Post-print version of Solmaz, Serkan, and Tom Van Gerven. "Automated integration of extract-based CFD results with AR/VR in engineering education for practitioners." Multimedia Tools and Applications (2021): 1-23. DOI: https://doi.org/10.1007/s11042-021-10621-9

Automated integration of extract-based CFD results with AR/VR in engineering education for practitioners

Serkan Solmaz (ORCID ID: 0000-0001-5431-4038) and Tom Van Gerven* (ORCID ID: 0000-0003-2051-5696)

* Corresponding author at tom.vangerven@kuleuven.be

Department of Chemical Engineering, KU Leuven, Celestijnenlaan 200F, B-3001 Leuven, Belgium

Abstract

Computational fluid dynamics (CFD) simulations can provide meaningful technical content in engineering education, broad engineering and business. However, computationally demanding data production and complex data processing environments of CFD simulations turn them into esoteric tools for potential non-expert users. This consequently limits applications and communications of CFD simulations and results. Augmented and virtual reality (AR/VR) technologies are opening new gates for visualization and interaction techniques. Despite the many recent attempts, the literature lacks an inclusive system development procedure for CFD simulations with AR/VR. The present study proposes a component-oriented system architecture to generate dedicated workflows for any kind of AR/VR environment supported by CFD simulations. The study further explores the potential of data processing options throughout the preparation of the simulation dataset with AR/VR. An automated data coupling strategy is additionally introduced to ease multiplatform integration. We provide an integration strategy with simple, easy-to-implement, end-to-end, automated and free-to-use utilities that the practitioners can readily pursue.

Keywords: Computational fluid dynamics, Virtual reality, Augmented reality, Engineering education, System architecture, Scientific visualization

1. Introduction

Computational Fluid Dynamics (CFD) simulation tools are powerful alternatives to experimental campaigns because of their cost- and time-effectivity. Visualization, virtual experiments, and prototyping are profound features of CFD simulations. Preparation, calculation, and post-processing are a set of subsequent steps that are executed manually by a competent analyst. CFD simulation results are generally analyzed in a heavily interactive process with specialized post-processing software to obtain meaningful data in support of modeling and design. However, meaningful data still can be hard-to-comprehend for inexperienced users (e.g. engineering students) due to the sophisticated structure of the conventional post-processing environments [1].

Decision-making in engineering is getting challenging because of the increasing complexity of technical subjects. Augmented and virtual reality (AR/VR) can help communities to deliver more visual and interactive digital products through a better understanding of physical phenomena with meaningful, realistic, easy-to-comprehend, functional and attractive digital environments [2–4]. This can reduce the communication gap between CFD simulations and broader engineering including students, clients, customers and stakeholders. CFD simulations may even be utilized to operate active learning

environments in engineering education (e.g. inquiry-based learning, game-based learning and learning-by-doing) due to its (edit: their) computational nature. Nonetheless, the lack of desired digital content and content integration methodologies are the missing building blocks in the development of digital resources (edit: applications) [1, 3, 5, 6].

The integration of CFD results with AR/VR recently received scholarly attention. Literature has still been missing an inclusive system architecture to guide the practitioners [3]. The term system architecture is used as a portrayal of the system that consists of software and hardware including their functionalities and interactions [7]. Besides, platform-specific, computationally-demanding, non-replicable and manual integration methodologies have mostly been reported by researchers [1, 3]. A conclusion has still not been made on how to post-process big CFD data into manageable parts, which an AR/VR development platform and user's device can handle. Additionally, no study has been reported on the multiplatform integrity to assess the data processing performance through data format, size, processing time, quality, automation and management. These ambiguities prevent practitioners to utilize AR/VR tools to visualize simulation data [1, 3].

1.1. Contributions

The objective of the present research is to provide a technical development strategy in support of practitioners to use CFD simulations with AR/VR in engineering education. Our research discloses significant contributions:

- **A system architecture to develop educational tools with CFD simulations and AR/VR.** It proposes an inclusive guideline on the multiplatform integration targeting miscellaneous workflows and excellence of each utility with component-oriented structure.
- **A robust approach to adapt CFD data with AR/VR.** We provide an extract-based data processing approach to integrate any type of CFD visual representations. Methodologies provide alternative features upon simple, easy-to-implement, end-to-end and free-to-use utilities.
- **A qualitative assessment on multiplatform integration.** Multiplatform integrity is examined based on data format, size, processing time and quality. This is significant to elaborate extract-based data processing methodologies and determine suitable utilities.
- **An automated one-way data processing pipeline.** A Python script enables a soft-coded, modular and automated end-to-end integration of CFD datasets in the game engine.
- **A descriptive guidance on the management of data in use.** We provided a comparison among data models to store and retrieve CFD datasets in manageable parts throughout the integration.

The findings of the present study will help practitioners to integrate desired simulation results with the aid of newly adapted technologies in engineering education.

2. Related work

2.1. Expectations in CFD simulations: experts vs. non-experts

A CFD simulation can be used in many contexts such as engineering design and optimization projects, training and educational materials, and video games. It is conjectured on a cost function that is composed of certain parameters to tune from geometry to representation of results. Accuracy and frame rate (or run-time) are the most significant outputs revealed from the cost function. The accuracy points out the

difference between the obtained and standard values. The frame rate refers to the total number of datasets produced per second. Depending on the context of the simulation, the accuracy and frame rate are modified to develop the desired simulation content. While a video game requires visually credible assets with relatively high frame rates, engineering simulations rely on physics-based models with high numerical accuracies.

Literature claims a continuum between accuracy and frame rate for CFD simulations [8], without, however, quantifying thresholds for these two criteria. Fig. 1 demonstrates a widened continuum to detail the current state of CFD simulations. Firstly, physics-based and fluid-like modeling approaches are the two distinctive methodologies to set a clear boundary between the levels of accuracy. Secondly, the frame rate of a simulation can be defined either batch or real-time. Compared to batch, real-time simulations should solve and process at least 24 frames per second (fps) [9]. The potential of real-time CFD simulations has been investigated targeting different contexts such as games [10, 11], fire simulation [12], chemical site evacuation scenarios [13] and fluid flow modeling [14]. Although, real-time simulations in design and modeling are still in their infancy, thus mostly impractical to use in multiphysics CFD modeling. Unless results are retrieved from a database, a time gap should inherently be taken into account by practitioners from simulations to the visualization of results.

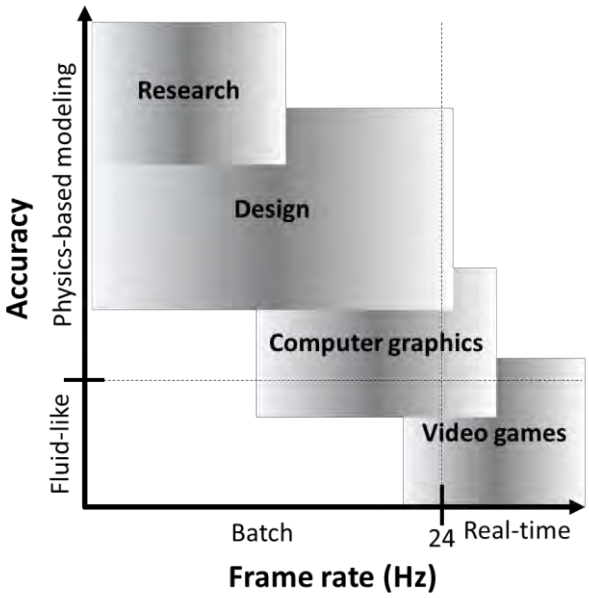


Fig. 1 The continuum of CFD simulations concerning accuracy, frame rate and context

2.2. Integration of CFD simulation data with AR/VR

The extraction of the simulation dataset is an important stage in CFD simulations, according to the NASA CFD Vision 2030. Visualization, management and integration of simulation data should be better developed in simulation software to enable engineers to carry out a complete simulation project within a restricted time and make decisions. Automation of the procedure, database management, validation, visual representation, real-time processing, multiphysics simulations, multiplatform processing, novel

visualization methods of simulations (AR/VR) are identified as the key areas for further improvement in this realm [15].

In the literature, various approaches have been proposed on the system development to integrate CFD simulation with AR/VR. Likewise, a variety of data processing methodologies have been suggested to structure digital resources. The following sections review available development approaches and data processing methodologies.

2.2.1. System development: hardware and computational challenges

He et al. proposed an integration methodology on CFD simulations with AR/VR via cloud computing [16]. The main disadvantage of the methodology is that only support of cloud computing is considered whereas other network possibilities were dismissed. Similarly, Kim et al. proposed a framework built upon the CFD simulation, AR post-processing, and AR visualization together with devices and cloud servers [17]. Although both studies explain well the system components, the procedures and requirements to develop the system were excluded. Another shortcoming is that the focus on specific AR/VR features provides a restricted system architecture to replicate the methodologies.

Cha et al. designed a system architecture for fire training simulators including four main components: training and evaluation logic, hardware and software resources, human-computer interaction, and end-user simulator [18]. A system development procedure was promoted with user-specific components and dataflow. Another system architecture was detailed by Yu et al. including dataflow and system requirements but thoroughly structured with dedicated system components too [19]. Software, hardware, and network options were defined as fundamental system elements to develop digital environments.

System development procedures for real-time simulations were recently reported as well. Harwood and Revell compared dataflow methodologies of interactive simulation to those of traditional simulation [20]. Even though procedures to run every single component can vary, both simulation techniques are designed on similar system elements.

Researchers showed interest in the Internet of Things (IoT) to interact with the physical environment. An approach was demonstrated to process energy performance data with augmented reality in buildings [21]. Another study evaluated the design phases of a system to use sensor data with AR/VR [22]. The studies revealed that IoT devices can leverage the interaction with simulation results through the real-time measured data from the physical environment.

Only a few studies have shown direct interest in database and management systems of simulation datasets in system architecture [23, 24]. The CFD Vision 2030 report from NASA states that real-time querying of large databases and continuous update of these will be quintessential in the near future [25]. A database management system is a must to save, store, reproduce, and reprocess simulation datasets.

To sum up, system development procedures were principally structured on software, hardware, dataflow, network, IoT, and database to design digital environments. Researchers' interest in task dedicated subjects to integrate simulation datasets resulted in very specific system development procedures. A generic system architecture to guide the practitioners is not available yet.

2.2.2. Data processing: software and data handling challenges

A CFD dataset is usually stored in a data exchange format pertinent to the CFD solver. It can be either a dedicated or text file format that is commonly used for the storage of information. Both can only be processed with CFD post-processors, not with any other 3D computer graphics software without required post-processing algorithms. Therefore, a data processing methodology is required to adapt CFD data with AR/VR development platforms such as game engines. In contrast, supporting multimedia (image and video) and analytic data (direct or derived results in text format) exported from the post-processor can be directly imported into game engines without any further manipulation. Both CFD post-processors and game engines give support for the most common multimedia types and data formats.

Visualization of CFD data can be categorized as vector-, surface-, and volume-based. In recent years several 3D data formats have been released to encode digital visual objects. A 3D data format mainly consists of information on geometry, texture, scene and animation. Researchers have investigated the applicability of different 3D data formats to integrate the CFD simulation dataset with AR/VR. STL [26] [27], 3DS [28], VRML [29–32], FBX [31–33], DAE [34], and X3D [26] were commonly applied with several CFD solvers, post-processors, and 3D computer graphics software to achieve the integration. It is observed that complex structure of CFD dataset and arbitrary nature of data processing make a comparison among studies almost impractical.

The processing of CFD dataset with Visualization Toolkit (VTK) is popular among researchers. Huang et al. developed an interactive AR-based finite element method (FEM) simulation environment with VTK [35]. In another study, Kim et al. processed a CFD dataset with ParaView and VTK to develop an AR application [17]. Wheeler et al. structured a plugin to export VTK data format to Unity [36]. These studies fundamentally reported user-defined data processing methodologies to import VTK files into Unity. Lin et al. additionally developed a data format to reduce the data size and loading time of CFD dataset against VTK format in a mobile device-based AR [37]. The main weakness in many of these studies is that they overlook the processing of datasets produced by any other CFD solver that cannot be turned into VTK format without a wrapper [38]. Such a processing method restricts the applicability of other CFD software. In addition, game engines do not support VTK format so that practitioners may rely on unofficial data processing pipelines. This can adversely affect data size, processing time and, more importantly, quality and replicability.

Visualization of simulation datasets with AR/VR embedded post-processors [39–41] and web-based post-processing environments [42, 43] were also found nascent applications. Yet, both approaches are designed on intricate post-processing environments of engineering software which restricts data exploration and user interaction significantly [1].

Contrary to 3D data formats, test-based data extractions (such as CSV and TXT) can be utilized to integrate CFD dataset with AR/VR. Berger et al. were among the first ones who developed a plausible workflow for a simulation dataset to create 3D visualization directly in the game engine [44]. Yao et al. [45] and Kim et al. [46] also reported similar user-defined and manual processing methodologies. These studies suffered from many pitfalls. Initially, the relevant CFD data post-processing algorithms should be reinterpreted and recompiled in the game engine to reproduce CFD visualizations from a text file. While this method may be computationally demanding for large datasets, it would also require compatible hardware to execute the data processing. This results in an enormous amount of redundant work that can readily be handled

by CFD post-processor and 3D data extracts. The methodology is also generally applicable for streamlines and similar structures, not any other visual representations to explore multiphysics CFD simulations (contours, iso-surfaces, and iso-volumes). Besides, switching to another game engine can undermine the effort put in the former platform because of differences in application programming interfaces (API). Nevertheless, reprocessing the CFD dataset in a game engine would be beneficial if an interaction is required in the end-user console to manipulate the CFD results for some reason.

Representation of analytic data can be handled in several ways. Graphs are often reproduced with CFD post-processors to interpret results. Exporting a graph in image data format enables the direct import of digital assets to a game engine without any further manipulation of the dataset. A CFD dataset can alternatively be reprocessed with 3D computer graphics software to generate interactive 3D graphs in AR/VR. Using game engines for scientific computing and data analytics extends their applicability to represent analytic results [47]. Several open-source plugins have been reported in the literature enabling direct reprocessing of TXT and CSV for visual interactive analytics [48, 49].

3. Methodology

In an attempt to develop generic system architecture, we propose an extended methodology on the integration of CFD simulations with AR/VR. Not only CFD simulation but also a well-structured dataset from an experimental study can be implemented in the present methodology. The prior aim of the system architecture is to present a scaffold for a multiplatform-based generic system development procedure targeting miscellaneous workflows. This allows practitioners to integrate simulations with AR/VR and orchestrate a system within its components.

The system essentially deals with multiple platforms through unique operations. We pursue a component-oriented development strategy to handle the diversity of software in the integration of CFD with AR/VR. A sustainable integration can solely be achieved considering the excellence of each component. A similar approach for a software development procedure was detailed to avoid monolithic entities as well as to reduce the development cost by utilizing components already available [50].

3.1. Generic system architecture

A system is composed of components, requirements, hierarchies, and other fundamental elements depending on the discipline it is used for [51]. In the present study, a system architecture directly targets a canvas that is made of software, hardware, functionalities, and connections between each system element.

An architecture is proposed in Fig. 2. At first, “components” are defined to express main tasks to achieve multiplatform integrity. Each component encompasses “sub-components” in which a part of a task can be executed subsequently. Secondly, “system requirements” are proposed to convene specifications for each component and sub-component where required. Operating system, software, hardware, add-in, network, programming language (application programming interface, compiler or interpreter), and IoT are classified under the system requirements. They are determined based on their added values as reviewed in the literature. System requirements offer a toolbox of functionalities to execute dedicated tasks in the development. The final element of the system architecture is the “dataflow” that indicates the procedural interaction between system components.

User experience may indirectly contribute to the system architecture. Molonet et al. illustrated a system to evaluate user's interaction with simulations inside a game engine together with learning analytics [52]. The primary focus was on how the end-user environment affects the system architecture and development. Besides, dataflow additionally comprises possible design strategies that can be applied upon pedagogical concerns [53]. Hamilton et al. reviewed an approach to design learning environments with relevant academic content and technological support [54]. The study claimed the significance of users' interferences in the development and design of learning environments. These aspects are included in the system architecture within the form of "indirect components".

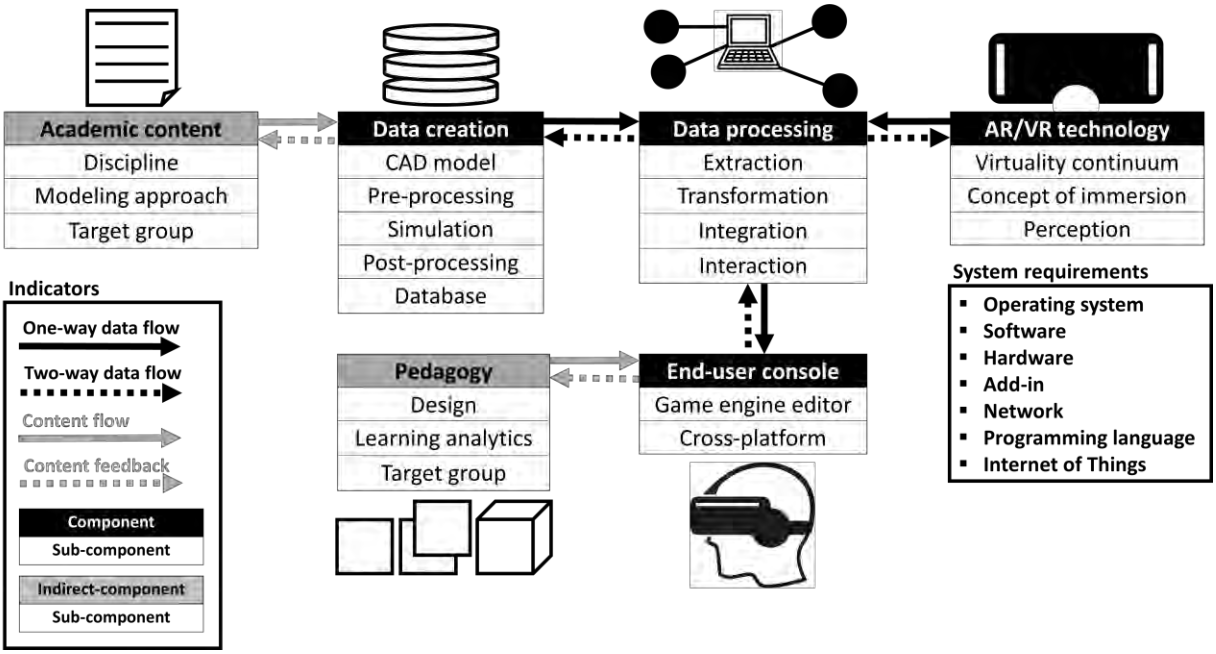


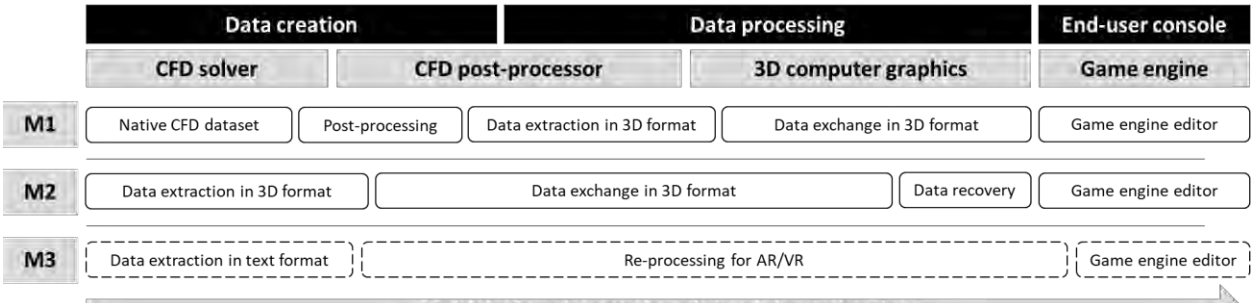
Fig. 2 Proposed system architecture to integrate CFD simulations with AR/VR

3.2. Visual representations of simulation datasets

A CFD solver produces simulation results in a native data format which points out specifically encoded analytic results. These results generally consist of large, complex, and transient datasets of solutions of partial differential equations. In that sense creating data extracts from native CFD data can contract data size drastically [23]. Results in native format can also be processed and exchanged to interoperable CFD data models during the extraction for multiplatform usage. Moreover, processing of a simulation dataset to generate meaningful visual representation requires technical expertise. Thus an analyst should optimize the CFD post-processing workflow in advance for inexperienced users.

Visual representation of CFD results can be compartmentalized as vector-, surface-, and volume-based. Vector-based visualization consists of a vector field within point and gyroids. It provides a very restricted visualization to interpret simulation results. Surface-based visualization enables layers with dedicated textures to visualize results. Likewise, volume-based representation, for instance, streamlines, is the most visual type of data representation due to the inclusion of 3D geometry and texture together. To export the visual representation of a CFD dataset, a post-processor or 3D computer graphics software should be employed to produce extracts in 3D data formats. Fig. 3 presents extract-based data processing

methodologies to integrate CFD data with AR/VR. The integration may comprise several intermediate data formats and software due to the interferences of multiple platforms. Methodologies M1 and M2 were thoroughly assessed in the present study due to the advantages brought in the data processing. M3 was neglected in the present study due to platform- and simulation-specific, computationally demanding, and time-consuming features, as reviewed in Section 2.



Methodology	User defined input and coding	Metadata production	Platform dependency	Multiphysics post-processing options
M1	Very low	Yes	Low	Versatile
M2	Very low	Yes	Low	Versatile
M3	Very high	No	Highly dependent	Very limited

Fig. 3 Methodologies on the integration of extract-based CFD data with AR/VR cross-development platforms. Components implemented in the methodologies are shown with solid outlines. Components with dashed outlines (only appeared in M3) means that no implementation is taken place in the present work

3D data formats, to generate 3D digital visual contents, comprise a geometry and can also be composed of texture, scene, and animation. Hence, visual representations of CFD datasets are likely to any 3D data exchange formats used in 3D computer graphics software as proposed with methodology M1 and M2. More details on 3D data formats (developer, license, data type, and data encoding characteristics) can be found in the Supplementary Material 1.

3.3. Database and management system

A CFD solver operates large and complex simulation datasets. A database and data management system might be required to access (save, store, search, trace, and compartmentalize) and process (import, export, reproduce, and reprocess) data, and prevent any data loss or falsified action. Database management systems (DBMS) are platforms to organize data and databases to avoid the drawback of a traditional file-based system. A DBMS prevents data redundancy and inconsistency, difficulty in accessing data, data isolation, insecurity, transaction problems, non-traceability, and several other downsides of file-based systems [55]. Commercial and open-source CFD software provide well-designed DBMSs. No further action is needed to manage simulation datasets except for the necessary hardware as physical storage. Nevertheless, the integration of CFD with AR/VR unleashes a multiplatform data processing procedure. The data can be encoded with a 3D data exchange format that is unfamiliar to the CFD simulation software. Thus, an additional database system can be implemented to manage intermittent and unfamiliar metadata formats. Beyond, a database system for CFD with AR/VR may enable the

practitioners to integrate many outsource data from CFD databases developed by scientific communities, application galleries of CFD software, and even academic publications.

4. System implementation

A case study was performed to assess the system architecture. We widened the case study to investigate the current data processing methodologies through data formats and software. Beforehand, exploratory work was carried out to review applicable data formats (Supplementary Material 1). As a benchmark, a CFD study for the 3D transient mixing process in a stirred tank was performed to sample CFD data. It was processed to develop a mobile marker-based AR application. A workflow was eventually consolidated to address the system requirements.

4.1. CFD simulation

The 3D CAD model was designed with FreeCAD v0.18.4. Grid generation, pre-processing and numerical solution of the mixing process in a stirred tank was executed with SnappyHexMesh grid generator and pimpledymfoam solver in OpenFOAM v3.0.1. Post-processing of the simulation dataset was carried out with ParaView v5.8.0. A Dell Precision 5530 Notebook (processor Intel(R) Core (TM) i7-8850H CPU @ 2.60GHz, 2592 MHz, 6 Cores, 12 Logical processors) with both Linux Ubuntu 16.04 and Microsoft Windows 10 as operating systems was utilized to run all relevant actions for the integration process. The simulation dataset was saved in the internal hard disk after the calculation was completed.

4.2. AR/VR technology

Developing an AR/VR application requires 3D computer graphics software to compile and build an executable final user application. Nowadays game engines support AR/VR projects via software development kits (SDK) as cross-platform developers. Unity and Unreal game engines have wide compatibility with AR/VR SDKs (details in Supplementary Material 1). It was decided that the best procedure for this integration study was to use a game engine with related AR/VR SDK. The AR application was developed with Vuforia SDK that was operated in the Unity 2019.1.0f2 game engine. A marked-based detection approach was selected to interact with the AR application with a real environment as being an easy-to-adapt feature. Vuforia SDK requires an online asset generation of marker for Unity. Although the present study was aiming to assess the integrity of the engineering simulation dataset, some of the interactive digital elements were also adapted to experience features of AR. Virtual buttons in Vuforia SDK were customized for use with the AR application to increase the users' interaction and immersion. Customization of the final-user console with either an AR or VR SDK does not affect the data processing procedure. A mobile application was generated with Unity and Vuforia SDK to visualize the simulation results into mobile marker-based AR. A Samsung A20e with Android Pie 9 was employed to test the final application.

4.3. Data processing

CFD post-processors are typically incompatible tools to process the CFD dataset throughout multiple platforms. Notably, these tools target visual representation of the dataset in a single or embedded platform upon engineering demands. Lack of 3D data export and connectivity options with external platforms is the main shortfall of CFD post-processors. The data format is one of the core elements of the system architecture along with the complete development procedure to store, extract, transfer, integrate

and interact simulations with AR/VR. In order to identify the selection of an appropriate data format, a case study was performed. Blender v7.9. was implemented as a data processor to exchange data among 3D formats.

4.3.1. Assessment on simulation dataset

An exploratory research was performed (Supplementary Material 1) to identify data import and export capabilities of software in use throughout multiplatform integration. Seven different cases were identified to assess the data import and export capabilities of software as shown in Table 1. In the scope of the present study, GLTF and WEBGL/VTKJS, newly developed mobile-friendly 3D formats for AR and web-based platforms [56, 57], were additionally taken into consideration.

Table 1 Case studies designed from data processing options supported through multiplatform integration

Software	OpenFOAM	ParaView		Blender		Unity
Process	**Export**	**Import**	**Export**	**Import**	**Export**	**Import**
Case 1	FOAM	FOAM	X3D	X3D	FBX	FBX
Case 2	FOAM	FOAM	X3D	X3D	GLTF2	-
Case 3	FOAM	FOAM	X3D	X3D	OBJ	OBJ
Case 4	FOAM	FOAM	X3D	X3D	DAE	DAE
Case 5	FOAM	FOAM	GLTF1	-	-	-
Case 6	FOAM	FOAM	X3D	X3D	STL	-
Case 7	FOAM	FOAM	VTKJS	-	-	-

Five different visual representations from the CFD dataset were sampled for the assessment (Fig. 4). Vector-based visualization was purposefully eliminated due to the same 3D data structure with volume-based visualization of streamlines. Initial cases were performed for each processing option to examine the intervention of visual representation techniques. Data size and processing times were compared for each case separately. Once the data processing procedure was finalized, the quality of data processing was evaluated to detect whether any data was lost. Repetitions were prompted under conditions at which only the required software was run. Finally, additional tests and comparisons were performed to anticipate enhancements that could be entailed in the future with a focus on data processing.

Fig. 4 Visual representations of CFD dataset assessed in the case study

4.3.2. Development of the end-user console

Integration of the CFD dataset with AR/VR was performed within the digital environment as illustrated in Fig. 5. End-user console was solely designed for the visualization of CFD dataset with mobile marker-based AR application.

Table *2* presents the dataflow diagram of the application built upon the system architecture as a transcript of the integration. The full transcript including system requirements can be found at Supplementary Material 2.

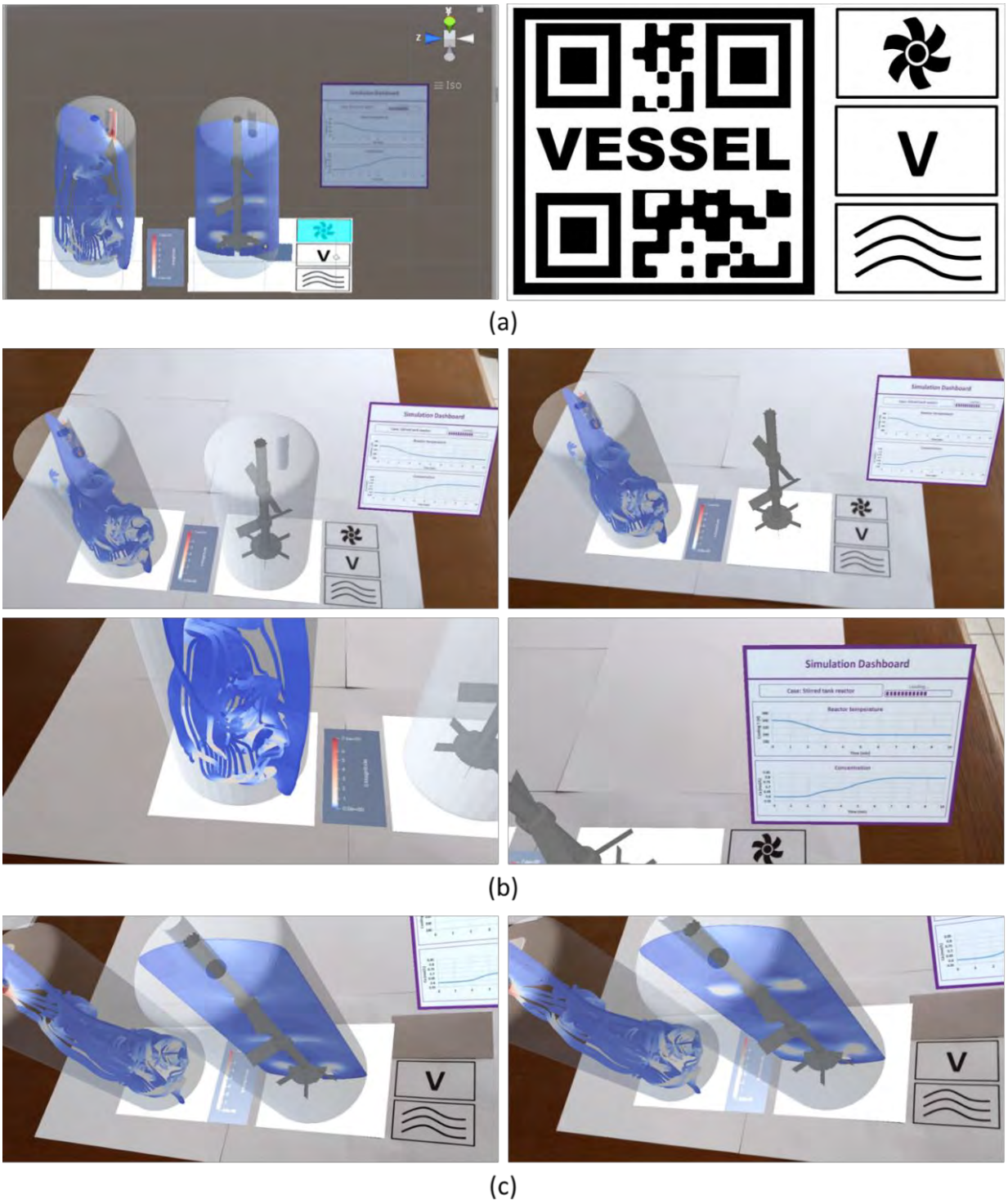


Fig. 5 End-user console developed for mobile marker-based AR application; (a) virtual design environment in the game engine and AR marker; (b) application; (c) initialization of transient visualization of 2D velocity contour with virtual button

Table 2 Dataflow diagram of the mobile AR application generated from the system architecture

Component	Sub-component	Data analytics				Data coupling
		Dataflow	Format	Size (MB)	Processing time (s)	One-way
Data creation	CAD	1	STL	2	-	-
	Pre-processing	2	FOAM	122	-	-
	Simulation	3	FOAM	4807	-	-
	Post-processing	4	FOAM	4807	-	
	Database	5	FOAM	4807	-	
Data processing	Extraction	6	X3D	85	4.8	Python in Anaconda Navigator with required packages
	Transformation	7	FBX	12	30.72	
AR/VR technology	Virtuality continuum	8	FBX	12	-	
	Concept of immersion	9	-	-	-	
	Perception	10	-	-	-	
Data processing	Integration	11	FBX	12	6.5	
	Interaction	12	-	-	-	-
End-user console	Built-in	13	FBX	12	65	-
	Simulator	14	APK	58	112	-

5. Results and discussion

5.1. Evaluation of extract-based data processing with suitable data formats: Case1, Case 3 and Case 4

Complete integration of the CFD dataset with the game engine was merely achieved with Case 1, Case 3 and Case 4. Even though the STL data exchange format (Case 6) could not be integrated into the game engine, it was also processed to assess its potential due to widespread fame in computer-aided design (CAD). It should be noted that proprietary data formats do not reveal data encoding techniques. Hence it is hard to predict the behavior under varying circumstances. Hence, a comparison among relevant formats was imperative to unveil data processing characteristics. Fig. 6 shows data processing analytics of the cases based on data size compression and data processing time. Each data presented in the analysis is the average of 10 repetitions with a 1.5% maximum error rate for the processing time. No change was observed in the data size. All other external and irrelevant activities were shut down and tracked with the total central processing unit (CPU) utilization in all cores, while the repetitions were committed.

The comparison between the case studies highlighted that FBX, OBJ and STL reduced data size significantly whereas DAE unexpectedly increased it. Large volumes of dataset is a major trait of CFD simulation datasets. Allocating lower data size in computer memory is, therefore, a desired trend. As a higher-size data format would comprise more information related to the 3D assets, it could also be designed without concerns over an inefficient data encoding technique. The data processing time and the quality of data should be considered concurrently so as to make comparison reliable. A striking observation emerging from the data format comparison was the processing time. Interestingly, DAE processed the simulation dataset in a far shorter processing time: at least 1.5 times faster than STL, 3 times faster than FBX, and 7 times faster than OBJ. Though data size was increased with DAE, it required less processing time.

The study did not confirm any significant variation among the visual representation techniques of the CFD datasets shown in Fig. 4. All data exchange formats revealed a particular but similar data processing analytics for visual representation techniques. The larger the data size was, the longer the processing time became, as illustrated in Fig. 6.

The complete integration of the CFD dataset in the game engine included four major processing steps: export from ParaView, import to Blender, export from Blender, and import to Unity. Further analysis of the data processing time underlined that import and export characteristics of software differ among data formats. In spite of a faster export from DAE with Blender, the import from DAE to Unity was slower than FBX and OBJ. As shown in Fig. 6b, processing times were compared among data formats through data processing. CFD visual representation techniques again did not reveal any noticeable difference. In all cases, OBJ required a longer data processing time compared to FBX and DAE. There was no significant deviation between FBX and DAE for CFD visual representations #1, #2 and #3. However, streamline-based CFD visual representations #4 and #5 took a longer time with FBX. This stresses the importance of an inclusive assessment concerning data format, data size, processing time, and software.

It is noteworthy that the data size and processing time are quantitative aspects of the integration. However, they are not related to the data quality. A further inspection should, therefore, be performed to assess data loss against relevant CFD post-processing (Fig. 4). To do that, the processed 3D data file was compared to the initial 3D data imported from the CFD post-processor. The study showed that both FBX and DAE import all relevant visual assets related to geometry, appearance, and the environment in a single file without any change. On the other hand, OBJ encoded the geometry and appearance in different files, and entirely excluded scene and animation. Despite the fact that the geometry was encoded without any data loss, encoding appearance in an external file caused a troublesome data import to game engine in which appearance was eventually lost. This can prevent practitioners to utilize the OBJ when appearance (color, legend, contour, etc.) is an essential part of processing. Caution must be exercised in the use of OBJ accordingly. Moreover, both FBX and DAE kept and encoded animation created with intermediate software while OBJ did not. Creating 3D animation with OBJ should be done in the final environment, for instance, the game engine itself. FBX and DAE further brought assets with layers at which enabling and disabling of particular visual contents could be performed. All three formats were open for scalability in the spatial domain.

Another comparison was performed for post-adjustment made in Unity and response time related to the data format used. Change of texture, scalability, and execution of manipulations were investigated. It was indicated that post-adjustments in Unity with FBX was 1.4 times and 5 times faster compared to DAE and OBJ, respectively.

The results point to the likelihood that each data format can enable unique characteristics to prompt data processing methodology upon data size, processing time, quality and post-treatment. Practitioners should decide on suitable data formats upon constraints of the multiplatform integration.

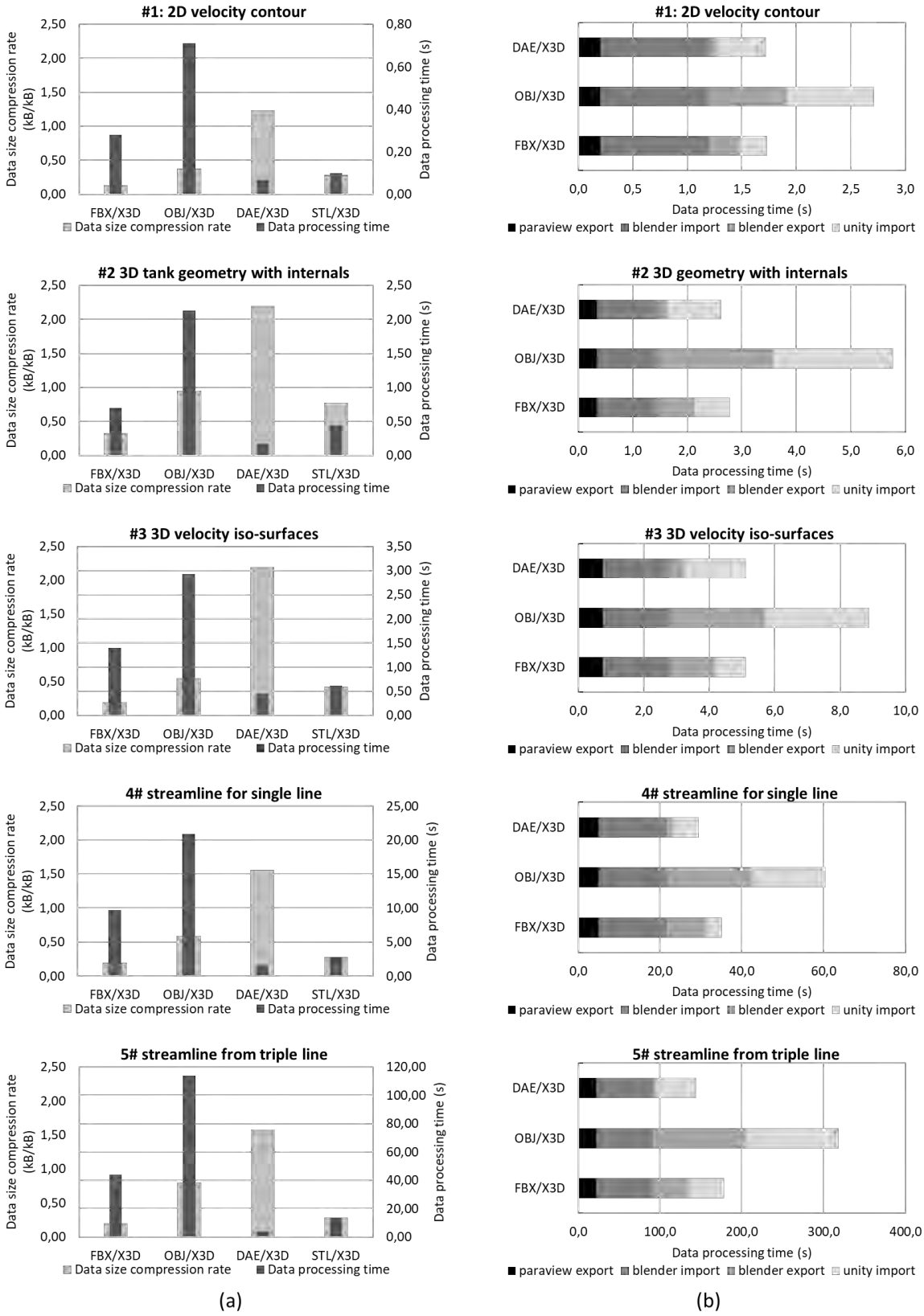

#1: 2D velocity contour
Data size compression rate (kB/kB)
Data processing time (s)
FBX/X3D
OBJ/X3D
DAE/X3D
STL/X3D
Data size compression rate
Data processing time
#2 3D tank geometry with internals
#3 3D velocity iso-surfaces
4# streamline for single line
5# streamline from triple line
paraview export
blender import
blender export
unity import
(a)
(b)

Fig. 6 Data analytics of processing options; (a) data size compression; (b) data processing time

5.2. Failed data processing options: Case 2, Case 5, Case 6 and Case 7

In this section, an investigation was elaborated on data formats for simulation datasets that would be potentially integrated into AR/VR in the future. STL has evolved to world-wide utilization due to the support of CAD software. Fig. 6 shows that data size and processing time were reduced significantly with the utility of STL against OBJ. The main downside of STL is no support of game engines, which makes its applicability impractical.

Comparison between software throughout the integration based on processing capabilities assured that X3D is the most convenient data exchange format to start with as it was applied in the present study. Moreover, GTLF1 and VTKJS data formats were also taken into account in separate data pipelines to assess data encoding performances. It was recently claimed that both formats could simplify data encoding procedure and shorten software pipeline [58, 59]. Another study stated that GLTF reduces complexity of 3D data and users' interaction, but also quality [58]. These formats currently cannot be a part of data processing options due to the lack of technical support from game engines. Nonetheless, concerning an update of the support in the future, we performed a comparison between the 3D data exchange formats X3D, GLTF1 and VTKJS as illustrated in Fig. 7. Both data size and processing time were clearly reduced with GLTF1 and VTKJS. Even though both data formats concluded promising results, the currently released versions of game engines do not give any official support. Hence, the quality of the data processing remains unanswered.

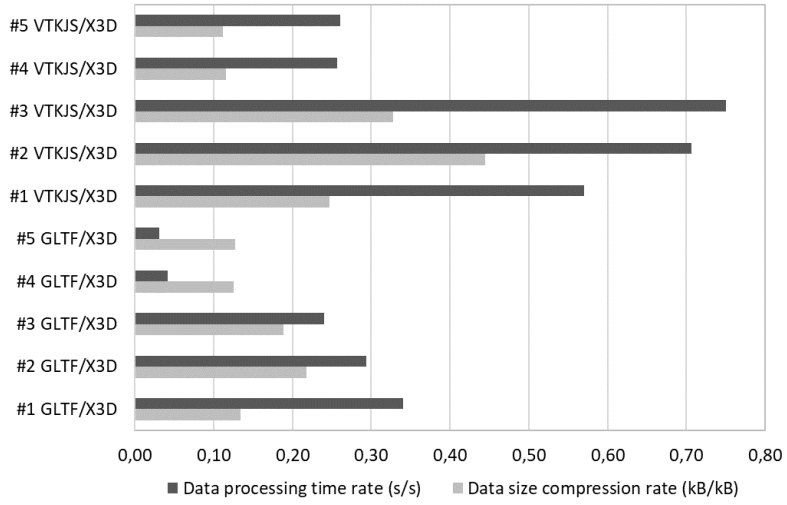


Fig. 7 Data export comparison among X3D, GLTF1, and VTKJS

The CFD dataset was processed to the GLTF2 data format with the use of Blender as proposed with Case 3. It surprisingly resulted in far adverse analytics: 5 times higher data size than FBX and 47 times higher data processing time than DAE. The data format was also not officially supported by game engines.

ParaView v5.8. extended its data saving option, including the OBJ format, as an ASCII data type. This enables an alternative data processing methodology, directly from ParaView to Unity without the use of any intermediate 3D computer graphics software. Despite this direct connection, several drawbacks come

along with the integration. Firstly, ParaView uses the OBJ format to store data rather than as an exchange format. This, therefore, prevents multiple data processing at the same time. Moreover, it was observed that 3D internal mesh cannot be saved with OBJ. Visual representations #1 and #4 were solely possible to process with OBJ. Also, due to the data format characteristics, OBJ only includes geometry in the main file. Compared to Case 3 in which OBJ transformed from X3D, the dataset can be transformed with at least 1.4 times higher data size compression and 3.1 times faster processing time.

Researchers should always keep an eye on recent developments in 3D data formats and software. Unofficial data import and export source-codes could be a remedy to carry out a process with targeted 3D data formats. Nonetheless, a thorough examination should be done on data analytics and quality in order to make integration reliable.

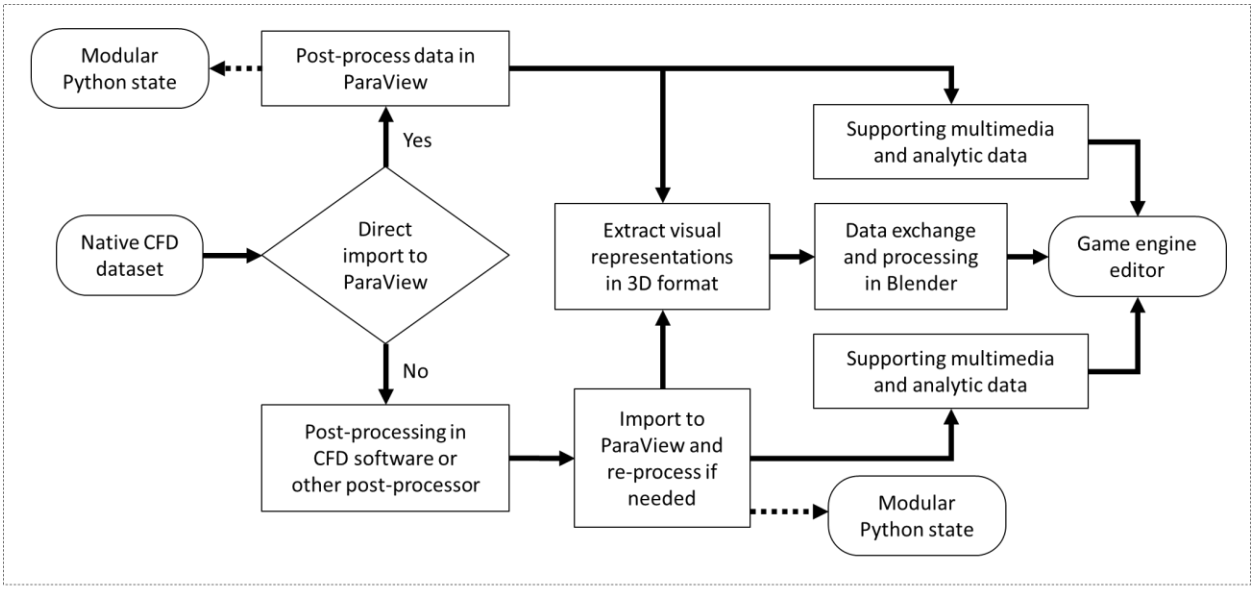
5.3. Data and workflow: automated data processing

One of the key advantages of the system architecture is the detailed data and workflow of the development procedure as shown in

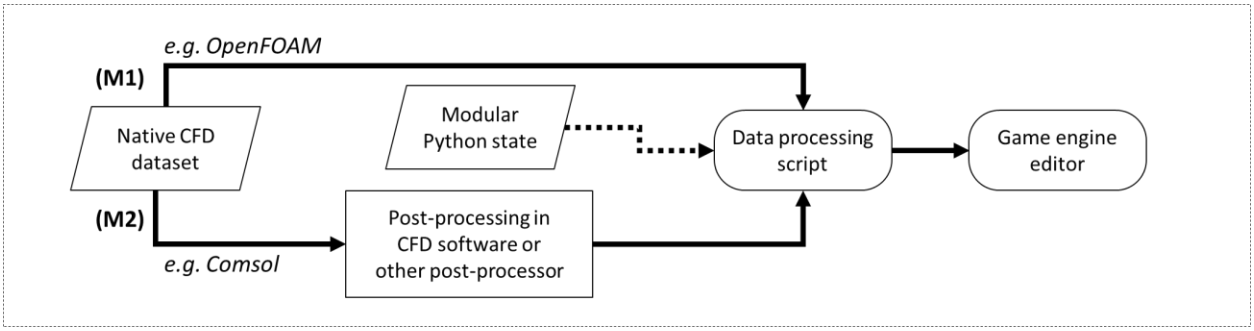
Table *2*. Whereas the system requirements provide features to prescribe given tasks, they also allow for suggestions on which enrichments could possibly be applied to the system components. For instance, the targeted hardware for the end-user console could be incompatible due to the requirements applied in another system component or vice versa. In this regard, the component-oriented trait of system architecture can provide a sustainable development strategy by reiterating development procedures upon the workflow. The data processing approach does not only serve for OpenFOAM, but also COMSOL Multiphysics, Ansys and Siemens Star-CCM+ can be tailored in the processing via compatible data formats (see Supplementary Material 1).

Communication among digital platforms can provide bridges to execute sub-components in an automated scheme. The data processing encompasses multiple manually driven steps that must be set by the practitioners as demonstrated in

Table *2*. In the framework of the present study, bridging between the CFD post-processor and the game engine was achieved to automate data processing based on methodologies M1 and M2 (Fig. 3). Fig. 8 illustrates an algorithmic approach on how to generate a baseline and configure the scheme for an automated process. Either a python state (M1) or data extracts (M2) can be entailed into the automated data processing approach. Once a baseline is generated, any data produced by a CFD solver can directly be processed and integrated to game engine via a data processing script develop in the present study. The automated processing would seem to imply that whole post-processing and data processing workflows can be standardized and provide a rigorous connection. Practitioners can accordingly customize the procedure upon the system requirements with minor changes in the data processing script.



(a) Baseline generation and configuration of processing pipeline.



(b) Automated data processing of configured pipeline.

Fig. 8 An algorithmic approach on data processing; (a) baseline generation; (b) automation of methodologies M1 and M2

A blueprint of the data processing script developed in the present work is detailed in Fig. 9. It essentially provides a simplified, optimized and automated one-way coupling procedure between CFD data and game engine. An input file of CFD data, either a python state or extract, should be set by the user before running the script. The full script was developed in Python with Spyder in Anaconda Navigator utilizing the packages ParaView v5.8. and Blender v7.9. It has capabilities to process both steady-state and transient CFD datasets to generate visual CFD simulation data, supporting multimedia files (e.g. colormap in image format), analytic data, and data processing performance. The Python code is available in the project GitHub repository[1]. It is noteworthy to mention that the platforms entailed in the data processing script should be consistent in terms of released versions to run a coupled dataflow without any disruption. Practitioners can take advantage of automated data processing due to the reduced complexity in data processing and the dataflow. They may further benefit from outsourced tools such as external databases

[1] https://github.com/sersolmaz/CFD_AR_VR

provided by research institutions and simulation companies to process and adapt digital content freely available.

In the next phase, a two-way coupling between the data creation and the end-user console would provide a fully automated integration. This could allow users to manipulate existing conditions and produce new simulation results actively from an optimized CFD workflow. The proposed data processing script originally complies with two-way coupling approach since nothing changes in the data processing approach from CFD solver to game engine. The users' interaction in the digital environment with simulations could be achieved in various techniques based on sub-components and system requirements. As an example, manipulating one of the boundary conditions in simulations could be done via hand-held devices, touch screens, image processing, IoT devices, system-on-chip, voice commands or any other methods that have so far been developed, as well as considering high-tech connection opportunities such as ubiquitous computing.

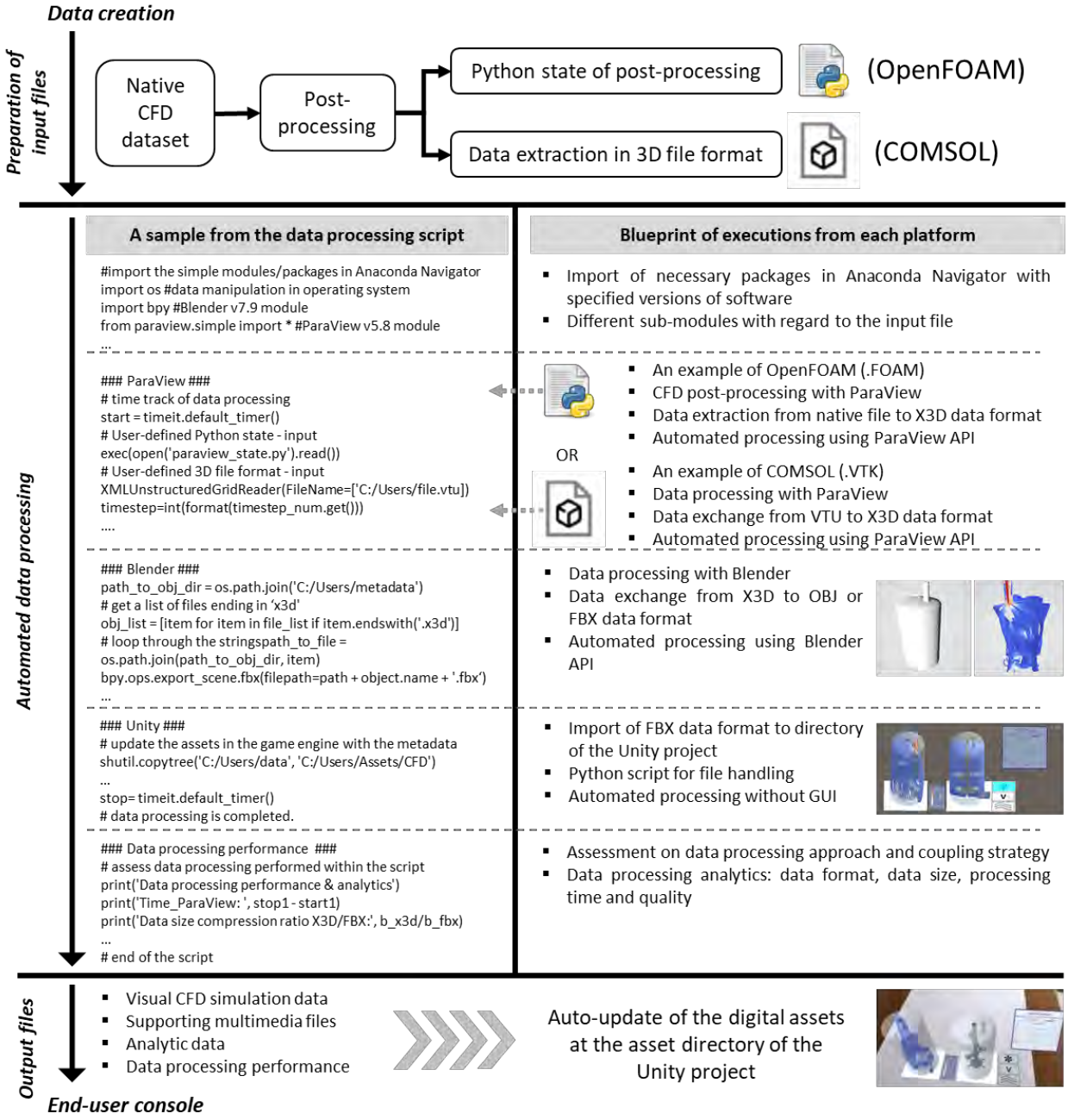


Fig. 9 Automated data processing script from data creation to end-user console, and the blueprint of the entire procedure

5.4. Data management

Both commercial (Ansys, Comsol, and Star-CCM+) and open-source (OpenFOAM) CFD solvers were found to be designed based on well-structured data management systems. Similarly, game engines also relied on management systems to compress and retrieve the data in use. A processed CFD data imported into Unity is automatically stored under a related assets directory where the project was initially created.

Surplus and repetitive metadata generated throughout the processing should be avoided to keep the data size of the final application optimal. Processed data is only stored in the game engine and the final application. Therefore, all desired CFD post-processing should be determined in advance.

In case a new CFD post-processing is to be integrated into the final application, the system should be able to access the native CFD data which is generally large and complex. Instead, extractions from native CFD data into manageable parts can alternatively serve while reducing data by orders of magnitude as previously promoted in Section 3. Extracted CFD datasets should be encoded by using a convenient data model to store, retrieve, reproduce, and reprocess metadata in a secure routine.

A study was prompted to examine data size, data saving time, and retrievability of data storage formats supported by ParaView. A particular time step with a size of 339 MB from the CFD results of the stirred tank reactor was sampled for the comparison. Table 3 shows the reduction in file size against the native CFD dataset. Whilst the smallest data size was reached with XDMF, the shortest operation time was achieved with VTM. All data formats enable us to retrieve CFD data in ParaView without data loss. Further reduction in data size was also achieved with traditional data compression type ZIP. CGNS, VTK, and XDMF gave satisfactory performances on data size and saving time. A database system might be developed to manage CFD parts intelligently using one of these formats whereas a frequent use of the data is targeted.

Table 3 Comparison among data models to store and retrieve CFD dataset with ParaView

Data model	Reduction in file size (%)	Data saving time (s)
VTM	53	0.2
VTM (zip)	77	13.4
CGNS	29	0.8
CGNS (zip)	75	27.2
XDMF	63	9.3
XDMF (zip)	77	16.5
OpenFOAM native file (zip)	27	4.3

6. Strengths and weaknesses

Literature overlooked technical challenges related to the processing of simulation data among cross-platform development tools, as being and inter-disciplinary concern. Researches carried out to data suffered from standalone, platform-specific, computationally-demanding, non-replicable, and manually integrated data processing methodologies. This hindered developers to consider AR/VR tools to visualize CFD simulation data [1, 3]. Our study provided a detailed technical examination regarding the integration of CFD simulation data with AR/VR applications. We developed an extract-based data processing methodology with lightweight, easy-to-implement, end-to-end, automated and free-to-use utilities. The academic community can readily benefit from our study to figure out best scenarios to implement CFD simulation data with AR/VR tools. Not only CFD simulations but also any type of visual data processed with ParaView (or any other post-processing software supporting suitable data formats) can be integrated into the workflow presented in this study.

The present study only investigated commonly used software and data formats in the multiplatform integration based on the feasibility study conducted prior to our investigation (see Supplementary Information 1). Due to the scope of our work, we did not provide any information relevant to the validation CFD simulation results, which is fatal to present accurate information. Likewise, except for virtual buttons in AR application, we obscured human-computer interactions, which are normally considered to one of the fundamental features of AR/VR by means immersion.

Speaking of the challenges for future research, this study concentrated on the integration of CFD simulation data with AR/VR development tools; however, more research is required in this area in order to remotely update integrated data in final user applications. Also, it would be interesting to investigate newly adapted lightweight data formats (for example GLTF) to reduce data size and increase data processing speed while preserving quality.

7. Conclusion

The present work proposed generic system architecture and examined methodologies on the integration of CFD simulations with AR/VR. The results indicated that the system architecture has a promising utility to prescribe a development strategy. The study provided a generic design pattern as well as miscellaneous workflows to develop educational tools with CFD and AR/VR. Extract-based processing of CFD dataset highlighted a simplified integration that the practitioners can readily pursue. A comprehensive investigation was also performed to evaluate data processing methodologies on their data format, size, processing time and quality. Findings had remarkable implications to overcome ambiguities related to data process options in the literature. The present work also disclosed an automated dataflow between native CFD dataset and the game engine. A descriptive guidance on the management of data in use additionally demonstrated to manage large CFD data parts.

In this study, we focused on a rigorous integration that can help practitioners to maintain long-lasting digital tools with rapid and easy-to-update features. Future work will concentrate on two-way data coupling and technical works to prototype such digital tools. User assessment studies will be performed to assess both digital tools and implementation in engineering education.

References

1. Su S, Perry V, Bravo L, et al (2020) Virtual and Augmented Reality Applications to Support Data Analysis and Assessment of Science and Engineering. Comput Sci Eng 22:27–39. https://doi.org/10.1109/MCSE.2020.2971188

2. Dong S, Behzadan AH, Chen F, Kamat VR (2013) Collaborative visualization of engineering processes using tabletop augmented reality. Advances in Engineering Software 55:45–55. https://doi.org/10.1016/j.advengsoft.2012.09.001

3. Li W, Nee A, Ong S (2017) A State-of-the-Art Review of Augmented Reality in Engineering Analysis and Simulation. Multimodal Technologies and Interaction 1:17. https://doi.org/10.3390/mti1030017

4. Suh A, Prophet J (2018) The state of immersive technology research: A literature analysis. Computers in Human Behavior 86:77–90. https://doi.org/10.1016/j.chb.2018.04.019

5. Li J (2015) Approaching virtual process engineering with exploring mesoscience. Chemical Engineering Journal 278:541–555. https://doi.org/10.1016/j.cej.2014.10.005

6. Ge W, Guo L, Liu X, et al (2019) Mesoscience-based virtual process engineering. Computers & Chemical Engineering 126:68–82. https://doi.org/10.1016/j.compchemeng.2019.03.042

7. Blach R, Landauer J, Rösch A, Simon A (1998) A highly flexible virtual reality system. Future Generation Computer Systems 14:167–178. https://doi.org/10.1016/S0167-739X(98)00019-3

8. Harwood ARG (2019) GPU-powered, interactive flow simulation on a peer-to-peer group of mobile devices. Advances in Engineering Software 133:39–51. https://doi.org/10.1016/j.advengsoft.2019.04.003

9. Woods JW (2006) Multidimensional signal, image, and video processing and coding. Elsevier/Academic Press, Amsterdam ; Boston, Mass

10. Stam J (1999) Stable Fluids. SIGGRAPH '99 Proceedings of the 26th annual conference on Computer graphics and interactive techniques 121–128

11. Zhang F, Wei Q, Xu L (2020) An fast simulation tool for fluid animation in VR application based on GPUs. Multimedia Tools and Applications 79:16683–16706. https://doi.org/10.1007/s11042-019-08002-4

12. Huang Z, Gong G, Han L (2014) Physically-based modeling, simulation and rendering of fire for computer animation. Multimedia Tools and Applications 71:1283–1309. https://doi.org/10.1007/s11042-012-1273-z

13. Xu Z, Lu XZ, Guan H, et al (2014) A virtual reality based fire training simulator with smoke hazard assessment capacity. Advances in Engineering Software 68:1–8. https://doi.org/10.1016/j.advengsoft.2013.10.004

14. Harwood ARG, Revell AJ (2018) Interactive flow simulation using Tegra-powered mobile devices. Advances in Engineering Software 115:363–373. https://doi.org/10.1016/j.advengsoft.2017.10.005

15. Slotnick J, Khodadoust A, Alonso J, et al (2013) CFD Vision 2030 Study: A Path to Revolutionary Computational Aerosciences. NASA Langley Research Center

16. He Z, You L, Liu RW, et al (2019) A Cloud-Based Real Time Polluted Gas Spread Simulation Approach on Virtual Reality Networking. IEEE Access 7:22532–22540. https://doi.org/10.1109/ACCESS.2019.2893919

17. Kim M, Yi S, Jung D, et al (2018) Augmented-Reality Visualization of Aerodynamics Simulation in Sustainable Cloud Computing. Sustainability 10:1362. https://doi.org/10.3390/su10051362

18. Cha M, Han S, Lee J, Choi B (2012) A virtual reality based fire training simulator integrated with fire dynamics data. Fire Safety Journal 50:12–24. https://doi.org/10.1016/j.firesaf.2012.01.004

19. Yu Y, Duan M, Sun C, et al (2017) A virtual reality simulation for coordination and interaction based on dynamics calculation. Ships and Offshore Structures 12:873–884. https://doi.org/10.1080/17445302.2017.1293762

20. Harwood ARG, Revell AJ (2017) Parallelisation of an interactive lattice-Boltzmann method on an Android-powered mobile device. Advances in Engineering Software 104:38–50. https://doi.org/10.1016/j.advengsoft.2016.11.005

21. Ham Y, Golparvar-Fard M (2013) EPAR: Energy Performance Augmented Reality models for identification of building energy performance deviations between actual measurements and simulation results. Energy and Buildings 63:15–28. https://doi.org/10.1016/j.enbuild.2013.02.054

22. Natephra W, Motamedi A, Yabuki N, Fukuda T (2017) Integrating 4D thermal information with BIM for building envelope thermal performance analysis and thermal comfort evaluation in naturally ventilated environments. Building and Environment 124:194–208. https://doi.org/10.1016/j.buildenv.2017.08.004

23. Kirby AC, Yang Z, Mavriplis DJ, et al (2018) Visualization and Data Analytics Challenges of Large-Scale High-Fidelity Numerical Simulations of Wind Energy Applications. In: 2018 AIAA Aerospace Sciences Meeting. American Institute of Aeronautics and Astronautics, Kissimmee, Florida

24. Badías A, Curtit S, González D, et al (2019) An augmented reality platform for interactive aerodynamic design and analysis. Int J Numer Methods Eng 120:125–138. https://doi.org/10.1002/nme.6127

25. Duque EP, Imlay ST, Ahern S, et al (2016) NASA CFD Vision 2030 Visualization and Knowledge Extraction: Panel Summary from AIAA AVIATION 2015 Conference. In: 54th AIAA Aerospace Sciences Meeting. American Institute of Aeronautics and Astronautics, San Diego, California, USA

26. Karmonik C, Boone TB, Khavari R (2018) Workflow for Visualization of Neuroimaging Data with an Augmented Reality Device. J Digit Imaging 31:26–31. https://doi.org/10.1007/s10278-017-9991-4

27. Bergonzi L, Colombo G, Redaelli D, Lorusso M (2019) An Augmented Reality Approach to Visualize Biomedical Images. CAD&A 16:1195–1208. https://doi.org/10.14733/cadaps.2019.1195-1208

28. Fukuda T, Yokoi K, Yabuki N, Motamedi A (2018) An indoor thermal environment design system for renovation using augmented reality. Journal of Computational Design and Engineering. https://doi.org/10.1016/j.jcde.2018.05.007

29. Sastry L, Boyd DRS (1998) Virtual environments for engineering applications. Virtual Reality 3:235–244. https://doi.org/10.1007/BF01408704

30. Wasfy TM, Noor AK (2001) Visualization of CFD results in immersive virtual environments. Advances in Engineering Software 32:717–730. https://doi.org/10.1016/S0965-9978(01)00020-5

31. Zhu Y, Fukuda T, Yabuki N (2019) Integrating Animated Computational Fluid Dynamics into Mixed Reality for Building-Renovation Design. Technologies 8:4. https://doi.org/10.3390/technologies8010004

32. Martins P, Pinto S, Andre A (2019) Interactive DEMOnstration of Medical Simulations using a Virtual Reality approach: Application to the Male Urinary System. In: 2019 5th Experiment International Conference (exp.at'19). IEEE, Funchal (Madeira Island), Portugal, pp 251–252

33. Zhao S, Zhang L, DeAngelis E (2019) Using augmented reality and mixed reality to interpret design choices of high-performance buildings. pp 435–441

34. Saitoh T, Noguchi G, Inoue T (2018) Tsunami Run-Up Simulation Using Particle Method and its Visualization with Unity. In: THE 9TH INTERNATIONAL CONFERENCE ON COMPUTATIONAL METHODS (ICCM2018)

35. Huang J, Ong SK, Nee AY-C (2019) An approach for augmented learning of finite element analysis. Comput Appl Eng Educ. https://doi.org/10.1002/cae.22125

36. Wheeler G, Deng S, Toussaint N, et al (2018) Virtual interaction and visualisation of 3D medical imaging data with VTK and Unity. Healthcare Technology Letters 5:148–153. https://doi.org/10.1049/htl.2018.5064

37. Lin J-R, Cao J, Zhang J-P, et al (2019) Visualization of indoor thermal environment on mobile devices based on augmented reality and computational fluid dynamics. Automation in Construction 103:26–40. https://doi.org/10.1016/j.autcon.2019.02.007

38. Tamura Y, Nakamura H, Fujiwara S (2016) An Intuitive Interface for Visualizing Numerical Data in a Head-Mounted Display with Gesture Control. Plasma and Fusion Research 11:2406060–2406060. https://doi.org/10.1585/pfr.11.2406060

39. Quam DJ, Gundert TJ, Ellwein L, et al (2015) Immersive Visualization for Enhanced Computational Fluid Dynamics Analysis. Journal of Biomechanical Engineering 137:031004. https://doi.org/10.1115/1.4029017

40. García-Hernández RJ, Kranzlmüller D (2019) NOMAD VR: Multiplatform virtual reality viewer for chemistry simulations. Computer Physics Communications 237:230–237. https://doi.org/10.1016/j.cpc.2018.11.013

41. El Beheiry M, Doutreligne S, Caporal C, et al (2019) Virtual Reality: Beyond Visualization. Journal of Molecular Biology 431:1315–1321. https://doi.org/10.1016/j.jmb.2019.01.033

42. Jüttner M, Zhao N, Grabmaier S (2017) A Standalone Interface for Web-Based Virtual Reality of Calculated Fields. In: Proceedings of the 2017 COMSOL Conference in Rotterdam

43. Zhao S, Jin S, Ai C, Zhang N (2019) Visual analysis of three-dimensional flow field based on WebVR. Journal of Hydroinformatics 21:671–686. https://doi.org/10.2166/hydro.2019.101

44. Berger M, Cristie V (2015) CFD Post-processing in Unity3D. Procedia Computer Science 51:2913–2922. https://doi.org/10.1016/j.procs.2015.05.476

45. Yao J, Lin Y, Zhao Y, et al (2018) Augmented reality technology based wind environment visualization. In: Learning, Adapting and Prototyping - Proceedings of the 23rd CAADRIA Conference. pp 369–377

46. Kim R, Kim J, Lee I, et al (2019) Development of a VR simulator for educating CFD-computed internal environment of piglet house. Biosystems Engineering 188:243–264. https://doi.org/10.1016/j.biosystemseng.2019.10.024

47. Horton BK, Kalia RK, Moen E, et al (2019) Game-Engine-Assisted Research platform for Scientific computing (GEARS) in Virtual Reality. SoftwareX 9:112–116. https://doi.org/10.1016/j.softx.2019.01.009

48. Sicat R, Li J, Choi J, et al (2019) DXR: A Toolkit for Building Immersive Data Visualizations. IEEE Trans Visual Comput Graphics 25:715–725. https://doi.org/10.1109/TVCG.2018.2865152

49. Cordeil M, Cunningham A, Bach B, et al (2019) IATK: An Immersive Analytics Toolkit. In: 2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR). IEEE, Osaka, Japan, pp 200–209

50. Lau KK, Cola SD (2017) An Introduction to Component-based Software Development. World Scientific, Singapore

51. Gianni D (2014) Modeling and Simulation-Based Systems Engineering Handbook, 1st ed. CRC Press

52. Moloney J, Globa A, Wang R, Khoo C (2019) Principles for the application of mixed reality as pre-occupancy evaluation tools (P-OET) at the early design stages. Architectural Science Review 1–10. https://doi.org/10.1080/00038628.2019.1675138

53. Lai JWM, Bower M (2019) How is the use of technology in education evaluated? A systematic review. Computers & Education 133:27–42. https://doi.org/10.1016/j.compedu.2019.01.010

54. Hamilton ER, Rosenberg JM, Akcaoglu M (2016) The Substitution Augmentation Modification Redefinition (SAMR) Model: a Critical Review and Suggestions for its Use. TechTrends 60:433–441. https://doi.org/10.1007/s11528-016-0091-y

55. Lemahieu W (2018) Principles of Database Management: The Practical Guide to Storing, Managing and Analyzing Big and Small Data. Cambridge University Press

56. Schilling A, Bolling J, Nagel C (2016) Using glTF for streaming CityGML 3D city models. In: Proceedings of the 21st International Conference on Web3D Technology - Web3D '16. ACM Press, Anaheim, California, pp 109–116

57. Xu Z, Zhang L, Li H, et al (2020) Combining IFC and 3D tiles to create 3D visualization for building information modeling. Automation in Construction 109:102995. https://doi.org/10.1016/j.autcon.2019.102995

58. Salehi V, Wang S (2018) Web-Based Visualization of 3D Factory Layout from Hybrid Modeling of CAD and Point Cloud on Virtual Globe DTX Solution. CAD&A 16:243–255. https://doi.org/10.14733/cadaps.2019.243-255

59. Julin A, Jaalama K, Virtanen J-P, et al (2019) Automated Multi-Sensor 3D Reconstruction for the Web. IJGI 8:221. https://doi.org/10.3390/ijgi8050221

60. Jung K, Nguyen VT, Yoo S-C, et al (2020) PalmitoAR: The Last Battle of the U.S. Civil War Reenacted Using Augmented Reality. IJGI 9:75. https://doi.org/10.3390/ijgi9020075

Declarations

Funding

This project has received funding from the European Union's EU Framework Programme for Research and Innovation Horizon 2020 under Grant Agreement 812716. This publication reflects only the author's view exempting the community from any liability. Project website: https://charming-etn.eu/.

Conflict of interest

The authors declare that they have no conflict of interest.

Supplementary material

Supplementary material 1 associated with this article can be found, in the online version, at https://github.com/sersolmaz/CFD_AR_VR/blob/master/Supplementary%20Materials/Supplementary%20Material%201.pdf.

Supplementary material 2 associated with this article can be found, in the online version, at https://github.com/sersolmaz/CFD_AR_VR/blob/master/Supplementary%20Materials/Supplementary%20Material%202.pdf.

Code availability

The custom code developed in this work is available at https://github.com/sersolmaz/CFD_AR_VR.