PLUME: Record, Replay, Analyze and Share User Behavior in 6DoF XR Experiences

Charles Javerliat, Sophie Villenave, Pierre Raimbaud, Guillaume Lavoué

PLUME: Record, Replay, Analyze and Share User Behavior in 6DoF XR Experiences

Charles Javerliat*, Sophie Villenave*, Pierre Raimbaud and Guillaume Lavoué

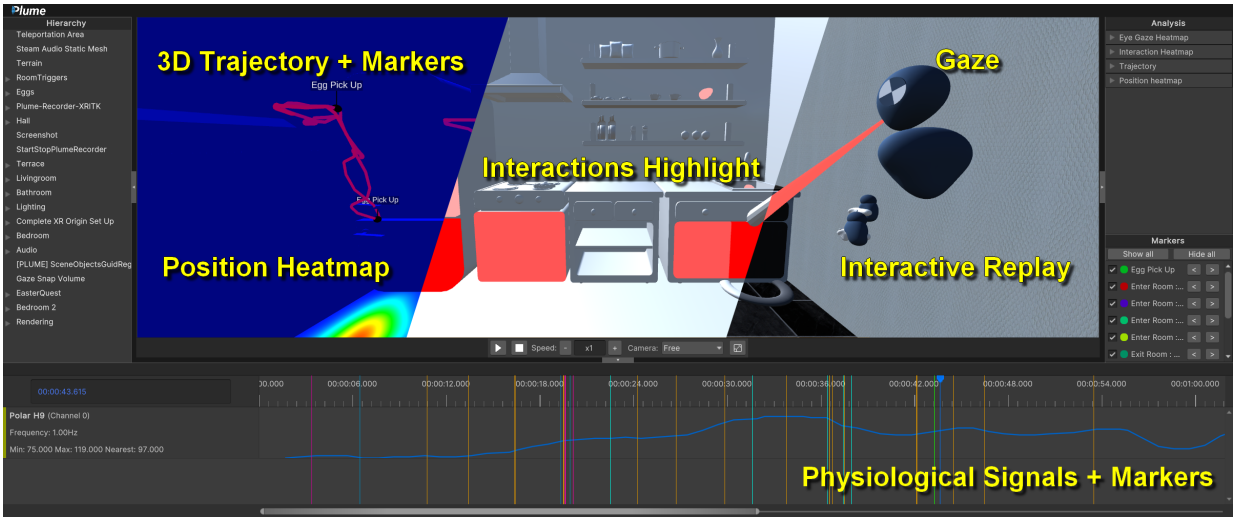


Fig. 1: The standalone PLUME Viewer application allows for the interactive visualization and analysis of behavioral and physiological data recorded with the PLUME Recorder Unity plugin. From left to right: hierarchy of objects in the virtual environment; heatmap of user position and 3D trajectory, highlighting of the most interacted objects; gaze direction; analysis control panel for visualizations; custom event markers; timeline with synchronized physiological signals tracks and markers.

Abstract— From education to medicine to entertainment, a wide range of industrial and academic fields now utilize eXtended Reality (XR) technologies. This diversity and growing use are boosting research and leading to an increasing number of XR experiments involving human subjects. The main aim of these studies is to understand the user experience in the broadest sense, such as the user cognitive and emotional states. Behavioral data collected during XR experiments, such as user movements, gaze, actions, and physiological signals constitute precious assets for analyzing and understanding the user experience. While they contribute to overcome the intrinsic flaws of explicit data such as post-experiment questionnaires, the required acquisition and analysis tools are costly and challenging to develop, especially for 6DoF (Degrees of Freedom) XR experiments. Moreover, there is no common format for XR behavioral data, which restrains data-sharing, and thus hinders wide usages across the community, replicability of studies, and the constitution of large datasets or meta-analysis. In this context, we present PLUME, an open-source software toolbox (PLUME Recorder, PLUME Viewer, PLUME Python) that allows for the exhaustive record of XR behavioral data (including synchronous physiological signals), their offline interactive replay and analysis (with a standalone application), and their easy sharing due to our compact and interoperable data format. We believe that PLUME can greatly benefit the scientific community by making the use of behavioral and physiological data available for the greatest, contributing to the reproducibility and replicability of XR user studies, enabling the creation of large datasets, and contributing to a deeper understanding of user experience.

Index Terms—Extended Reality, Virtual Reality, User Behavior, Human-Computer Interaction, Quality of Experience, Data Collection, Physiological Signals.

1 Introduction

Application areas for eXtended Reality (XR) can be very diverse, from entertainment to training, marketing, cultural heritage, or even therapy. XR technologies accessibility and popularity have significantly improved over the last decade, leading to an exponential increase of research studies in XR environments. Even though each experiment has its own specificity, one usual main objective is to **understand user experience**. This can take several forms i.e., with different focuses: i) the evaluation of the quality of user experience according to newly designed devices or systems, in various senses (e.g., user presence, immersion, flow, or sensory perceived quality), ii) the acquisition of data to feed behavior prediction models (e.g., deep learning networks for gaze prediction), iii) the evaluation of an immersive system from its domain-specific perspective (e.g., therapy effects, learning effects), or iv) the improvement of global knowledge on human beings understanding (e.g., cognitive processes, sociological aspects).

In research studies, the understanding of user experience in virtual environments (VE) can derive from subjective or objective data collected during experiments. Halbig and Latoschik proposed a classification relying on the way users provide data, between explicit – when users are asked to rate, describe etc. their experience, and implicit data – collected without asking users for dedicated actions to do so [27]. Con-

*All authors are with École Centrale de Lyon, CNRS, LIRIS UMR5025, ENISE. E-mails: {charles.javerliat, sophie.villenave, pierre.raimbaud}@ec-lyon.fr, guillaume.lavoue@enise.ec-lyon.fr. * These authors contributed equally and should be seen as joint first authors.*

Manuscript received xx xxx. 201x; accepted xx xxx. 201x. Date of Publication xx xxx. 201x; date of current version xx xxx. 201x. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org. Digital Object Identifier: xx.xxxx/TVCG.201x.xxxxxxx

cretely, explicit data correspond to self-reported data and implicit data to behavioral and physiological ones. Self-reported data are usually collected via tools such as questionnaires (e.g. presence [65,79], embodiment [62], usability [7], workload [28]), or semi-structured interviews. Behavioral data are usually collected in the VE (XR controller inputs, interaction traces) or the real world (positions of XR Head-Mounted Display – HMD, controllers or trackers). Physiological data such as heart rate, eye-tracking, electrodermal activity, electrocardiogram and electroencephalography are collected directly on users. In the literature, the majority of XR studies use self-reported data to validate their hypotheses. Popularity of this kind of data can be explained by their simplicity of acquisition, analysis and sharing. However, relying only on them has limitations: post-experiment data collection means that users assess the memory of their experience rather than their actual experience [36]; questionnaires easily become outdated and unsuitable to evaluate the newest technologies [25]; generic questionnaires can be irrelevant to study-specific content [34], and biases can be introduced by the users e.g., by trying to guess the right answer, adjusting rates depending on the proposed scale over the asked items [12], or misinterpreting questions because of language translations [3]. To avoid these limitations, researchers are increasingly collecting behavioral and physiological data in addition to self-reported ones to address their questions [27]. However, the former are more tedious to acquire, analyze and share, especially in 6DoF (Degrees of Freedom) XR studies where e.g., the mapping of eye tracking data onto the 3D environment is more complex to implement, as well as the logging of interactions and events in a freely-accessible 3D space and the synchronization of physiological data with body and ocular 3D motions. Using behavioral and physiological data in XR studies is thus not readily accessible at the moment, even more so for research teams whose academic field is outside computer science as it requires many specific developments.

Using behavioral and physiological data in XR studies is also hampered by the lack of standardization in their representation, making them difficult to share and reuse for experiment replication, follow-up studies or meta-analyses. Moreover, the heterogeneity of data representation across studies makes almost impossible to aggregate data from different experiments. This prevents the creation of datasets of user behavior in 6DoF-XR, making them rare or even non-existent. Consequently, the construction of models able to classify behavior or predict body or gaze motions remains underdeveloped in 6DoF-XR.

In this context, we introduce PLUME, an open-source software toolbox (PLUME Recorder, PLUME Viewer, PLUME Python) to facilitate the collection, exploitation and sharing of behavioral and physiological data from 6DoF-XR experiments created with Unity [76]. Its main features are: **i)** exhaustive capture of VE data (objects positions, orientations, appearance, user interactions, gaze, custom events) with the PLUME Recorder plugin for Unity; **ii)** synchronous capture of physiological data with Lab Streaming Layer [21]; **iii)** offline 3D interactive replay of XR sessions and gathered data interactive visualization inside the VE (3D trajectories, 6DoF position heatmap, 6DoF visual attention heatmap, interaction heatmap, physiological signals timeline) with PLUME Viewer; **iv)** standardized data format for ease of use, sharing, replicability, comparison and integration into analysis pipelines; **v)** scripts to load and export recorded data in common formats and frameworks (CSV, dataframe, XDF...). PLUME contributions are:

- A fast, lightweight and accurate recording engine that is easy to install in any new or pre-existing Unity project with little to no configuration required.
- A modular architecture allowing both ease of use and a high degree of customization for users with specific needs.
- In-context 6DoF user behavior visualizations in a standalone application including a new 6DoF eye-gaze projection algorithm for visual attention visualization in dynamic XR environments.

To enable the widest adoption of our framework, PLUME is open-source and available on GitHub[1].

[1]GitHub Page: https://www.github.com/liris-xr/PLUME

PLUME is the result of months of development, and required cross-disciplinary skills in software architecture, computer graphics and XR. We believe that PLUME can greatly benefit the scientific community, enabling the acquisition and analysis of complex behavioral data for all academic teams, even those without computer engineering resources. PLUME is our commitment to enable the replication of XR user studies, improve results repeatability by offering an open and interoperable acquisition pipeline and data format, and enable the production of large scale datasets and meta-analyses. We also propose a novel, interactive and accessible way of communicating 6DoF-XR experiments with the PLUME Viewer standalone application. This paper focuses on the use of PLUME for XR applications, but it can be used for a larger variety of Unity-based application (2D, 3D, XR, networked).

The following section surveys related work in the capture and visualization of XR user data. Next, we present PLUME main functionalities, followed by a description of its architecture and an in-depth explanation of our 6DoF visual attention heatmap visualization technique. We present a PLUME usage case-study, along with an evaluation of the tool impact on runtime performances in XR. We discuss the ethics of capturing and using user data from XR studies, notably not self-reported ones. Finally, we conclude on our framework, and outline our future work.

2 Related Work

In this section, we first give an overview of the types of data currently collected during user studies in XR. Among these data, visual attention data are particularly difficult to collect and analyze in 3D 6DoF environments. Then, as PLUME presents a novel algorithm for their visualization, we dedicate here a specific subsection on the existing techniques for visualization of visual attention, especially heatmaps. Finally, we present and discuss existing tools that contribute to facilitate behavioral and physiological data collection and visualization in XR.

2.1 Data Collection to Evaluate User Experience in XR

In various research and industry fields, XR is used to run studies that seek understanding of the lived experience. We review here the different types of data that can be collected in this context.

2.1.1 Self-Reported Questionnaires

The most widely used kind of tool to assess user experience in XR studies is the questionnaire, as it is a simple and affordable way to collect data in such studies. Questionnaires can be used to assess a wide range of factors, including quality of experience [72], perceptual quality of content [4], sense of presence in the VE [65, 79], system usability [28] or cybersickness [35]. Most often, these questionnaires are administered after the XR experience [36], measuring more the memory of the experience than the experience itself. Moreover, despite their psychometric validity, the administration of questionnaires is subject to various biases [12], impacting on the reliability of the results.

2.1.2 Behavioral Data

User Trajectory and Movement Users' trajectories in the VE are a strong indicator of their behavior. In a 6DoF experiment, a trajectory is a time series of an object's position and rotation, the object often being the head (through the HMD) or the hands (through XR controllers or trackers). Trajectories can be used to compare navigation conditions, as in Berton et al. [5] who studied differences introduced by haptic sensory feedback, or in Raimbaud et al. [57] depending on gazing conditions from a virtual crowd. Visualizing users' trajectories in VE context can facilitate their comparison between users as shown by Homps et al. [31] who offered an immersive tool to display trajectories and interact with their visualizations. The recent development of machine learning algorithms for time-series analysis makes possible the classification of 6DoF experiment trajectories that can be used for short-term prediction of motions [42], future events prediction (e.g., to highlight risks or user interface elements) or to cluster user behavior from motion data [61]. Also using machine learning, with user trajectory compression, Monteiro et al. [49] predicted cybersickness when navigating uncomfortable VEs such as VR games. Nonetheless, these models remain rare because

of a lack of motion datasets from 6DoF experiments and dedicated open-source analysis and visualization software.

Eye Gaze Understanding human visual attention during immersive experiences is crucial for many applications: predicting visual attention [32,43,44,64], optimizing storytelling and game balancing [37], enabling virtual humans with realistic gaze behaviors [24], improving rendering performance (e.g., using Variable Rate Shading) by focusing the computations on the areas of interest from the user perspective [45] etc. The visualization and analysis of visual attention in the VE context can be achieved with real-time replay of gaze data, showing the gaze with a sphere, a ring or a trail [68]. Accumulated gaze data visualization can be done with 3D scanpath where spheres represent gaze accumulations and lines saccadic movements, although it is more commonly done by building heatmaps from eye-tracking data. Such visual attention mapping is well mastered for 3 Degrees of Freedom (3DoF) experiences (*i.e.*, with 360 images or videos), but much less for 6DoFs data where the user can freely move in the 3D space and interact with objects. Section 2.2 provides more details about the existing techniques and challenges to generate 6DoF eye-gaze heatmaps.

User Input and Actions To interact with a VE, users can use a variety of input tools, such as XR controllers, their hands or their eyes. A sequence of inputs enables them to perform actions – selecting, manipulating, navigating, using interfaces [6], the latter allowing for the completion of user tasks [1,56]. Logging the input and the resulting actions can be a strong indication of user behavior. These indicators are often used in learning analytics [19] to better understand learners' behavior when interacting with a user interface, e.g., a gamified learning platform [40]. In VR, the use of these indicators remains rare [36], as the tasks to be performed are not the same from an experiment to another, requiring further development to clearly decompose and log them. In this regard, Wu et al. proposed an XR application design framework called GuST-3D [80] that enables tasks to be defined by the developer and decomposed into actions for automatic recording.

2.1.3 Physiological Signals

Given limitations of self-reported data for behavioral evaluations when using XR technologies, researchers are showing an increasing interest in measuring physiological signals to infer the cognitive state of the user during the XR experience. Physiological signals that can be measured are notably the heart rate (HR), heart rate variation (HRV), electrocardiogram (ECG), pupil dilatation, electrodermal activity (EDA) and electroencephalogram (EEG). For example, previous studies have used EEG data to measure cognitive workload [74], a combination of EEG, HR and EDA data for anxiety [11], or a combination of ECG and respiration data to measure stress [71]. Physiological signals acquisition has improved in the last years since physiological sensors have become more affordable, and the integration of these measurements into research workflows has become more accessible through tools such as the Lab Streaming Layer (LSL) framework. The use of physiological sensors in 3DoF VR experiments has thus become more widespread, leading to the production of high-quality multimodal datasets [26,70]. However, using physiological sensors in 6DoF-XR studies remains complicated as they are sensitive to user motions, which introduces noise that can rapidly invalidate collected data. Another factor that hinders the use of physiological signals is the interpretability of data. Although some signals have been shown to be good indicators of stress or arousal, it is still difficult to demonstrate high-level psychological states such as presence or feelings about the quality of experience. Therefore, there is a need for more 6DoF-XR studies using physiological data, and to achieve this, their use and analysis should be simplified. PLUME is a step towards this goal, enabling simultaneous capture, visualization, and analysis of physiological signals in conjunction with behavioral data from 6DoF-XR experiments.

2.2 6DoF Visual Attention Maps

2D images visual attention maps are usually created applying Gaussian convolutions on gaze fixation points. This process aims at modeling the gaze dispersion around the points of fixation (due to the size of the fovea and the imprecision of tracking device). In 6DoF XR experiments, the difficulty to use this process lies in the eye-gaze projection in the 3D space instead of 2D, and in mapping data in dynamic 3D scenes.

Previous works on 6DoFs visual attention maps include various techniques to map the data onto a 3D scene. Pfeiffer et al. [54] proposed a *volume-based* representation via a voxel grid storing the eye-gaze projection values in each cell and applying a 3D convolution. While straightforward, it is bound to the world space and does not apply with dynamic scenes with moving objects. Moreover, a volumetric density map does not fully respect occlusion. Stellmach et al. [67] proposed a coarse-grained *object-based* representation with per object scalar values. It contains information about which objects are fixated the most, but without details on the fixation area on the mesh surface. It is applicable to dynamic scenes as the values are linked to objects and not to world space, but is restricted to coarse-grained analysis. For finer results, *surface-based* representations highlight the parts of the mesh that have been fixed. This is particularly useful to get information on which sections of a mesh attracts the most attention. The values can be stored directly alongside vertices [18,41,67] or triangles. However, this method is strongly dependent on the mesh sampling resolution, and does not allow for the mapping of data at higher resolution and uniformly inside a triangle. This can cause issues, e.g., with meshes with very large triangles such as walls and floors where the eye-gaze or position projection may land inside a limited region of a triangle.

To overcome limitations due to mesh resolution, Maurus et al. [46] proposed a *pixel-based* method that projects a Gaussian from screen space onto surfaces to form the visual attention map. They added the use of z-buffers to determine occluded areas for each point of view. This technique has the advantage of being relatively fast as it does not require computing raycasts into the scene for occlusion testing. However, it is GPU-memory expensive as each point of view requires a z-buffer, making it not scale well for long records. Moreover, it is not designed to work with dynamic scenes as data are not bound to objects. To keep the benefits of a detailed visual attention map without being too heavy memory-wise and to have values bound to separate scene objects, Pfeiffer and Memili [55] proposed a *texel-based* method that stores values into a texture image. While being suitable for real-time and dynamic scenes, the computed texture is heavily coupled with the UV mapping of objects, which can be missing (e.g. for CAD objects), overlapping or non-uniform. To tackle these issues, within the PLUME Viewer, we propose a real-time surface-based mapping method agnostic to UV mapping and which resolution can go beyond the mesh one.

2.3 Software for Behavioral and Physiological Data Collection and Visualization in 6DoF XR

In the scientific literature as well as in the industry, software tools have been proposed to collect and visualize or even replay behavioral or/and physiological data of XR users, possibly in the VE context. For *behavioral* data like *motions*, REC [23] has been designed as a Unity open-source plugin to record users' body motions in XR and transfer them to avatars afterwards. For user *trajectories*, the CrowdMP [59] open-source project allows for the recording and replay in Unity VEs of a VR user's displacements among a crowd of virtual humans whose runtime trajectories are recorded and replayed too. The EyeCVE system [50], among others, had been developed to capture and replay multiple CAVEs users' *gaze*. For *user inputs and actions*, the commercial software NVIDIA VCR [52] enables their capture, editing (filtering, assembling) and replay, for OpenVR applications only. To generalize to more XR applications and to ease their design, where such user data would be logged and analyzed, Wu, Robert and colleagues proposed the framework GuST-3D [60,80]. This "design framework" approach is in line with other tools such as UXF [8] that focused on logging behavioral events, MRAT [51] that enabled in-situ replays in the real world for Mixed Reality (MR) applications, and Ubiq [66] that was designed to create supervised remote and collaborative XR studies. For *physiological* data, LSL4Unity [20] can be used to record physiological sensor signals synchronously with virtual objects positions.

The analysis of the aforementioned tools highlights challenges for this kind of software, beyond signals capture and restitution: i) the syn-

Tool	License	Collected Data	Visualization Tools	Use	Format
REC [23]	GPLv3	Skeleton Transforms	Skeleton Animation Replay	1	CSV
CrowdMP [59]	MIT	User / Agents Transform	-	1	CSV
NVIDIA VCR [52]	Proprietary	HMD and Controller Inputs	3D Playback	1	Binary
GuST-3D [80]	CeCILL	User Transform and Eye Tracking*, Physiological Signals*	Actions Graph	1	BVH, CSV
UXF [8]	MIT	User / Objects Transforms, Mouse, Custom	-	1, 3	CSV, JSON, DB
MRAT [51]	N/A	User / Objects Transforms, Eye Tracking, Tasks, Gesture and Voice Commands, Event Markers, Screenshots, Custom	Timeline, Floor Plan	4, 5	JSON
Ubiq [66]	ALv2	Experiment Logs and Network Traces	-	1-5	JSON
LSL4 Unity [20]	MIT	User / Objects Transforms, Eye Tracking*, Physiological Signals, Custom	-	1, 2	XDF
ReLive [33]	MIT	User / Objects Transforms, Audio, Meshes, Questionnaires	3D Playback, Trajectories, Events Timeline, Additional Metrics, Custom	4, 5	JSON
MIRIA [10]	MIT	User Transform, Interactions, Events	Trajectories, Position Heatmap, Scatterplots	4	CSV
XREcho [77]	MPLv2	User / Objects Transforms, Meshes, Eye Tracking*	3D Playback, Position Heatmap, Trajectories	1	CSV
VRSTK [17, 29]	MIT	User / Objects Transforms, Eye Gaze*, Physiological Signals*, Questionnaires, Custom	3D Playback, Position Heatmap, Object Look Graph	1	JSON, CSV
Tobii Ocumen [73]	Proprietary	Eye Tracking*, User / Objects Transforms, Colliders, Meshes, Rendered Video	3D Playback, Eye Gaze Heatmap, Gaze Plots, Perception Maps, Object Total Gaze Time, Object Focus Count, Correlated Objects	1	Custom, CSV
Vizard [69]	Proprietary	User / Objects Transforms, Meshes, Eye Tracking*, Physiological Signals*	Eye Gaze Path, Eye Gaze Heatmap, Intersects and Fixations	1	CSV
Cognitive 3D [14]	Proprietary	User / Objects Transforms, Meshes, Eye Tracking, Physiological Signals*, Audio, Events	3D Playback, Eye Gaze Heatmap, Physiological Signals	1, 4, 5	JSON
PLUME (ours)	GPLv3	User / Objects Transforms, Audio, Meshes, Lights, Canvas (UI), Eye Tracking, Interactions, Event Markers, Physiological Signals, Custom	3D Playback, Position / Eye-Gaze / Interaction Heatmaps, Trajectories, Physiological Signals and Events Markers	1-7	PLM, CSV, XDF, PLY**

*Compatible with a reduced range of devices **Heatmaps Point Clouds (1) PC-VR (2) Standalone-VR (3) Web-VR (4) AR (5) MR (6) 2D (7) 3D

Table 1: Comparison of available software for behavioral and physiological data collection and visualization (in 6DoF). The comparison is based on information available in original papers and code repositories.

chronous capture of different kinds of data and from multiple sources, ii) the visualization of VE-contextualized signals i.e., in the light of the VE state (user viewport, objects positions, time markers of previous interactions etc.), iii) the ease of the recording tool integration into existing projects, iv) the production of record files in interoperable formats. From this, other studies attempted to address some of these challenges simultaneously. ReLive [33] and MIRIA [10] offer in/ex-situ visualizations from multiple participants' motions and actions, only in MR and without handling physiological data. AvatAR [58] proposes advanced visualizations of behavioral data in MR: gaze collisions and footprint visualization frame by frame, user touches on objects, 3D trajectories and trajectory heatmaps for post-analyses. However, no code or software are available. For VR, XREcho plugin for Unity [77] and VRSTK [17, 29] enable the recording and replay of contextualized sessions with advanced visualizations such as gaze collisions on the VE and trajectory heatmaps. Both remain limited for physiological data capture and data visualization, e.g. with the absence of gaze heatmaps (see Section 2.2). Tobii Ocumen commercial software [73] enables the capture of VR user's gaze, hand and head motions. It provides contextualized replays based on virtual objects position and user inputs recordings in a standalone viewer. Eye-related physiological data such as pupil dilation are recorded too; however this software remains mainly gaze-oriented with no options for other physiological data integration.

The two most complete software – commercial, are Vizard [69] and Cognitive3D [14]. Their main strength is the capture and visualization of behavioral but also physiological data. The former is a Python-based VR development solution that integrates visualization tools; the latter is a plugin for Unity and Unreal Engine that can be integrated to existing experiments. Their accessibility remains a concern, partly because of their cost, but also Vizard's incompatibility with Unity, and Cognitive3D's dependence on proprietary servers for data storage and processing. Table 1 provides a comparison of tools functionalities.

According to our review of the current literature, available software address parts of the challenges previously listed (synchronous capture of data, VE-contextualized visualizations, ease of integration within XR applications, interoperability), but none answers them all for 6-DoF XR experiments. PLUME addresses all these challenges within an open-source and modular architecture, focusing on community engagement to foster improvements and adaptations. It allows the capture of both behavioral and physiological data, and provides in-context visualizations in a standalone viewer (including a new 6DoF eye-gaze projection algorithm), even for built applications. PLUME can be easily integrated into any Unity-based experiment and XR device (even standalone HMDs). Its interoperable data format and open architecture enable the development of new analysis modules by the community. Finally, we ensure the preservation of XR performances and thus user experience, through an in-depth evaluation of PLUME's possible impact at runtime, a question mainly not tackled in other tools.

3 Motivations and Design Choices

In this section, we highlight our motivations for the creation of PLUME and its main contributions to the scientific community.

Review and Share 6DoF-XR Experimentations Our tool can be used to run post-experimental checks to ensure that user sessions in XR were conducted as expected. Reviewing sessions through PLUME can help understanding unforeseen causes of outlier recordings, if post-hoc data analyses had revealed unexpected results. PLUME also offers an interactive way to share experiment content, helping reviewers and readers to understand experimental protocols.

Interactive Data Exploration To facilitate the analysis of XR experiments data, which are multimodal and dense, we propose an interactive visualization tool that links the recorded data to the VE context. PLUME Viewer allows reviewers, researchers, or developers to review an experience. Unlike videos where only one viewpoint is displayed, our interactive 3D tool enables exploration of the VE, and the

display of visualizations on it (trajectories, visual attention, interactions, physiological signals). These visualisation tools are precious assets to visualize and understand XR users' behavior and experience.

Create a File Format for XR Behavioral Data Conducting experiments using XR technologies is complex and time-consuming, often limiting the number of people per experiment. Paradoxically, statistical studies need large numbers of data to validate a result, as XR studies involve subjects with strong interpersonal differences that introduce uncontrollable biases. This observation is particularly relevant to experiments where users move with 6DoF, since more options are available to them e.g., moving directions, time to interact with objects. To facilitate data sharing and building large cross-laboratory datasets, PLUME defines an open, compact, language-neutral and platform-neutral file format that contains data from Unity, LSL, and even custom data.

Replicability, reproducibility and reliability of XR Experiments The lack of replicability and reproducibility is a major issue in XR research [30], as in many other fields that involve human subjects, from psychology to neuroscience. This so-called replication crisis is due to several factors such as intrinsic bias of subjective data collection, lack of raw data sharing, lack of structured systematic procedures and lack of data analysis transparency. The PLUME toolkit and its interoperable data format greatly contribute to resolving these issues, as it allows for consistent and accurate data collection, repeatable analysis, and easy sharing of experiments and results. In particular, it facilitates multi-site collaborations allowing for larger studies, as recommended by Carlier et al. [39] to increase VR studies reliability and statistical power.

Making implicit data accessible for everyone As raised in the introduction, collecting, and analyzing implicit data may require high-level software engineering resources, especially for visualizing or analyzing object trajectories, user gaze, and relevant heatmaps, or when it comes to synchronizing physiological and behavioral data, for joint analysis. This requirement impairs the utilization of these rich data, especially for academic teams outside the field of computer science. With Plume, which has been designed to be as easy as possible to use, we are committed to allowing any academic team, even without computer engineering resources or knowledge, to capture, review, and analyze implicit data from XR experiments.

4 Functionalities

We highlight here the main functionalities of PLUME (See Fig. 2), which consists of three tools: i) PLUME Recorder, a plug-and-play package for Unity to exhaustively record experiments, ii) PLUME Viewer, a standalone application to interactively replay and analyze the records, and iii) PLUME Python, a Python package to load records for external analysis and export to different data formats.

4.1 PLUME Recorder: a Plug-and-Play Unity Package

For our recorder, we targeted Unity3D game engine since it is a free, easy-to-use, and community-oriented graphics and physics engine to develop 2D, 3D and XR applications. It is widely used by researchers who need a VE for their experiments. Even though their main effort is usually on developing the VE, they also need *ad-hoc* recording tools to track users' behaviors for post-experiment analyses. To save them the time they spend developing logging tools and let them focus on their scientific objectives, we designed the PLUME Recorder to be plug-and-play, generic, lightweight and compatible with already existing projects. PLUME Recorder comes with a pre-configured module that can be dragged and dropped in a project to record with no extra configuration needed. It records the experiment as exhaustively as possible[2], including objects positions, appearances, user interfaces, perceived audio and user interactions. As experiments can run into difficulties, we integrated fail-safe mechanisms to the PLUME Recorder.

[2]Recorder modules currently implemented for: Transform, Mesh and Skinned Mesh Renderer, Audio, Canvas, Terrain, Light, Camera, Render Settings, Render Pipeline, Input Actions, LSL streams, custom markers, XRBaseInteractor + XRBaseInteractable (XRITK)

Physiological Signals Recording The PLUME Recorder offers the capability to synchronously record physiological signals with the virtual environment itself (eye tracking, heart rate, electrocardiogram, electroencephalogram etc.). In order to support a wide variety of devices, the PLUME Recorder integrates a receiver for applications that uses the Lab Streaming Layer (LSL) [21]. LSL is "an open-source networked middleware ecosystem to stream, receive, synchronize, and record neural, physiological, and behavioral data streams acquired from diverse sensor hardware". LSL has an already well-developed community and, as stated in their documentation, is compatible with "the majority of EEG systems on the market" and other biosignal hardware. By making PLUME able to record LSL streams, this further reduces the development time needed for researchers that seek to integrate synchronized physiological data in their recordings.

XR Interactions Recording XR experiments include specific interactions with the VE that are relevant to log for behavioral analyses. To integrate this kind of logging in the PLUME Recorder, we developed an extension to support the XR Interaction Toolkit (XRITK) [75], Unity's solution to create XR applications based on OpenXR, providing thus the largest compatibility. This pre-configured module can be added to the XR projects to log generic user interactions. These logs include the object used to interact, interacted objects, the interaction type (hover, select or activate) and status (begin/end). These interactions can later be visualized using the PLUME Viewer (See Section 4.4). The PLUME Recorder can also be customized to add the logging of application-specific interactions (See Section 4.5).

Broad Compatibility Our recorder is a versatile module, and while our main target is XR experiment recording, its design makes it also usable in most non-XR Unity projects, whether 2D or 3D. It supports different rendering pipelines (built-in, universal, high definition), and is compatible with applications built for Windows, Android and iOS. We have tested it on current Unity Long Term Support versions – 2021.3 and 2022.3 at the writing time. Although our recorder has been developed to be hardware agnostic, we have conducted specific testing for standalone HMDs to ensure compatibility. It has notably been tested with: HTC Vive Pro 2, HTC Vive Pro Eye, Varjo XR-3, Meta Quest 2 and Meta Quest Pro (PC VR and standalone mode for these last two).

4.2 Interoperable Record File

To the authors' knowledge on the current state of the literature, there is no open file format that is able to encapsulate the diversity, richness and complexity of VE experiments in an efficient manner. We propose our own file format based on the already established Protobuf format. We use Protobuf to serialize record samples in a language-neutral, platform-neutral manner, maximizing interoperability of the save file between languages and workflows. The data collected by the recorder is saved in a single file as the recording progresses. To save space, we compress the data on-the-fly using a dynamic compression algorithm and the gzip file format. This allows data compression as soon as it becomes available, unlike static compression that requires the record to be complete. As data are frequently flushed on disk and compression is dynamic, data loss would be limited in case of application failure. The record file can then be used to replay the experiment in PLUME Viewer (See Section 4.4) without sharing the whole Unity project, making it ideal to share exhaustive results of an experiment with peers.

4.3 PLUME Python: Load And Convert Records in Python

Our record files can easily be loaded in Python using the Protobuf python package due to their interoperability. To further improve workflows, we provide a python package *PLUME Python* that contains a set of utilities: load records, filter samples and convert the file to commonly used data formats. For instance, one can convert the samples to a pandas dataframe [47] for statistical analysis, export it to CSV, or implement their own exportation process. To maximize the integration in the LSL ecosystem, PLUME Python can also export the physiological signals as well as event markers to Extensible Data Format (XDF) to be loaded in SigViewer [9], EEGLAB [16] and MoBILAB [53].

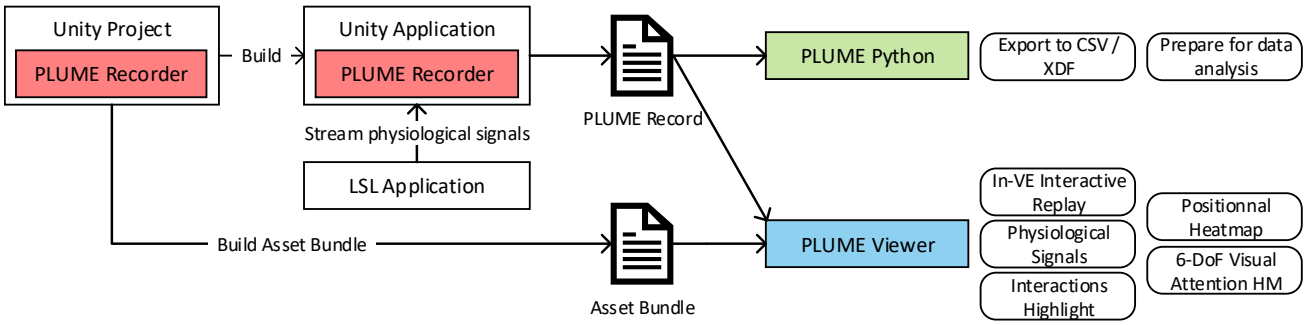


Fig. 2: Overview of the PLUME Toolbox

4.4 PLUME Viewer: a Standalone Viewer and Analyzer

Records can be replayed interactively in our separate and standalone application PLUME Viewer, fully decoupled with the original Unity project of an experiment. It facilitates the sharing of a record with peers, as it does not require any cumbersome installation of the project. To share a record, one simply needs to send the record file and the associated asset bundle[3] exported by the recorder (See Fig. 2). Given these files, the viewer can replay the experiment in an interactive manner to visualize behavioral data inside the VE. One can navigate inside the VE, play or pause the replay, or go to a specific timestamp. A major benefit of this viewer is the contextualisation of temporal data such as event markers or physiological signals within the spatial environment. The viewer also comes with several analysis modules described below.

3D Trajectory. For an object, positional data are loaded and used to create a polyline that represents the object 3D trajectory. This trajectory can be computed for the entire duration of the record or a specific time range. Multiple trajectories can be displayed at the same time.

Interaction Highlighting. This feature highlights objects that have been the most interacted with. This visualization can be computed for one or more interactors interacting with one or more interactables. While non-interactable objects are displayed in transparency, interactables with the fewest interactions are shown in white, and those with the most interactions in red.

6DoF Position Heatmap. For a designated object, positional data are loaded and orthogonally projected towards the ground (y-axis) using an adjustable Gaussian distribution. Positional heatmaps can be computed for the entire duration of the record or a specific time range.

6DoF Eye-gaze Heatmap. Positional gaze data, i.e., origin and direction, are loaded and projected from the user viewpoint onto selected objects in the VE using a Gaussian distribution that simulates eye gaze. Since gaze data are projected onto objects in the scene, the position of some dynamic ones will change throughout the session. To consider this when projecting data, the scene is replayed in fast-forward to respect the occlusions generated by these dynamic objects. Visual attention heatmaps can be computed for the entire record or a section. Implementation details are given in Section 5.4.

4.5 Extensibility

As every experiment is different, one might need to record custom event as markers, or more complex data. To meet the needs of as many researchers as possible, we designed PLUME with extensibility in mind. Custom data can be recorded by sending a custom Protobuf payload to the recording system in one line of code, from anywhere in the project. The sample is automatically timestamped, written to the record file, and accessible in our other tools (PLUME Python, PLUME Viewer).

5 Technical Highlights

This section describes the key technical points that allow PLUME to operate efficiently and accurately, from the recording mechanism and its clock system to the mechanism for replaying records, to optimizations making everything more performant. Finally, the last part is dedicated to the algorithm we introduced for 6DoF heatmaps.

5.1 Recording the Virtual Environment

5.1.1 Event-based vs State-Based Recording

When it comes to creating a recording engine, two approaches exist: event-based and state-based recording.

The event-based method relies on capturing any event that may result in a change in the VE e.g., by recording the input sent to the game. This method requires the engine to be deterministic so that replaying the events lead to the same result in the VE for every simulation. It has the benefit of being very lightweight, i.e., with a frugal quantity of data collected. However, event-based recording comes with several downsides: i) the VE state at any time is implicitly defined by the inputs, making harder the analysis of a record without fully re-simulating the VE, and strongly coupled with the game engine used for recording; ii) Unity physics engine is not natively deterministic, so re-simulating a record twice might not result in the same behavior, leading to authenticity issues. For these reasons, we decided to avoid using this method.

The state-based method relies on capturing the changes in the state of the virtual environment, i.e., when objects move, rotate, change their appearance, etc. Saving the state of the VE has the benefit of being explicit, meaning that one can access the state of the VE in the record at any time without having to re-simulate the whole VE. In opposition to the event-based system, data are decoupled with the game engine. This major benefit comes at the cost of a more intensive recording process and a higher volume of generated data. To mitigate these issues, we fragmented the description of the VE state into small samples. This allows for the recording of partial changes in the state of the VE only, i.e., when one property of an object changes, we do not record any other unchanged properties. Additionally, record files are compressed using Deflate to save disk space. Finally, we optimized the PLUME Recorder to be as fast as possible to minimize its impact on experiments. Performance considerations are described in Section 5.3.

5.1.2 Time Measurement

Recording the state of a VE requires an adequate timing system so that the temporal context remains as authentic as possible. Two components are important: i) the choice of the clock to add timestamps to samples, and ii) the synchronization of the clocks between PLUME Recorder and external systems, eg., Lab Streaming Layer for physiological signals.

Each recorded sample is associated with a timestamp expressed in nanoseconds since the beginning of the record on a 64-bits unsigned integer. The timestamp is automatically added when the sample is sent to the recording system using a high resolution Stopwatch clock [48]. Time measurement accuracy depends on the frequency of this clock. During our experimentations with PLUME, the Stopwatch implementation on our machine was accurate within 100 nanoseconds (10^7 ticks/s).

PLUME Recorder integrates synchronized recording of physiological data with LSL. LSL samples timestamps are adjusted to be on the same time frame as PLUME's clock. Network latency between clients emitting on LSL streams and the PLUME Recorder receiving the data

[3] An asset bundle is a file that contains assets from a Unity project. It can easily be built during the development of the application since the PLUME Recorder package provides a build feature directly usable within Unity editor

is corrected using LSL time correction estimate. According to their documentation, the LSL clock has a "sub-millisecond accuracy on a local network of computers", to analyze fast changes in signals such as brain activity. A major benefit and contribution resulting from this synchronization is the ability to contextualize the physiological signals within the spatial context of the VE.

5.1.3 Referencing Scene Objects And Assets

In a record, each object property update is associated with an object identifier. This one is crucial to keep the context of which object emitted which data and when. Unity provides at runtime a unique instance identifier (an integer) for each object in the scene. It is regenerated each time the application is run, making it unsuitable as a stable identifier for the same object across multiple records. As this could be useful for inter-record analysis, we implemented a stable globally unique identifier system (GUID) in our recorder. When building a project, a GUID registry stores a stable mapping between object references in the scenes and a unique GUID. When a new object is detected and does not have a GUID yet, a new one is automatically generated and inserted in the registry. In the same manner, we added a separate GUID registry specific to assets (meshes, materials, lightmaps...) that also stores the asset path so it can be found in the asset bundle exported by PLUME.

5.2 Replaying a Record

The exhaustivity of data contained in record files allows for a complete replay in our standalone PLUME Viewer. Samples are chronologically replayed with a mapping between the record object identifiers and instantiated replay objects. Assets such as meshes, materials, lightmaps, are loaded from the asset bundle using the recorded asset path. All of the assets required for the replay can be contained into the asset bundle, and no extra file apart from the record file and the asset bundle are needed to replay the record in PLUME Viewer.

5.3 Performance Considerations

5.3.1 Sample Pooling

When recording several dynamic objects in a scene, a lot of samples are instantiated each frame. Garbage collector operations, including allocation and release of the memory, have a direct impact on performance. For example, a transform update sample (position and rotation) contains approximately 300 bytes, mainly due to the size of the GUIDs. Allocating 300 bytes for each moving object at 50Hz can result in a lot of garbage collection operations. To reduce allocations number and memory release, the PLUME Recorder can reuse instance of samples for which the content has already been written to the file buffer. Sample instances are placed inside an object pool when available for reuse. When creating a new sample, one can get an old instance in the pool and fill it with new values instead of allocating a totally new instance.

5.3.2 Threaded Sample Packing and Writing

To provide extensibility as defined in Section 4.5, each sample contains a payload packed in the Protobuf's *Any* type. The packing process is executed in a separate thread to reduce its impact. When recording a sample, the user sends the payload. It is then wrapped in an unpacked sample marked with its timestamp, and added to a thread-safe queue. The packing thread consumes the unpacked sample, creates a sample with the payload packed inside the *Any* type, and writes it to the disk.

5.4 GPU Accelerated 6DoF Heatmap

As previously seen in Section 2.2, to our knowledge, there is no algorithm for 6DoF heatmap generation that supports dynamic scenes, is agnostic to meshes UV mapping, and is able to go beyond mesh resolution. We propose a new method to compute and render these maps that overcomes these limitations. Our representation to store the values is inspired by Yuksel et al. [81] *Mesh Colors*, consisting of a sub-sampling of each triangle of the mesh. Our method can generate 6DoF heatmaps in real-time from the recorded data with adjustable parameters, and renders it in real-time as well. We use this heatmap generation process in our PLUME Viewer to project and analyze the eye-gazes and objects positions in the virtual environment.

5.4.1 Mesh sampling

We resolve issues related to UV mapping and mesh resolution by storing heatmap values on points sampled on the surface. We adapted the Yuksel et al. [81] representation that uses an equally subdivided barycentric space to store mesh color data. The original Mesh Colors algorithm proposes a way to reference samples by their barycentric coordinates in the triangle. To better fit our need, we implemented a sample indexing in $O(1)$ complexity. This way, GPU kernels used for accelerating the projection algorithm can directly reference samples using this index. We adapt the number of samples in triangles depending on their surface area to achieve an acceptable quasi-uniform distribution over all the mesh. While not perfectly uniform, our representation allows for a $O(1)$ indexing of samples and a good compromise between uniformity and computational cost. To reduce memory usage, each sample stores a single float instead of a RGBA value, and colorization of the heatmap is done in the shader used for rendering.

Mesh triangles are rarely uniform in terms of scale across a model, e.g., it may have a very large amount of small triangles in high detailed areas, and fewer but larger triangles in low detailed sections. Thus, applying the same resolution r to all the triangles of the mesh would result in an unbalanced distribution of samples. We propose to adapt the sampling for each triangle based on their surface area to obtain a globally quasi-uniform distribution (See adaptive sampling resolution example in supplementary materials). Let k be a parameter corresponding to the number of sample points per squared meter we want to achieve, $\mathscr{A}$ the area of the triangle and r its resolution. We want to compute r so that number of samples $\#p$, computed using the n-th triangle formula, satisfies the following equation $\frac{\#p}{\mathscr{A}} = k$. We constrain the resolution r to be greater or equal to 1, ensuring that each triangle has at least its three vertices storing a value. Finally r can be expressed as:

$$\begin{cases} r \geq 1 \\ \#p = \frac{(r+1)(r+2)}{2} \\ \frac{\#p}{\mathscr{A}} = k \end{cases} \Rightarrow r = \begin{cases} 1 & 1 + 8k\mathscr{A} < 25 \\ \dfrac{-3 + \sqrt{1 + 8k\mathscr{A}}}{2} & 1 + 8k\mathscr{A} \geq 25 \end{cases}$$

The adaptive resolution results in good quasi-uniform distribution of sample points on different size triangles. The triangle subdivision we use ensures the samples are evenly spaced across an equilateral triangle surface. Triangles are rarely equilaterals in practice in a model, but this representation offers a cheap way of indexing samples in a $O(1)$ complexity, as demonstrated above, and is good enough to achieve an overall quasi-uniform distribution of samples on the mesh. Additionally, when a mesh resolution becomes too high (with very small triangles) the sampling can be coupled to the mesh resolution again if the sample density k is not high enough. We chose a default value of $k = 1000$ samples per squared meter which fits most of our use cases.

5.4.2 Gaussian projection

Heatmap generation is composed of several steps depicted in Fig. 3. We start by accumulating projected Gaussian distributions corresponding to eye gazes or projected positions. The projection of the Gaussian is done through a virtual camera with a custom projection matrix, where approximately 100% of the Gaussian non-zero values are included in the resulting view frustum. We chose to include values corresponding to 4 standard deviations (includes approximately 99.99% of all values). In the case of eye gaze, the projection matrix is a perspective matrix, modelling the eye-gaze dispersion as a cone. The field of view of this perspective matrix is set to match the fovea angle of the human eye – we considered approximately 2.5° from the optical axis of an eye. In the case of user positions, an orthographic matrix is used to project them onto selected surfaces (in most cases the VE floor), with a diameter corresponding approximately to the average shoulder width (0.5 meters considered here). These parameters can be adjusted in PLUME Viewer.

In both cases, the projection step takes into account any occlusion that could occur. Gaussian values computation should only be performed on visible samples. The easiest and fastest way to perform an occlusion check is to use the z-buffer of our virtual camera, as used

by previous works [46, 55]. This buffer stores the minimal depth of 3D points projected on the screen. When two 3D points are projected onto the same buffer pixel, the shortest depth is kept. To know if a sample is visible, we compare its depth with the closest depth stored in the z-buffer. If the values are close enough ($\pm\varepsilon$), the sample is considered visible from the virtual camera. Using a z-buffer has the benefit of being compatible with any model, without requiring colliders and casting rays into the VE to check occlusions.

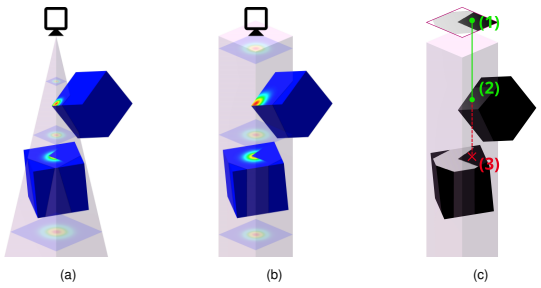


Fig. 3: Gaussian projection into 3D space to build heatmaps (a) perspective projection for eye-gaze mapping (b) orthographic projection for position mapping (c) z-buffer for occlusion check. Sample (2) is closer to the camera than occluded sample (3). (1) is the smallest depth value projected in screen space, stored in the z-buffer, here it equals (2). As (3) does not equal (1) ($\pm\varepsilon$), it does not receive the Gaussian projection.

A final pass normalizes the values after aggregation. Normalization is useful for filtering or applying transformations on the values later on, e.g., to tweak the render of the heatmap by changing color scale to enhance contrast. To normalize values, we keep track of the maximum value attributed to the sample points during aggregation.

5.4.3 Optimizations

Our algorithm is easily parallelizable on a GPU due to our samples indexing system and because Gaussian values can be independently computed for each sample. We use one GPU kernel to accumulate values, and one to normalize them and run it on all samples on a mesh. To avoid another pass through all samples to find the maximum value for normalization, the maximum value is saved inside the accumulation pass. The latter is subject to a race condition as each thread can update the maximum value at any time. We ensure that no race condition can occur in this critical code section using a mutex inside the accumulation kernel, locking the writing operations on the maximum value.

When accumulating values for a fixation, the naive approach is to run the kernel for all samples contained in the meshes of the VE, which is greedy, does not scale well, and is not applicable for complex VEs. To reduce the samples number where the projection is computed, we discard those not visible by the virtual camera, as their value are outside of 4σ of the Gaussian i.e., very close to 0.

5.4.4 Export

Position and eye-gaze heatmaps are exportable as point clouds. Each object in the scene is exported in a separated file. Samples from each objects are exported as points with their 3D position expressed in model space and a scalar field containing the aggregated value. This makes results available for external numerical analysis and visualization tools (eg. CloudCompare [22], MeshLab [13], Point Cloud Library [63]).

6 Performances Evaluation

Considering that the PLUME Recorder is executed at the same time as the XR application, it may have an impact on in-XR performance. To limit potential negative effects on quality of experience (QoE) and prevent cybersickness, this performance impact must be as low as possible since XR applications must run at high frame rates (between 60 and 120 fps, depending on the applications and HMDs) [78]. In this regard, we designed and optimized PLUME Recorder to be as fast as possible (see Sections 5.1 and 5.3). We designed a benchmark to evaluate the impact of PLUME Recorder on performance, knowing that this impact is directly related to the number of objects (GameObjects in Unity), since they are all tracked with their motion. Our benchmark is as follows: we created three artificial scenes with respectively 300, 600, and 900 objects (simple cubes, since the object complexity has no impact on PLUME performance). For each of these configurations, we derived three sub-configurations for which respectively 0%, 25% and 50% objects are moving. For all these configurations, we measured the mean (and the 2.5^{th} and 97.5^{th} percentiles) increase in frame time due to our recorder. This benchmark was run on a PC with up-to-date graphics drivers, and these specifications: Windows 10; Intel Core i9-10900K, 20 core @3.7GHz; 32 Go RAM; NVIDIA GeForce RTX 3070, 8Go VRAM. We used a simple screen display (not VR) since this does not affect PLUME performance. Table 2 provides the results. For the most tedious configuration (900 objects, of which 50% are moving) the performance loss is 2.2ms on average, which remains quite reasonable – 900 objects of which 450 are moving represent a realistic upper limit of standard VR experiments. These performance measures are valid for PC-VR experiments; however, autonomous HMDs have less CPU capability thus the impact may be higher. Accurate performance measures are difficult to run on HMDs since frame rates are usually capped at some predefined values. Still, we measured the loss in performance due to PLUME Recorder for the VR experiment conducted on a Meta Quest Pro and presented in section 7 (the scene contains ~600 objects, of which ~20% are moving). The measured average loss in performance is 1.04ms (67 fps with PLUME, 72 fps without).

VE objects count	300			600			900		
Moving object %	0	25	50	0	25	50	0	25	50
Mean (ms)	0.08	0.31	0.68	0.19	0.59	1.66	0.25	0.87	2.16
2.5^{th} percentile	0.00	0.12	0.50	0.06	0.36	1.32	0.10	0.60	1.76
97.5^{th} percentile	0.20	0.56	0.78	0.30	1.24	2.78	0.42	1.64	3.30

Table 2: Frame time increase due to PLUME Recorder, various VEs

7 Case Study

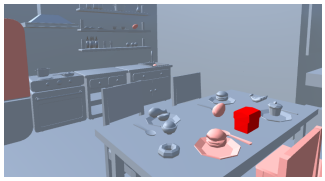
To demonstrate the potential of PLUME in helping researchers analyze XR experiments, we developed an experimental protocol based on a searching task: to find as many Easter eggs as possible in 3 minutes. This takes place in a 6DoF highly interactable VE: a house with a living room, a kitchen, two bedrooms and a bathroom, all linked by a corridor.

For the VR system we used a Meta Quest Pro in standalone mode coupled with Touch Pro controllers. In addition, we used a cardiac chest strap (Polar H9) to measure user heart rate. This device sent its data via Bluetooth and the computer streams them on a local network using LSL. Since the HMD was also connected to the local network via Wi-Fi, it received the streamed physiological data, which allowed PLUME to record them synchronously with the VR session (See a diagram of this setup in supplementary materials). Users were free to move in a *2.5m * 4m* area and to teleport in VR, granting them 6DoF. PLUME Recorder was integrated to the scene to record sessions as described in Section 4. Custom event markers were created to be recorded when the user finds an egg and enters a room; another marker recorded the total number of eggs retrieved. The PLUME record was saved on the Meta Quest Pro and could be reviewed later using PLUME Viewer on Windows. Using PLUME Viewer, anyone can replay the recordings in real time, to review users' actions and create visualizations as illustrated in Figures 1 and 5. In particular, the custom event markers can easily be connected with the physiological signal and the user behavior (e.g., the user trajectory). Recorded data can also be loaded into a conventional statistical analysis process using PLUME Python to produce relevant analysis. Figure 5 shows: a) a graph displaying the number of collected eggs over time for two users, and b) a table with the following statistics: total number of collected eggs, number of grabbed objects, the most hovered object, total distance covered, and number of teleportations.

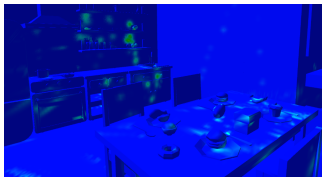
8 Ethics

By design PLUME records data anonymously, but depending on how it is used, personal data may be linked to the recordings. We would

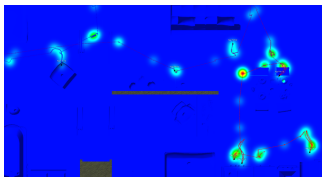
(a) 3D Trajectory + Event markers



(b) Hovered objects highlight

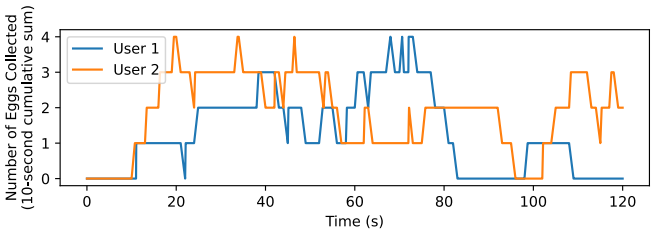


(c) Visual attention HM



(d) Position HM + Trajectory

Fig. 4: PLUME Viewer visualization examples



(a) Collected eggs over time: 2-users comparison.

	Eggs	Grabbed	Most Hovered	Traveled (m)	Teleportation
User 1	23	15	Refrigerator	123.3	67
User 2	15	42	Wine Bottle	88.90	42

(b) Various behavior statistics: 2-users comparison

Fig. 5: Computed analytics with PLUME Python

therefore warn researchers that wish to use the tool that they need to be careful when recording experimental data, especially when XR session data may be linked to personal data. Actually, recorded data such as positional data from the headset, controllers or trackers can be used to accurately re-identify yet anonymized data, e.g., through gait motion recognition [38]. Eye-gaze [2] and other bio-signals also provide ways of re-identification. Therefore, the use of PLUME with XR applications must be discussed with local ethical committees and data protection officers as the type of data collected with PLUME may lead to experiment participants' anonymity break. When using PLUME for public datasets creation, some solutions could be inspired from David-John et al. [15] work, who presented mechanisms to effectively reduce re-identification rate while maintaining high usability when training machine learning models. Despite these potential risks (leakage of personal data, re-identification, links with sensitive data), we believe that PLUME will positively contribute to academic research, and to the growth and accessibility of human-centered experimentation in XR.

9 Conclusion

We presented PLUME, a collection of tools to record, visualize, and analyze experiments conducted in 6DoF virtual environments. PLUME is aimed at researchers seeking to use behavioral and physiological data from Unity XR applications. PLUME Recorder for Unity is a lightweight, efficient, plug-and-play and multi-platform recorder that can be integrated at any development stage. To review and analyze records, we propose a standalone viewer, enabling interactive replays, synchronized with physiological signals and visualizations (trajectories, interactions highlights, position and eye-gaze heatmaps). Records can be loaded into traditional statistical workflows too, using PLUME Python for post-hoc analyses. To promote its adoption, PLUME is specifically designed to be as easy as possible to use, even for in-the-wild experiments with autonomous HMDs. In addition, PLUME is open-source and available on GitHub for the community to use and contribute to.

Limitations Much of the remaining technical challenges for PLUME development reside in improving performances even further. This is especially true for the viewer application that is yet to receive as much attention to its performances as the recorder did, particularly to handle large recordings. Besides, analysis modules are currently limited to one single-user record, which can be a major limitation for some use-cases, notably i) repeated single-user experiment, and ii) multi-user experiment. For the former, it would require adapting the existing modules to run aggregated data analysis such as navigation comparison across conditions. Regarding multi-user cases, their recording can already be done with PLUME on every instance of the application, creating as many records as there are users. This solution has two major advantages: the actual experience of each participant is recorded, even preserving latency and network issues; each record contains the data for the entire experiment – e.g. other users' positions, state of the VE etc. from the viewpoint of one user, and is therefore self-sufficient. However, this raises problems when aggregating and visualizing data, as the VE state is duplicated in every record but might not be synchronized. Enabling multi-users recording synchronization would require specific development. Coupled with aggregated data analysis, this would cater to the needs of the emerging social XR use-cases.

Perspectives We are actively working on resolving the aforementioned limitations. We anticipated the creation of analysis modules for aggregated records of the same experiment through the implementation of unique identification of virtual objects (See Section 5.1.3). Moreover, some XR use-cases could require external data feeds (e.g. external webcam, integrated MR passthrough video, microphone, etc.) to be synchronously recorded and replayed. Although not optimal, their capture is already feasible using LSL integration in PLUME. Specific modules would be necessary to replay the data in the viewer. Finally, the use of Protobuf as our sample serializer could facilitate monitoring the experiment in real-time from a remote instance of PLUME-Viewer. This requires improving serialization performance and sending samples over the network. Remote monitoring would be useful for crowdsourced experiments or for live analyses from a distant machine.

We are looking forward to new collaborations with the community to improve PLUME. Apart from these technical challenges, our tool must convince the scientific community of its relevance and genuine utility. We plan to promote PLUME in showcase demos, workshops and tend towards a systematic use in our XR studies. PLUME is part of an open science approach where collected data can be shared and aggregated into standardized datasets for meta-analysis and prediction models. We hope that it would facilitate the creation of new scientific contributions based on collected 6DoF XR data and help XR experiments replicability, repeatability, and reliability.

Acknowledgments

This work was supported by Auvergne-Rhône-Alpes region as part of the PROMESS project and by the French National Research Agency as part of the RENFORCE project (ANR-22-CE31-0023-03).

References

[1] J. Annett. Hierarchical task analysis. *Handbook of cognitive task design*, 2:17–35, 2003. 3

[2] S. M. Asish, A. K. Kulshreshth, and C. W. Borst. User identification utilizing minimal eye-gaze features in virtual reality applications. *Virtual Worlds*, 1(1):42–61, 2022. doi: 10.3390/virtualworlds1010004 9

[3] D. E. Beaton, C. Bombardier, F. Guillemin, and M. Ferraz. Guidelines for the Process of Cross-Cultural Adaptation of Self-Report Measures. https://journals.lww.com/spinejournal/citation/2000/12150/guidelines_for_the_process_of_cross_cultural.14.aspx. Accessed: 2023-12-19. 2

[4] A. Bellazzi, L. Bellia, G. Chinazzo, F. Corbisiero, P. D'Agostino, A. Devitofrancesco, F. Fragliasso, M. Ghellere, V. Megale, and F. Salamone. Virtual reality for assessing visual quality and lighting perception: A systematic review. *Building and Environment*, 209:108674, 2022. 2

[5] F. Berton, F. Grzeskowiak, A. Bonneau, A. Jovane, M. Aggravi, L. Hoyet, A.-H. Olivier, C. Pacchierotti, and J. Pettre. Crowd navigation in vr: exploring haptic rendering of collisions. *IEEE Transactions on Visualization and Computer Graphics*, 28(7):2589–2601, 2020. 2

[6] D. A. Bowman and L. F. Hodges. Formalizing the design, evaluation, and application of interaction techniques for immersive virtual environments. *Journal of Visual Languages & Computing*, 10(1):37–53, 1999. 3

[7] J. Brooke. Sus: A quick and dirty usability scale. *Usability Eval. Ind.*, 189, 11 1995. 2

[8] J. Brookes, M. Warburton, M. Alghadier, M. Mon-Williams, and F. Mushtaq. Studying human behavior with virtual reality: The Unity Experiment Framework. *Behavior Research Methods*, 52(2):455–463, Apr. 2020. doi: 10.3758/s13428-019-01242-0 3, 4

[9] C. Brunner. SigViewer repository. https://github.com/cbrnr/sigviewer. Accessed: 2023-12-19. 5

[10] W. Büschel, A. Lehmann, and R. Dachselt. Miria: A mixed reality toolkit for the in-situ visualization and analysis of spatio-temporal interaction data. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, pp. 1–15, 2021. 4

[11] O. Bălan, G. Moise, A. Moldoveanu, M. Leordeanu, and F. Moldoveanu. An Investigation of Various Machine and Deep Learning Techniques Applied in Automatic Fear Level Detection and Acrophobia Virtual Therapy. *Sensors*, 20(2):496, Jan. 2020. Number: 2 Publisher: Multidisciplinary Digital Publishing Institute. doi: 10.3390/s20020496 3

[12] B. C. Choi and A. W. Pak. A Catalog of Biases in Questionnaires. *Preventing Chronic Disease*, 2(1):A13, Dec. 2004. 2

[13] P. Cignoni, A. Muntoni, and colleagues. MeshLab website. https://www.meshlab.net/. Accessed: 2023-12-19. 8

[14] Cognitive3D. Collect and measure spatial data to bring visibility to user participation, and optimize simulations for success. https://cognitive3d.com/. Accessed: 2023-12-19. 4

[15] B. David-John, K. Butler, and E. Jain. Privacy-preserving datasets of eye-tracking samples with applications in XR. *IEEE Transactions on Visualization and Computer Graphics*, 29(5):2774–2784, May 2023. Conference Name: IEEE Transactions on Visualization and Computer Graphics. doi: 10.1109/TVCG.2023.3247048 9

[16] A. Delorme and S. Makeig. Eeglab: an open source toolbox for analysis of single-trial eeg dynamics including independent component analysis. *Journal of Neuroscience Methods*, 134(1):9–21, Mar 2004. doi: 10.1016/j.jneumeth.2003.10.009 5

[17] J. Deuchler, W. Hettmann, D. Hepperle, and M. Wölfel. Streamlining physiological observations in immersive virtual reality studies with the virtual reality scientific toolkit. In *2023 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW)*, pp. 485–488. IEEE, 2023. 4

[18] X. Ding and Z. Chen. Towards mesh saliency detection in 6 degrees of freedom. *arXiv:2005.13127 [cs, eess]*, Jun 2020. arXiv: 2005.13127. 3

[19] E. Fincham, A. Whitelock-Wainwright, V. Kovanović, S. Joksimović, J.-P. van Staalduinen, and D. Gašević. Counting Clicks is Not Enough: Validating a Theorized Model of Engagement in Learning Analytics. In *Proceedings of the 9th International Conference on Learning Analytics & Knowledge*, LAK19, pp. 501–510. Association for Computing Machinery, New York, NY, USA, Mar. 2019. doi: 10.1145/3303772.3303775 3

[20] S. C. for Computational Neuroscience. A integration approach of the LabStreamingLayer framework for Unity3D. https://github.com/labstreaminglayer/LSL4Unity. Accessed: 2023-12-19. 3, 4

[21] S. C. for Computational Neuroscience. LabStreamingLayer super repository comprising submodules for LSL and associated apps. https://github.com/sccn/labstreaminglayer. Accessed: 2023-12-19. 2, 5

[22] D. Girardeau-Montaut. CloudCompare: 3D point cloud and mesh processing software. https://www.danielgm.net/cc/. Accessed: 2023-12-19. 8

[23] G. Gorisse, O. Christmann, and C. Dubosc. REC: A Unity Tool to Replay, Export and Capture Tracked Movements for 3D and Virtual Reality Applications. In *Proceedings of the 2022 International Conference on Advanced Visual Interfaces*, AVI 2022, pp. 1–3. Association for Computing Machinery, New York, NY, USA, June 2022. doi: 10.1145/3531073.3534472 3, 4

[24] I. Goudé, A. Bruckert, A.-H. Olivier, J. Pettré, R. Cozot, K. Bouatouch, M. Christie, and L. Hoyet. Real-time multi-map saliency-driven gaze behavior for non-conversational characters. *IEEE Transactions on Visualization and Computer Graphics*, 2023. 3

[25] S. Graf and V. Schwind. Inconsistencies of presence questionnaires in virtual reality. In *Proceedings of the 26th ACM Symposium on Virtual Reality Software and Technology*, pp. 1–3, 2020. 2

[26] Q. Guimard, F. Robert, C. Bauce, A. Ducreux, L. Sassatelli, H.-Y. Wu, M. Winckler, and A. Gros. PEM360: a dataset of 360° videos with continuous physiological measurements, subjective emotional ratings and motion traces. In *Proceedings of the 13th ACM Multimedia Systems Conference*, MMSys '22, pp. 252–258. Association for Computing Machinery, New York, NY, USA, Aug. 2022. doi: 10.1145/3524273.3532895 3

[27] A. Halbig and M. E. Latoschik. A Systematic Review of Physiological Measurements, Factors, Methods, and Applications in Virtual Reality. *Frontiers in Virtual Reality*, 2, 2021. 1, 2

[28] S. G. Hart and L. E. Staveland. Development of NASA-TLX (Task Load Index): Results of Empirical and Theoretical Research. In P. A. Hancock and N. Meshkati, eds., *Advances in Psychology*, vol. 52 of *Human Mental Workload*, pp. 139–183. North-Holland, Jan. 1988. doi: 10.1016/S0166-4115(08)62386-9 2

[29] D. Hepperle, T. Dienlin, and M. Wölfel. Reducing the human factor in virtual reality research to increase reproducibility and replicability. In *2021 IEEE International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct)*, pp. 100–105. IEEE, 2021. 4

[30] D. Hepperle, T. Dienlin, and M. Wölfel. Reducing the human factor in virtual reality research to increase reproducibility and replicability. In *2021 IEEE International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct)*, pp. 100–105, 2021. doi: 10.1109/ISMAR-Adjunct54149.2021.00030 5

[31] F. Homps, Y. Beugin, and R. Vuillemot. ReViVD: Exploration and Filtering of Trajectories in an Immersive Environment using 3D Shapes. In *2020 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, pp. 729–737, Mar. 2020. ISSN: 2642-5254. doi: 10.1109/VR46266.2020.00096 2

[32] Z. Hu, A. Bulling, S. Li, and G. Wang. Fixationnet: Forecasting eye fixations in task-oriented virtual environments. *IEEE Transactions on Visualization and Computer Graphics*, p. 1–1, 2021. doi: 10.1109/TVCG.2021.3067779 3

[33] S. Hubenschmid, J. Wieland, D. I. Fink, A. Batch, J. Zagermann, N. Elmqvist, and H. Reiterer. Relive: Bridging in-situ and ex-situ visual analytics for analyzing mixed reality user studies. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, pp. 1–20, 2022. 4

[34] A. Iop, V. G. El-Hajj, M. Gharios, A. de Giorgio, F. M. Monetti, E. Edström, A. Elmi-Terander, and M. Romero. Extended reality in neurosurgical education: A systematic review. *Sensors*, 22(16):6067, 2022. 2

[35] H. K. Kim, J. Park, Y. Choi, and M. Choe. Virtual reality sickness questionnaire (VRSQ): Motion sickness measurement index in a virtual reality environment. *Applied Ergonomics*, 69:66–73, May 2018. doi: 10.1016/j.apergo.2017.12.016 2

[36] Y. M. Kim, I. Rhiu, and M. H. Yun. A Systematic Review of a Virtual Reality System from the Perspective of User Experience. *International Journal of Human–Computer Interaction*, 36(10):893–910, June 2020. doi: 10.1080/10447318.2019.1699746 2, 3

[37] G. A. Koulieris, G. Drettakis, D. Cunningham, and K. Mania. An automated high-level saliency predictor for smart game balancing. *ACM Transactions on Applied Perception*, 11(4):1–21, Jan 2015. doi: 10.1145/2637479 3

[38] K. Kurita. Human identification from walking signal based on measurement of current generated by electrostatic induction. *Kansei Engineering International Journal*, 11(4):183–189, 2012. 9

[39] M. Lanier, T. F. Waddell, M. Elson, D. J. Tamul, J. D. Ivory, and A. Przybylski. Virtual reality check: Statistical power, reported results, and the validity of research on the psychology of virtual reality and immersive environments. *Computers in Human Behavior*, 100:70–78, 2019. doi: 10.1016/j.chb.2019.06.015 5

[40] E. Lavoué, Q. Ju, S. Hallifax, and A. Serna. Analyzing the relationships between learners' motivation and observable engaged behaviors in a gamified learning environment. *International Journal of Human-Computer Studies*, 154:102670, Oct. 2021. doi: 10.1016/j.ijhcs.2021.102670 3

[41] G. Lavoué, F. Cordier, H. Seo, and M.-C. Larabi. Visual attention for rendered 3d shapes. *Computer Graphics Forum*, 37(2):191–203, May 2018. doi: 10.1111/cgf.13353 3

[42] F. Lemic, J. Struye, and J. Famaey. Short-term trajectory prediction for full-immersive multiuser virtual reality with redirected walking. In

GLOBECOM 2022-2022 IEEE Global Communications Conference, pp. 6139–6145. IEEE, 2022. 2

[43] C. Maranes, D. Gutierrez, and A. Serrano. Exploring the impact of 360° movie cuts in users' attention. *Proceedings - 2020 IEEE Conference on Virtual Reality and 3D User Interfaces, VR 2020*, p. 73–82, 2020. doi: 10.1109/VR46266.2020.1580727911717 3

[44] D. Martin, A. Serrano, A. W. Bergman, G. Wetzstein, and B. Masia. Scangan360: A generative model of realistic scanpaths for 360 images. *IEEE Transactions on Visualization and Computer Graphics*, 28(5):2003–2013, 2022. 3

[45] S. L. Matthews, A. Uribe-Quevedo, and A. Theodorou. Rendering optimizations for virtual reality using eye-tracking. In *2020 22nd symposium on virtual and augmented reality (SVR)*, pp. 398–405. IEEE, 2020. 3

[46] M. Maurus, J. H. Hammer, and J. Beyerer. Realistic heatmap visualization for interactive analysis of 3d gaze data. In *Proceedings of the Symposium on Eye Tracking Research and Applications*, p. 295–298. ACM, Safety Harbor Florida, Mar 2014. doi: 10.1145/2578153.2578204 3, 8

[47] W. McKinney. Pandas Dataframe documentation. https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.html. Accessed: 2023-12-19. 5

[48] Microsoft. MSDN Stopwatch documentation. https://learn.microsoft.com/en-us/dotnet/api/system.diagnostics.stopwatch. Accessed: 2023-12-19. 6

[49] D. Monteiro, H.-N. Liang, X. Tang, and P. Irani. Using Trajectory Compression Rate to Predict Changes in Cybersickness in Virtual Reality Games. In *2021 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pp. 138–146. IEEE, Bari, Italy, Oct. 2021. doi: 10.1109/ISMAR52148.2021.00028 2

[50] A. Murgia, R. Wolff, W. Steptoe, P. Sharkey, D. Roberts, E. Guimaraes, A. Steed, and J. Rae. A tool for replay and analysis of gaze-enhanced multiparty sessions captured in immersive collaborative environments. In *2008 12th IEEE/ACM International Symposium on Distributed Simulation and Real-Time Applications*, pp. 252–258. IEEE, 2008. 3

[51] M. Nebeling, M. Speicher, X. Wang, S. Rajaram, B. D. Hall, Z. Xie, A. R. Raistrick, M. Aebersold, E. G. Happ, J. Wang, et al. Mrat: The mixed reality analytics toolkit. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, pp. 1–12, 2020. 3, 4

[52] Nvidia. NVIDIA VCR. https://docs.nvidia.com/vcr-sdk/overview/overview.html/. Accessed: 2024-01-03. 3, 4

[53] A. Ojeda, N. Bigdely-Shamlo, and S. Makeig. Mobilab: an open source toolbox for analysis and visualization of mobile brain/body imaging data. *Frontiers in Human Neuroscience*, 8, Mar 2014. doi: 10.3389/fnhum.2014.00121 5

[54] T. Pfeiffer. Measuring and visualizing attention in space with 3d attention volumes. In *Proceedings of the Symposium on Eye Tracking Research and Applications - ETRA '12*, p. 29. ACM Press, Santa Barbara, California, 2012. doi: 10.1145/2168556.2168560 3

[55] T. Pfeiffer and C. Memili. Model-based real-time visualization of realistic three-dimensional heat maps for mobile eye tracking and eye tracking in virtual reality. In *Proceedings of the Ninth Biennial ACM Symposium on Eye Tracking Research & Applications*, p. 95–102. ACM, Charleston South Carolina, Mar 2016. doi: 10.1145/2857491.2857541 3, 8

[56] P. Raimbaud. *Virtual reality for building industry needs: guiding the design of user interactions through a task-centred methodology*. PhD thesis, Arts-et-Metiers and Universidad de los Andes, 2020. 3

[57] P. Raimbaud, A. Jovane, K. Zibrek, C. Pacchierotti, M. Christie, L. Hoyet, J. Pettré, and A.-H. Olivier. The stare-in-the-crowd effect when navigating a crowd in virtual reality. In *ACM Symposium on Applied Perception 2023*, pp. 1–10, 2023. 2

[58] P. Reipschläger, F. Brudy, R. Dachselt, J. Matejka, G. Fitzmaurice, and F. Anderson. Avatar: An immersive analysis environment for human motion data combining interactive 3d avatars and trajectories. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, pp. 1–15, 2022. 4

[59] I. Rennes. Virtual Crowds! CrowdMP is a Unity project used as an experimentation platform. https://project.inria.fr/crowdscience/project/ocsr/crowdmp/. Accessed: 2023-12-19. 3, 4

[60] F. A. S. Robert, H.-Y. Wu, L. Sassatelli, S. Ramanoel, A. Gros, and M. Winckler. An Integrated Framework for Understanding Multimodal Embodied Experiences in Interactive Virtual Reality. In *Proceedings of the 2023 ACM International Conference on Interactive Media Experiences*, June 2023. doi: 10.1145/3573381.3596150 3

[61] S. Rossi, I. Viola, L. Toni, and P. Cesar. Extending 3-DoF Metrics to Model User Behaviour Similarity in 6-DoF Immersive Applications. In *Proceedings of the 14th Conference on ACM Multimedia Systems*, MMSys '23, pp. 39–50. Association for Computing Machinery, New York, NY, USA, June 2023. doi: 10.1145/3587819.3590976 2

[62] D. Roth and M. E. Latoschik. Construction of the Virtual Embodiment Questionnaire (VEQ). *IEEE Transactions on Visualization and Computer Graphics*, 26(12):3546–3556, Dec. 2020. Conference Name: IEEE Transactions on Visualization and Computer Graphics. doi: 10.1109/TVCG.2020.3023603 2

[63] R. B. Rusu and S. Cousins. Point Cloud Library website. https://pointclouds.org/. Accessed: 2023-12-19. 8

[64] V. Sitzmann, A. Serrano, A. Pavel, M. Agrawala, D. Gutierrez, B. Masia, and G. Wetzstein. How do people explore virtual environments? *arXiv:1612.04335 [cs]*, Sep 2017. arXiv: 1612.04335. 3

[65] M. Slater, M. Usoh, and A. Steed. Depth of Presence in Virtual Environments. *Presence: Teleoperators and Virtual Environments*, 3(2):130–144, May 1994. doi: 10.1162/pres.1994.3.2.130 2

[66] A. Steed, L. Izzouzi, K. Brandstätter, S. Friston, B. Congdon, O. Olkkonen, D. Giunchi, N. Numan, and D. Swapp. Ubiq-exp: A toolkit to build and run remote and distributed mixed reality experiments. *Frontiers in Virtual Reality*, 3, 2022. 3, 4

[67] S. Stellmach, L. Nacke, and R. Dachselt. Advanced gaze visualizations for three-dimensional virtual environments. In *Proceedings of the 2010 Symposium on Eye-Tracking Research & Applications - ETRA '10*, p. 109. ACM Press, Austin, Texas, 2010. doi: 10.1145/1743666.1743693 3

[68] V. Sundstedt and V. Garro. A Systematic Review of Visualization Techniques and Analysis Tools for Eye-Tracking in 3D Environments. *Frontiers in Neuroergonomics*, 3, 2022. 3

[69] B. Systems. Complete VR eye tracking system – 1 user. https://www.biopac.com/product/vr-eye-tracking-lab-1user/. Accessed: 2023-12-19. 4

[70] L. Tabbaa, R. Searle, S. M. Bafti, M. M. Hossain, J. Intarasisrisawat, M. Glancy, and C. S. Ang. VREED: Virtual Reality Emotion Recognition Dataset Using Eye Tracking & Physiological Measures. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 5(4):178:1–178:20, Dec. 2022. doi: 10.1145/3495002 3

[71] G. Tartarisco, N. Carbonaro, A. Tonacci, G. Bernava, A. Arnao, G. Crifaci, P. Cipresso, G. Riva, A. Gaggioli, D. De Rossi, A. Tognetti, and G. Pioggia. Neuro-Fuzzy Physiological Computing to Assess Stress Levels in Virtual Reality Therapy. *Interacting with Computers*, 27(5):521–533, Sept. 2015. doi: 10.1093/iwc/iwv010 3

[72] K. Tcha-Tokey, O. Christmann, E. Loup-Escande, and S. Richir. Proposition and Validation of a Questionnaire to Measure the User Experience in Immersive Virtual Environments. *International Journal of Virtual Reality*, 16(1):33–48, Jan. 2016. doi: 10.20870/IJVR.2016.16.1.2880 2

[73] Tobii. Tobii Ocumen software website. https://developer.tobii.com/xr/solutions/tobii-ocumen/. Accessed: 2024-01-03. 4

[74] C. Tremmel, C. Herff, T. Sato, K. Rechowicz, Y. Yamani, and D. J. Krusienski. Estimating Cognitive Workload in an Interactive Virtual Reality Environment Using EEG. *Frontiers in Human Neuroscience*, 13, 2019. 3

[75] Unity. XR Interaction Toolkit documentation. https://docs.unity3d.com/Packages/com.unity.xr.interaction.toolkit@2.4/manual/index.html. Accessed: 2023-12-19. 5

[76] Unity Technologies. Unity Real-Time Development Platform | 3D, 2D, VR & AR Engine. https://unity.com/. Accessed: 2023-12-19. 2

[77] S. Villenave, J. Cabezas, P. Baert, F. Dupont, and G. Lavoue. Xrecho: A unity plug-in to record and visualize user behavior during xr sessions. In *13th ACM Multimedia Systems Conference (MMSys 2022)*, Jun 2022. doi: 10.1145/3524273.3532909 4

[78] J. Wang, R. Shi, W. Zheng, W. Xie, D. Kao, and H.-N. Liang. Effect of frame rate on user experience, performance, and simulator sickness in virtual reality. *IEEE Transactions on Visualization and Computer Graphics*, 29(5):2478–2488, 2023. 8

[79] B. G. Witmer and M. J. Singer. Measuring Presence in Virtual Environments: A Presence Questionnaire. *Presence: Teleoperators and Virtual Environments*, 7(3):225–240, 1998. doi: 10.1162/105474698565686 2

[80] H.-Y. Wu, F. Robert, T. Fafet, B. Graulier, B. Passin-Cauneau, L. Sassatelli, and M. Winckler. Designing Guided User Tasks in VR Embodied Experiences. *Proceedings of the ACM on Human-Computer Interaction*, 6(EICS):158:1–158:24, June 2022. doi: 10.1145/3532208 3, 4

[81] C. Yuksel, J. Keyser, and D. H. House. Mesh colors. Technical report, Department of Computer Science, Texas A&M University, 2008. 7