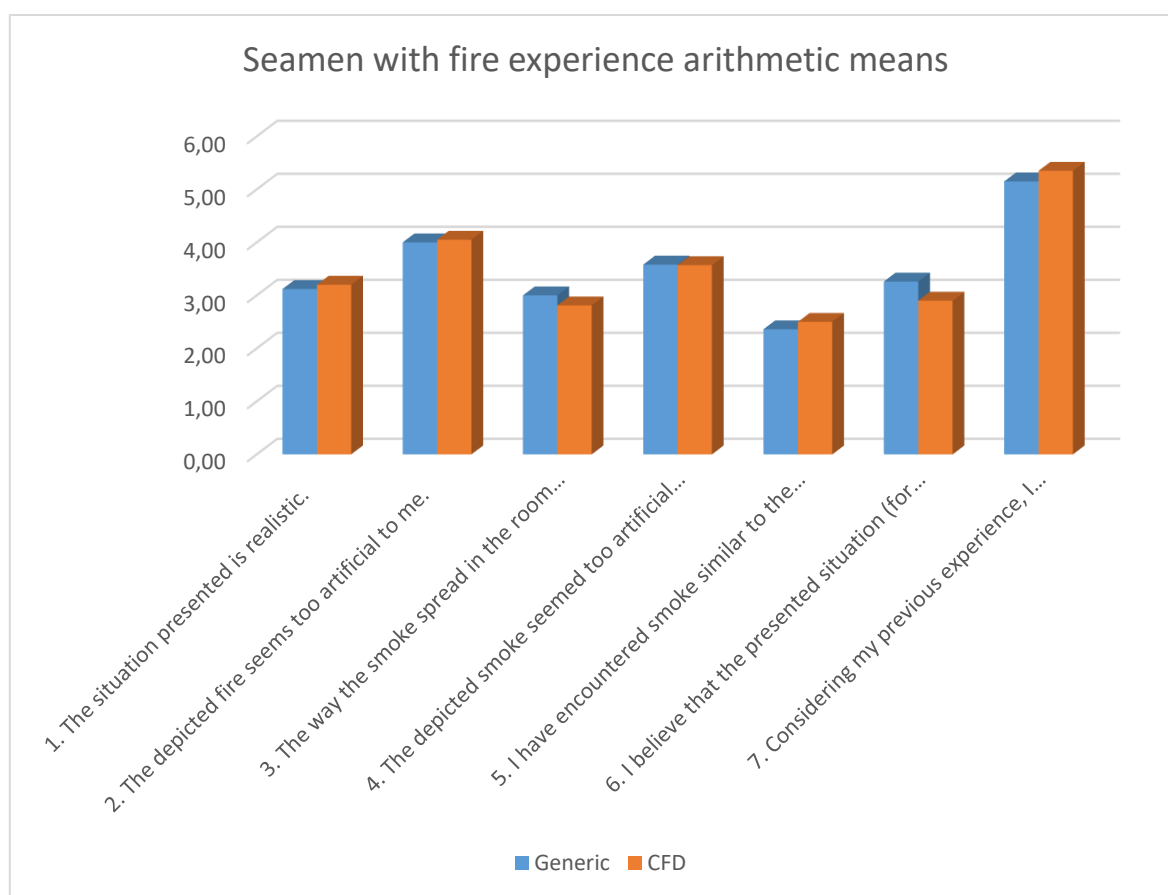


Figure 64 – Seamen with experience, arithmetic means.

To compare the respondents' answers after the two video clips, a series of analyses of variance (ANOVA) was calculated in the same manner as with the firefighters. When the analyses were conducted on all the sailors (that is, the 22 who provided answers after both clips), the only significant effect was the type of smoke on the responses to question 7 $F(1,18) = 5.23$, $p = .035$; $\eta^2p = 0.22$ (F-statistic, P-value and Analysis of Variance). The sailors reported that they would be better able to handle the situation shown in the video when it involved CFD smoke as opposed to generic smoke, based on their previous experience. Since five sailors reported that they had not previously experienced a fire inside a ship, all analyses were repeated, excluding their responses. The results obtained were identical to the previous analyses. Although none of the differences in responses to the remaining six questions were statistically significant ($p > .05$), given the small number of respondents, it is possible to comment on the differences at a descriptive level. The general pattern can be described as the absence of meaningful differences.

Since only 17 sailors with fire experience provided answers after both clips and 33 sailors provided answers to at least some of the questions, additional analyses were performed only on the responses after the first clip, primarily because there was a dropout of respondents after the second clip. In other words, between-subject analyses were performed, which represents a stricter check for the existence of differences between responses to the video clips, but in this case, allows for the calculation of results on the largest possible number of respondents (28 respondents after excluding five sailors who had no fire experience). Table 8 contains descriptive data related to the responses of sailors who had experienced a fire to all seven questions posed after the first video clip, which showed either generic or CFD smoke. The graphical representation is in Figure 65.

Table 8 - Arithmetic means and standard deviations of responses to 7 questions that were asked after the first video clip showing either generic (N = min 15, max 16) or CFD smoke (N = min 9, max 12) to seamen who had previous experience with fire in the interior of a ship.

	Generic AM/(SD)	CFD AM/(SD)
1. The situation presented is realistic.	2.81 (1.80)	3.55 (1.21)
2. The depicted fire seems too artificial to me.	4.19 (1.60)	3.92 (1.24)
3. The way the smoke spread in the room seems realistic to me.	2.56 (1.71)	3.00 (1.54)
4. The depicted smoke seemed too artificial to me.	3.69 (1.74)	3.50 (1.57)
5. I have encountered smoke similar to the one shown before.	1.80 (1.08)	2.78 (1.56)
6. I believe that the presented situation (for VR: the displayed task) could help me practice dealing with such situations in reality.	2.81 (1.90)	3.25 (1.54)

7. Considering my previous experience, I believe I could handle solving the situation shown in the video (VR: in the task).	5.20 (1.32)	5.45 (0.69)
---	-------------	-------------

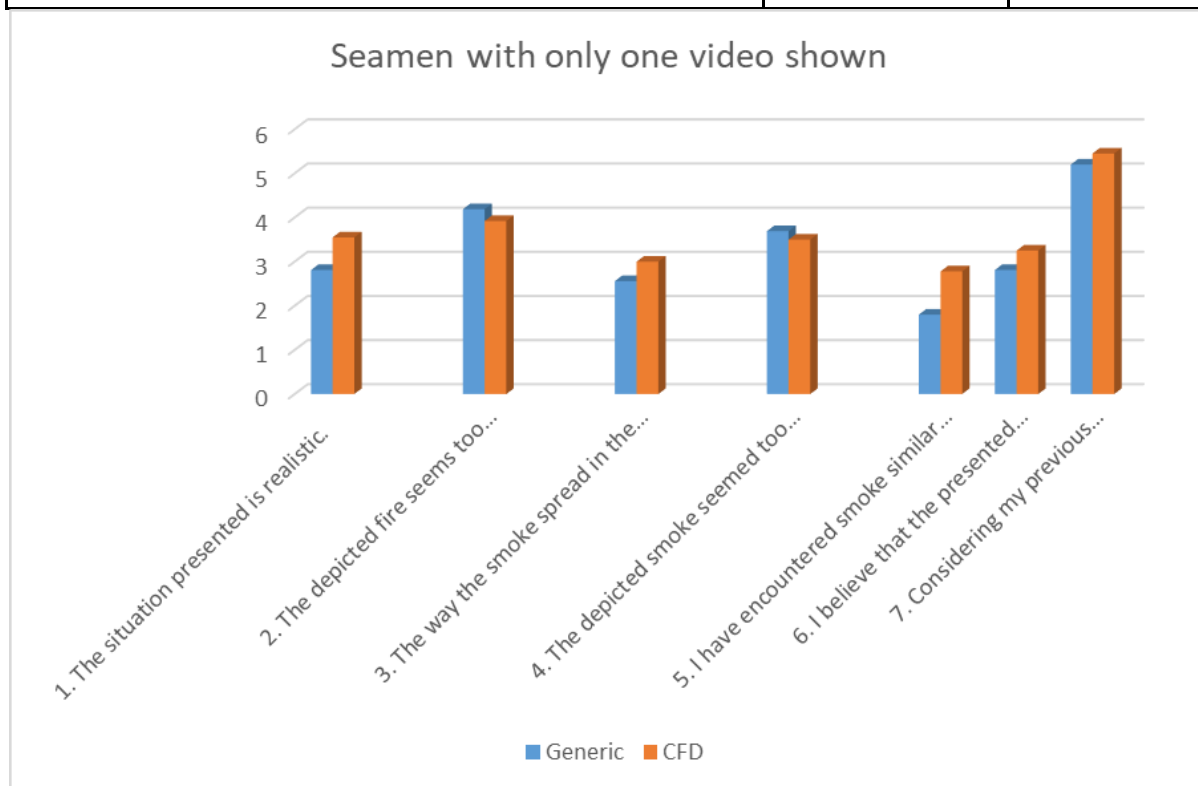


Figure 65 – Seamen with only one video shown, and had fire experience.

To compare the responses of participants who watched the clip with generic and with CFD smoke, a series of independent samples t-tests was calculated. No effect proved to be statistically significant. However, the effect of the type of smoke on responses to question 5 was marginally significant: $t(22) = 1.81$, $p = .083$; $d = 0.73$ (t-statistic, p-value and Cohen's d). As in the case of firefighters, sailors more frequently indicated that they had previously had the opportunity to encounter smoke that looked like the smoke just shown when it was CFD smoke as opposed to when it was generic smoke. Although none of the differences in responses to the remaining six questions were statistically significant ($p > .05$), given the small number of respondents, as before, the differences can be commented upon with certain caution at the descriptive level. In this context, it should be emphasized that the differences in arithmetic means of responses to all six questions are in the direction of a greater perception of realism and more appropriate reactions regarding video clips that show CFD smoke.

8.5. Conclusion of survey results

The observed differences in reactions to the two video clips are relatively small and, in most cases, statistically insignificant. However, when observed, they are consistent both in terms of statistical significance and at a descriptive level, and both among firefighters and sailors are in the direction of more appropriate evaluations for the video with smoke based on the CFD model. It is highly likely that with a larger number of respondents, most of these differences would have shown to be statistically significant.

9. CONCLUSION

In this work, an innovative fire spread model for a ship's engine room has been developed, utilizing the capabilities of CFD and VR technologies. The project uses the CFD software Smartfire to simulate fire behavior within the confined space of a ship's engine room for three predefined scenarios. This detailed simulation, which factors in various elements like airflow, temperature, and smoke, is then visualized through the power of VR using the Unreal Engine software, offering a more immersive and realistic depiction of the fire spread, thus confirming scientific hypothesis H0: It is possible to transfer the fire spread model obtained through computational fluid dynamics into advanced virtual reality.

Auxiliary hypothesis, AH1: It is possible to improve fire training in the ship's engine room by using a numerical model of fire spread transferred into advanced virtual reality and AH3: The adoption of fire-fighting skills will be more successful using the advanced virtual reality model compared to the basic one, are also confirmed with answers to survey question number six (I believe that the presented situation (for VR: the displayed task) could help me practice dealing with such situations in reality), where both, seamen and firefighters answered with higher probability in favor of CFD modelled scenario..

Unfortunately, auxiliary hypothesis number two, AH2: Response times of subjects in the advanced virtual reality model will significantly differ from responses in

the basic model, was not confirmed due to current bug in software which allows CFD modelled smoke to be displayed only on one eye inside VR goggles, making it impossible to run scenario properly and measure time.

Three typical scenarios, according to fire origin, have been modeled. Fire of the main engine fuel oil line, fire of the fuel oil purifier, and fire of the oily rags in a bucket. The results of those scenarios, modeled in CFD, are represented in VR.

This research contributes significantly to creating a software interface that effectively connects the CFD model with the VR representation. This interface ensures the complex data from the CFD model is accurately transferred into the virtual reality environment, making the simulation more practical for real-time interaction and analysis. The mentioned software interface is an innovative way of CFD data presentation with vast capabilities.

This advancement is particularly crucial for firefighting training exercises, where the immersive nature of VR enhances the training's realism and effectiveness, according to trainees with prior firefighting experience.

Preliminary data from user experience testing shows a good response from test subjects. It is indicated that CFD smoke results represented in VR look more realistic than the generic VR (game) smoke.

Moreover, the capabilities of the VR system allow for an in-depth examination and interpretation of the CFD model's results. This aspect facilitates a comprehensive analysis, providing insights that might be less apparent in traditional methods.

In this work, only the surface of the vast possibilities of pairing CFD and VR has been scratched, and the potential for importing/exporting results has been demonstrated. To further exploit the capabilities of both technologies, beyond advancements in terms of more powerful hardware or additional software features, it is necessary to continue research in several directions.

After fixing the bug inside the Unreal Engine with .vdb data showing only on one eye in the VR, questionnaires should be repeated, but with real VR experience for the test subjects. Test subjects should pass the familiarization phase on the VR simulator, and afterward, the fire scenarios should be initialized, after which questionnaires are to be answered.

In the next phase, temperatures will also be imported into the simulation. For the first testing, it would be only displayed on screen.

Furthermore, adding some highly specialized equipment to research is desirable. Heated vests should be acquired, meaning temperatures would become not just information on the screen but would highly affect the overall user experience.

The introduction of haptic technology into simulations is inevitable. For instance, haptic gloves that provide force or vibration on interaction with virtual objects can enhance immersion and overall VR experience [91].

The creation of some specialized equipment is a must. Creating holders for VR sensors, placing them on fire extinguishing equipment, and creating digital routines for them opens a new research bearing.

A VR walking platform is to be introduced to reduce the possibility of hitting an object and/or causing an injury.

Multi-user training is to be researched and modeled. Either to train as a team or a crew, or one of the users has the role of instructor, this kind of experience should benefit the trainees even more.

It is highly plausible that future research will lead to augmented or mixed reality firefighting scenarios, which could result in even better assimilation of technical and procedural knowledge among trainees [92].

Table of figures:

Figure 1 – Number of fires in relation to a part of a ship where it originated [4].	1
Figure 2 – Difference between CFD methods [70].	19
Figure 3 – Heat transfer differences [72].	24
Figure 4 – Six-Flux Radiation model graphic illustration [73].	25
Figure 5 – Unreal Engine Niagara properties for the fluid dynamics calculations	34
Figure 6 – Layout of the upper deck engine room.	38
Figure 7 – Layout of the lower deck engine room.	38
Figure 8 – General dimensions and equipment distances.	39
Figure 9 - How to use the Smartfire Environment [71].	42
Figure 10 – Completed geometry.	43
Figure 11 – Staircase properties.	44
Figure 12 – Case specification, general arrangement.	45
Figure 13 – Problem define section.	46
Figure 14 – Combustion model options.	48
Figure 15 – Radiation model properties.	49
Figure 16 – Inlet properties	50
Figure 17 – List of solvers.	51
Figure 18 – Solver types by variables.	52
Figure 19 – Fire of main engine – fire location.	53
Figure 20 – Fire of the main engine fuel oil pipeline - defining fire properties.	53
Figure 21 – Fire of main engine – Fire statistics.	54
Figure 22 – Fire of the purifier – fire location.	55
Figure 23 – Fire of the purifier – Fire properties.	55
Figure 24 – Fire of the purifier – Fire statistics.	56
Figure 25 – Fire of oily rags left in a bucket – fire location.	57
Figure 26 – Fire of oily rags left in a bucket – fire properties.	58
Figure 27 – Fire of oily rags left in a bucket – fire statistics.	58
Figure 28 – Geometry type.	61
Figure 29 – Mesh properties.	62
Figure 30 – Mesh for the main engine fuel oil pipeline fire scenario.	63

Figure 31 – Mesh for the purifier fire scenario.....	63
Figure 32 – Mesh for the oily rags left in a bucket scenario.	64
Figure 33 – Solving of scenarios.	66
Figure 34 – DataView software.	67
Figure 35 – Main engine fuel oil pipeline fire scenario – smoke density isosurfaces at $t = 600s$	68
Figure 36 – Main engine fuel oil pipeline fire scenario – temperature isosurfaces at $t = 600s$	68
Figure 37 – Fuel oil purifier fire scenario – smoke density isosurfaces at $t = 600s$	69
Figure 38 – Fuel oil purifier fire scenario – temperature isosurfaces at $t = 600s$	69
Figure 39 –Oily rags left in a bucket fire scenario – smoke density isosurfaces at $t = 600s$	70
Figure 40 – Oily rags left in a bucket fire scenario – temperature isosurfaces at $t = 600s$	70
Figure 41 – Steckler's experiment arrangement [82].....	72
Figure 42 – Steckler's experiment results graph [82].	73
Figure 43 – Geometry and fire and sensor locations for Steckler's simulation.	75
Figure 44 – Fire statistics for Steckler's simulation.	76
Figure 45 – Mesh for the Steckler's simulation.	77
Figure 46 - Comparison of the corner sensors for the Steckler's experiment and simulation.	78
Figure 47 – Comparison of the door sensors for the Steckler's experiment and simulation.	79
Figure 48 – Deviation of results for corner sensor.....	80
Figure 49 – Deviation of results for the door sensor.....	80
Figure 50 – Graphical voxel representation, compared to pixel [84].	86
Figure 51 – ParaView preview of smoke data from .vtu file.	94
Figure 52 – Data representation in Houdini software.	96
Figure 53 – Simplified block diagram of basic converter functionalities.	100
Figure 54 – CFD imported smoke data of fire of the main engine fuel line scenario to the VR engine room.	108

Figure 55 – CFD imported smoke data of fire of oily rags left in a bucket scenario to the VR engine room.....	109
Figure 56 – CFD imported smoke data of fire of the fuel oil purifier scenario to the VR engine room.....	109
Figure 57 – All firefighters arithmetic means results.....	114
Figure 58 - Experienced firefighters arithmetic means results.	114
Figure 59 - All seamen arithmetic means results.	116
Figure 60 – Seamen with experience, arithmetic means.....	117
Figure 61 – Seamen with only one video shown, and had fire experience...	119

Table of tables:

Table 1 – Constants for the turbulence model equations	24
Table 2 – Steckler's results for the corner sensors.	73
Table 3 - Steckler's results for the door sensors.	74
Table 4 – First set of questionnaire questions.....	111
Table 5 – Second set of questionnaire questions.	112
Table 6 - Arithmetic means and standard deviations of responses to 7 questions asked after each of 2 video clips, the first containing generic smoke and the second CFD modeled smoke, for all firefighters (N=22) and the group of experienced firefighters (N=15).....	113
Table 7 - Arithmetic means and standard deviations of responses to 7 questions asked after each of the 2 video clips, the first containing generic smoke and the second CFD modeled smoke, for all sailors (N=28) and for the group of sailors who have experienced fire onboard a ship at least once (N=23).	115
Table 8 - Arithmetic means and standard deviations of responses to 7 questions that were asked after the first video clip showing either generic (N = min 15, max 16) or CFD smoke (N = min 9, max 12) to sailors who had previous experience with fire in the interior of a ship.	118

References:

- [1] Hess M, Pavić IF, Kos S, Brčić D. Global shipbuilding activities in the modern maritime market environment. *Pomorstvo* 2020;34:270–81. <https://doi.org/10.31217/p.34.2.8>.
- [2] Stavroulakis PJ, Papadimitriou S. Total cost of ownership in shipping: a framework for sustainability. *J Shipp Trade* 2022;7:14. <https://doi.org/10.1186/s41072-022-00116-7>.
- [3] Zhang L, Wang H, Meng Q, Xie H. Ship accident consequences and contributing factors analyses using ship accident investigation reports. *Proc Inst Mech Eng Part O J Risk Reliab* 2019;233:35–47. <https://doi.org/10.1177/1748006X18768917>.
- [4] Engine room fires - causes, contributors and preventive measures 2023:1. <https://www.dnv.com/news/engine-room-fires-causes-contributors-and-preventive-measures-250621>.
- [5] Sim H, Ha W-J, Park Y-S. A basic study on standardization of fire-fighting drill scenarios on board. *J Int Marit Safety, Environ Aff Shipp* 2019;3:28–35. <https://doi.org/10.1080/25725084.2019.1698897>.
- [6] Ferrer RA, Klein WM. Risk perceptions and health behavior. *Curr Opin Psychol* 2015;5:85–9. <https://doi.org/10.1016/j.copsyc.2015.03.012>.
- [7] Smutny P. Learning with virtual reality: a market analysis of educational and training applications. *Interact Learn Environ* 2022:1–14. <https://doi.org/10.1080/10494820.2022.2028856>.
- [8] Asad MM, Naz A, Churi P, Guerrero AJM, Salameh AA. Mix Method Approach of Measuring VR as a Pedagogical Tool to Enhance Experimental Learning: Motivation from Literature Survey of Previous Study. *Educ Res Int* 2022;2022:1–9. <https://doi.org/10.1155/2022/8262304>.
- [9] Ramaseri Chandra AN, El Jamiy F, Reza H. A Systematic Survey on Cybersickness in Virtual Environments. *Computers* 2022;11:51. <https://doi.org/10.3390/computers11040051>.
- [10] Vukelić G, Vizentin G, Frančić V. Prospects for use of extended reality

- technology for ship passenger evacuation simulation. *Pomorstvo* 2021;35:49–56. <https://doi.org/10.31217/p.35.1.6>.
- [11] Tao R, Ren H, Zhou Y. A Ship Firefighting Training Simulator with Physics-Based Smoke. *J Mar Sci Eng* 2022;10:1140. <https://doi.org/10.3390/jmse10081140>.
- [12] Organization IM. International Convention on Standards of Training, Certification and Watchkeeping for Seafarers (STCW). International Maritime Organization: London, UK; 1978.
- [13] IMO. Revised Guidelines on Evacuation Analysis for New and Existing Passenger Ships. IMO: London, UK; 2016.
- [14] Puisa R, Williams S, Vassalos D. Towards an explanation of why onboard fires happen: The case of an engine room fire on the cruise ship “Le Boreal.” *Appl Ocean Res* 2019;88:223–32. <https://doi.org/10.1016/j.apor.2019.04.020>.
- [15] Chu B, Chang D. Effect of full-bore natural gas release on fire and individual risks: A case study for an LNG-Fueled ship. *J Nat Gas Sci Eng* 2017;37:234–47. <https://doi.org/10.1016/j.jngse.2016.11.043>.
- [16] Vanem E, Skjong R. Designing for safety in passenger ships utilizing advanced evacuation analyses—A risk based approach. *Saf Sci* 2006;44:111–35. <https://doi.org/10.1016/j.ssci.2005.06.007>.
- [17] Pawling R, Grandison A, Lohrmann P, Mermiris G, Pereira Dias C. Methods and Tools for Risk-Based Approach to Fire Safety in Ship Design. *Sh Technol Res* 2012;59:38–49. <https://doi.org/10.1179/str.2012.59.3.003>.
- [18] Xie Y, Liu J, Hao Z, Xu Z, Qin J, Zhu J. Numerical simulation and experimental study of gas diffusion in a ship engine room. *Ocean Eng* 2023;271:113638. <https://doi.org/10.1016/j.oceaneng.2023.113638>.
- [19] Iannaccone T, Scarponi GE, Landucci G, Cozzani V. Numerical simulation of LNG tanks exposed to fire. *Process Saf Environ Prot* 2021;149:735–49. <https://doi.org/10.1016/j.psep.2021.03.027>.
- [20] Xie Q, Guo S, Zhang Y, Wang C, Ma C, Li Q. An integrated method for assessing passenger evacuation performance in ship fires. *Ocean Eng* 2022;262:112256. <https://doi.org/10.1016/j.oceaneng.2022.112256>.

- [21] Kim I, Kim H, Chang D, Jung D-H, Sung HG, Park S-K, et al. Emergency evacuation simulation of a floating LNG bunkering terminal considering the interaction between evacuees and CFD data. *Saf Sci* 2021;140:105297. <https://doi.org/10.1016/j.ssci.2021.105297>.
- [22] Miyagawa D, Ichinose G. Cellular automaton model with turning behavior in crowd evacuation. *Phys A Stat Mech Its Appl* 2020;549:124376. <https://doi.org/10.1016/j.physa.2020.124376>.
- [23] Sumic D, Males L, Rosic M. An Agent-Based Ship Firefighting Model. *J Mar Sci Eng* 2021;9:902. <https://doi.org/10.3390/jmse9080902>.
- [24] Nagai R, Fukamachi M, Nagatani T. Experiment and simulation for counterflow of people going on all fours. *Phys A Stat Mech Its Appl* 2005;358:516–28. <https://doi.org/10.1016/j.physa.2005.04.024>.
- [25] Mei Y, Liang Y, Tu Y. A Multi-Granularity 2-Tuple QFD Method and Application to Emergency Routes Evaluation. *Symmetry (Basel)* 2018;10:484. <https://doi.org/10.3390/sym10100484>.
- [26] Ventikos NP, Sotiralis P, Annetis M, Podimatas VC, Boulougouris E, Stefanidis F, et al. The Development and Demonstration of an Enhanced Risk Model for the Evacuation Process of Large Passenger Vessels. *J Mar Sci Eng* 2023;11:84. <https://doi.org/10.3390/jmse11010084>.
- [27] Liu Y, Zhang H, Zhan Y, Deng K, Dong L. Evacuation Strategy Considering Path Capacity and Risk Level for Cruise Ship. *J Mar Sci Eng* 2022;10:398. <https://doi.org/10.3390/jmse10030398>.
- [28] Liu K, Ma Y, Chen M, Wang K, Zheng K. A survey of crowd evacuation on passenger ships: Recent advances and future challenges. *Ocean Eng* 2022;263:112403. <https://doi.org/10.1016/j.oceaneng.2022.112403>.
- [29] Arshad H, Emblemssvåg J, Li G, Ostnes R. Determinants, methods, and solutions of evacuation models for passenger ships: A systematic literature review. *Ocean Eng* 2022;263:112371. <https://doi.org/10.1016/j.oceaneng.2022.112371>.
- [30] Smith SP, Trenholme D. Rapid prototyping a virtual fire drill environment using computer game technology. *Fire Saf J* 2009;44:559–69.

<https://doi.org/10.1016/j.firesaf.2008.11.004>.

- [31] Menzemer LW, Ronchi E, Karsten MMV, Gwynne S, Frederiksen J. A scoping review and bibliometric analysis of methods for fire evacuation training in buildings. *Fire Saf J* 2023;136:103742. <https://doi.org/10.1016/j.firesaf.2023.103742>.
- [32] Zhang J, Zhu J, Dang P, Wu J, Zhou Y, Li W, et al. An improved social force model (ISFM)-based crowd evacuation simulation method in virtual reality with a subway fire as a case study. *Int J Digit Earth* 2023;16:1186–204. <https://doi.org/10.1080/17538947.2023.2197261>.
- [33] Ding N, Chen T, Zhu Y, Lu Y. State-of-the-art high-rise building emergency evacuation behavior. *Phys A Stat Mech Its Appl* 2021;561:125168. <https://doi.org/10.1016/j.physa.2020.125168>.
- [34] Jeong W, Seong J. Comparison of effects on technical variances of computational fluid dynamics (CFD) software based on finite element and finite volume methods. *Int J Mech Sci* 2014;78:19–26. <https://doi.org/10.1016/j.ijmecsci.2013.10.017>.
- [35] Shen R, Jiao Z, Parker T, Sun Y, Wang Q. Recent application of Computational Fluid Dynamics (CFD) in process safety and loss prevention: A review. *J Loss Prev Process Ind* 2020;67:104252. <https://doi.org/10.1016/j.jlp.2020.104252>.
- [36] McGrattan K, McDermott R, Floyd J, Hostikka S, Forney G, Baum H. Computational fluid dynamics modelling of fire. *Int J Comput Fluid Dyn* 2012;26:349–61. <https://doi.org/10.1080/10618562.2012.659663>.
- [37] Sharma, Pavan & Gera, B. & Singh R. A CFD Validation of Fire Dynamics Simulator for Corner Fire. *CFD Lett An Int J* 2010;2:137–48.
- [38] Mijorski S, Stankov P. CFD modeling of Dalmarnock uncontrolled fire test. *Open Eng* 2012;2. <https://doi.org/10.2478/s13531-011-0064-z>.
- [39] Yan Z. Development in comprehensive CFD simulation of fire and explosion. *J Saf Sci Resil* 2023;4:203–19. <https://doi.org/10.1016/j.jnlssr.2022.12.003>.
- [40] Ding X, Li Z. A review of the application of virtual reality technology in higher education based on Web of Science literature data as an example. *Front Educ*

- 2022;7. <https://doi.org/10.3389/feduc.2022.1048816>.
- [41] Braun P, Grafelmann M, Gill F, Stolz H, Hinckeldeyn J, Lange A-K. Virtual reality for immersive multi-user firefighter-training scenarios. *Virtual Real Intell Hardw* 2022;4:406–17. <https://doi.org/10.1016/j.vrih.2022.08.006>.
- [42] Jeon S, Paik S, Yang U, Shih PC, Han K. The More, the Better? Improving VR Firefighting Training System with Realistic Firefighter Tools as Controllers. *Sensors* 2021;21:7193. <https://doi.org/10.3390/s21217193>.
- [43] H. R. Development of Virtual Reality Training for Fire Safety Education. *Int J Adv Trends Comput Sci Eng* 2020;9:5906–12. <https://doi.org/10.30534/ijatcse/2020/253942020>.
- [44] Jiang M, Zhou G, Zhang Q. Fire-fighting Training System Based on Virtual Reality. *IOP Conf Ser Earth Environ Sci* 2018;170:042113. <https://doi.org/10.1088/1755-1315/170/4/042113>.
- [45] Yu X, Yu P, Wan C, Wang D, Shi W, Shou W, et al. Integrating Virtual Reality and Building Information Modeling for Improving Highway Tunnel Emergency Response Training. *Buildings* 2022;12:1523. <https://doi.org/10.3390/buildings12101523>.
- [46] Cha M, Han S, Lee J, Choi B. A virtual reality based fire training simulator integrated with fire dynamics data. *Fire Saf J* 2012;50:12–24. <https://doi.org/10.1016/j.firesaf.2012.01.004>.
- [47] Solmaz S, Van Gerven T. Automated integration of extract-based CFD results with AR/VR in engineering education for practitioners. *Multimed Tools Appl* 2022;81:14869–91. <https://doi.org/10.1007/s11042-021-10621-9>.
- [48] Kim R, Kim J, Lee I, Yeo U, Lee S, Decano-Valentin C. Development of three-dimensional visualisation technology of the aerodynamic environment in a greenhouse using CFD and VR technology, Part 2: Development of an educational VR simulator. *Biosyst Eng* 2021;207:12–32. <https://doi.org/10.1016/j.biosystemseng.2021.02.018>.
- [49] Kim R, Kim J, Lee I, Yeo U, Lee S, Decano-Valentin C. Development of three-dimensional visualisation technology of the aerodynamic environment in a

- greenhouse using CFD and VR technology, part 1: Development of VR a database using CFD. *Biosyst Eng* 2021;207:33–58. <https://doi.org/10.1016/j.biosystemseng.2021.02.017>.
- [50] Solmaz S, Van Gerven T. Integration of Interactive CFD Simulations with AR and VR for Educational Use in CRE, 2020, p. 2011–6. <https://doi.org/10.1016/B978-0-12-823377-1.50336-0>.
- [51] Williams-Bell FM, Kapralos B, Hogue A, Murphy BM, Weckman EJ. Using Serious Games and Virtual Simulation for Training in the Fire Service: A Review. *Fire Technol* 2015;51:553–84. <https://doi.org/10.1007/s10694-014-0398-1>.
- [52] Morélot S, Garrigou A, Dedieu J, N’Kaoua B. Virtual reality for fire safety training: Influence of immersion and sense of presence on conceptual and procedural acquisition. *Comput Educ* 2021;166:104145. <https://doi.org/10.1016/j.compedu.2021.104145>.
- [53] Feng X, Cui R, Zhao J. The Effectiveness of Virtual Reality for Studying Human Behavior in Fire, 2015, p. 13–21. https://doi.org/10.1007/978-3-319-21067-4_2.
- [54] Daylamani-Zad D, Spyridonis F, Al-Khafaaji K. A framework and serious game for decision making in stressful situations; a fire evacuation scenario. *Int J Hum Comput Stud* 2022;162:102790. <https://doi.org/10.1016/j.ijhcs.2022.102790>.
- [55] Carvalho PVR de, Ranauro DO, De Abreu Mol AC, Jatoba A, Legey de Siqueira AP. Using Serious Game in Public Schools for Training Fire Evacuation Procedures. *Int J Serious Games* 2022;9:125–39. <https://doi.org/10.17083/ijsg.v9i3.484>.
- [56] Fu Y, Li Q. A Virtual Reality–Based Serious Game for Fire Safety Behavioral Skills Training. *Int J Human–Computer Interact* 2023:1–17. <https://doi.org/10.1080/10447318.2023.2247585>.
- [57] Tao R, Ren H, Peng X. Ship Fire-Fighting Training System Based on Virtual Reality Technique, 2017, p. 249–60. https://doi.org/10.1007/978-981-10-6502-6_22.
- [58] Lawande SR, Jasmine G, Anbarasi J, Izhar LI. A Systematic Review and Analysis of Intelligence-Based Pathfinding Algorithms in the Field of Video

- Games. Appl Sci 2022;12:5499. <https://doi.org/10.3390/app12115499>.
- [59] Ren H, Tao R, Yin J. The Simulation System of Ship Fixed Water Extinguishing Training Based on Virtual Reality Technique. 2019 4th Int. Conf. Mech. Control Comput. Eng., IEEE; 2019, p. 1005–10055. <https://doi.org/10.1109/ICMCCE48743.2019.00225>.
- [60] Shaoyang Q, Hongxiang R. Ship Life-Saving Training System Based on Virtual Reality Technology. 2018 IEEE 4th Int. Conf. Control Sci. Syst. Eng., IEEE; 2018, p. 559–63. <https://doi.org/10.1109/CCSSE.2018.8724684>.
- [61] Ting Y, Feng C, Wenqiang W, Kai Y. Design and Realization of Ship Fire Simulation Training System Based on Unity3D. IOP Conf Ser Earth Environ Sci 2018;108:052101. <https://doi.org/10.1088/1755-1315/108/5/052101>.
- [62] Wu H, Yang J, Chen C, Wan Y, Zhu X. Research of Virtual Ship Fire-fighting Training System Based on Virtual Reality Technique. IOP Conf Ser Mater Sci Eng 2019;677:042100. <https://doi.org/10.1088/1757-899X/677/4/042100>.
- [63] Markopoulos E, Luimula M. Immersive Safe Oceans Technology: Developing Virtual Onboard Training Episodes for Maritime Safety. Futur Internet 2020;12:80. <https://doi.org/10.3390/fi12050080>.
- [64] Luimula M, Talvitie R, Rantalaiho-Kulo N. Virtual Training in Safety and Security – TUAS Visions for Next Generation Learning. Proc. 5th CARPE Conf. Horiz. Eur. beyond, Valencia: Universitat Politècnica València; 2019. <https://doi.org/10.4995/CARPE2019.2019.10542>.
- [65] Bellemans M, Lamrnens D, De Sloover J, De Vleeschauwer T, Schoofs E, Jordens W, et al. Training Firefighters in Virtual Reality. 2020 Int. Conf. 3D Immers., IEEE; 2020, p. 01–6. <https://doi.org/10.1109/IC3D51119.2020.9376336>.
- [66] Braun P, Grafelmann M, Gill F, Stolz H, Hinckeldeyn J, Lange A-K. Virtual reality for immersive multi-user firefighter-training scenarios. Virtual Real Intell Hardw 2022;4:406–17. <https://doi.org/10.1016/j.vrih.2022.08.006>.
- [67] Pitana T, Prastowo H, Mahdali AP. The Development of Fire Safety Appliances Inspection Training using Virtual Reality (VR) Technology. IOP Conf Ser Earth

- Environ Sci 2020;557:012064. <https://doi.org/10.1088/1755-1315/557/1/012064>.
- [68] Lomax H, Pulliam TH, Zingg DW. Fundamentals of Computational Fluid Dynamics. Berlin, Heidelberg: Springer Berlin Heidelberg; 2001. <https://doi.org/10.1007/978-3-662-04654-8>.
- [69] Lomax H, Pulliam TH, Zingg DW. Fundamentals of Computational Fluid Dynamics. Berlin, Heidelberg: Springer Berlin Heidelberg; 2001. <https://doi.org/10.1007/978-3-662-04654-8>.
- [70] Milbradt P. The most commonly used numerical methods and their essential differences n.d. https://www.researchgate.net/figure/The-most-commonly-used-numerical-methods-and-their-essential-differences_fig3_266465949.
- [71] J. EWER, F. JIA AG, PATEL, E. GALEA M. SMARTFIRE v4.3 TECHNICAL REFERENCE MANUAL. Document r. 213AD.
- [72] Ceramicx.com n.d. <https://www.ceramicx.com/information/support/why-infrared-types-of-heat-transfer/>.
- [73] Acosta-Herazo R, Monterroza-Romero J, Mueses MÁ, Machuca-Martínez F, Li Puma G. Coupling the Six Flux Absorption–Scattering Model to the Henyey–Greenstein scattering phase function: Evaluation and optimization of radiation absorption in solar heterogeneous photoreactors. Chem Eng J 2016;302:86–96. <https://doi.org/10.1016/j.cej.2016.04.127>.
- [74] Lu J. Local Existence for Boussinesq Equations with Slip Boundary Condition in a Bounded Domain. J Appl Math Phys 2017;05:1951–63. <https://doi.org/10.4236/jamp.2017.510165>.
- [75] Carmigniani J, Furht B, Anisetti M, Ceravolo P, Damiani E, Ivkovic M. Augmented reality technologies, systems and applications. Multimed Tools Appl 2011;51:341–77. <https://doi.org/10.1007/s11042-010-0660-6>.
- [76] A Taxonomy of Mixed Reality Visual Displays. IEICE Trans Inf Syst 1994;Vol.E77-D:pp.1321-1329.
- [77] Zhu Y, Li N. Virtual and augmented reality technologies for emergency management in the built environments: A state-of-the-art review. J Saf Sci Resil

- 2021;2:1–10. <https://doi.org/10.1016/j.jnlssr.2020.11.004>.
- [78] Mihai V, Rusu L, Presură A. Main requirements for ventilation of different rooms on the ships. *Analele Univ “Dunărea Jos” Din Galați Fasc XI Construcții Nav Ann “Dunărea Jos” Galati Fascicle XI Shipbuild* 2022;45:21–30. <https://doi.org/10.35219/AnnUgalShipBuilding/2022.45.03>.
- [79] Lung JCV, Sulaiman J, Sunarto A. The application of successive overrelaxation method for the solution of linearized half-sweep finite difference approximation to two-dimensional porous medium equation. *IOP Conf Ser Mater Sci Eng* 2021;1088:012002. <https://doi.org/10.1088/1757-899X/1088/1/012002>.
- [80] Adsua JE, Cordero-Carrión I, Cerdá-Durán P, Aloy MA. Scheduled Relaxation Jacobi method: Improvements and applications. *J Comput Phys* 2016;321:369–413. <https://doi.org/10.1016/j.jcp.2016.05.053>.
- [81] Preventing an engine room fire n.d.
- [82] Steckler KD, Quintiere JG, Rinkinen WJ. Flow induced by fire in a compartment. *Symp Combust* 1982;19:913–20. [https://doi.org/10.1016/S0082-0784\(82\)80267-1](https://doi.org/10.1016/S0082-0784(82)80267-1).
- [83] Lee T-Y, Weng T-L, Lin C-H, Sun Y-N. Interactive voxel surface rendering in medical applications. *Comput Med Imaging Graph* 1999;23:193–200. [https://doi.org/10.1016/S0895-6111\(99\)00015-4](https://doi.org/10.1016/S0895-6111(99)00015-4).
- [84] Tino R, Yeo A, Brandt M, Leary M, Kron T. The interlace deposition method of bone equivalent material extrusion 3D printing for imaging in radiotherapy. *Mater Des* 2021;199:109439. <https://doi.org/10.1016/j.matdes.2020.109439>.
- [85] Shen Y, Moore RH, Deo A. Visualizing Abaqus output database in ParaView: A universal converter in Python and C++. *SoftwareX* 2023;22:101331. <https://doi.org/10.1016/j.softx.2023.101331>.
- [86] ParaView Data (pvd) file format and writing n.d. <https://getwelsim.medium.com/paraview-data-pvd-file-format-and-writing-946b4c65da4a>.
- [87] The VTK user’s guide. 11th ed. Kitware, Inc.; 2010.

- [88] Erolin J. What Is Unreal Engine? n.d.
- [89] Introduction to Houdini. SideFX; n.d.
- [90] Glasow PA. Fundamentals of Survey Research Methodology. MITRE; 2005.
- [91] WANG D, GUO Y, LIU S, ZHANG Y, XU W, XIAO J. Haptic display for virtual reality: progress and challenges. Virtual Real Intell Hardw 2019;1:136–62. <https://doi.org/10.3724/SP.J.2096-5796.2019.0008>.
- [92] Hoffmann, J.; Polikarpus S. Application of Augmented Reality in firefighters training: From Safe to SafAR. GI VR / AR Work., 2022.