Bauhaus-Universität Weimar
Faculty of Media
Degree Computer Science for Digital Media

Recording & Immersive Exploration of Virtual Reality Experiences

Master's Thesis

Anton Benjamin Lammert
b. 5th October 1997 in Weimar

Matriculation Number 118592

First Referee: Prof. Dr. Bernd Fröhlich
Second Referee: Prof. Dr. Eva Hornecker

Submission date: 06.10.2022

Declaration of Academic Honesty

I hereby confirm that I worked on this thesis with the title
Recording & Immersive Exploration of Virtual Reality Experiences
independently and that I did not use sources of any kind other than the ones that I have acknowledged and cited.

Furthermore, I confirm that this is the first time that this thesis - in any form - is presented to a supervising staff.

Weimar, 06.10.2022

ANTON BENJAMIN LAMMERT

Abstract

The ability to record and replay user interactions and changes in immersive single and multi-user virtual environments plays a crucial role in many different domains, including asynchronous collaboration and virtual analysis. The aim of this work is two-fold: Firstly, we design and develop a cross-platform plugin that enables efficient recording and playback of interactions and changes in virtual environments with several thousand simultaneously moving objects. Secondly, we investigate appropriate techniques to support the exploration of such immersive recordings in virtual reality that support the user's change awareness, reduce the fear of missing out, and allow for precise temporal navigation. However, there is only limited publicly available scientific research on such techniques for the exploration of immersive recordings in virtual reality. In this thesis, we propose novel techniques inspired by research on guidance in omnidirectional videos that aim to support the user's change awareness during temporal navigation while reducing the fear of missing out. Based on research towards precise temporal navigation for digital 2D and omnidirectional video, we propose a set of different techniques for precise temporal navigation of immersive recordings in virtual reality. In addition, we propose techniques designed to support the collaborative exploration of immersive recordings in multi-user virtual environments. We evaluate the usability of a system for the exploration of immersive recordings employing a set of the proposed techniques in a pilot study. Our results indicate that the proposed techniques to inform users of changes in the virtual environment during temporal navigation support the exploration of immersive recordings.

Table of Contents

1 Introduction

For the development of suitable interaction techniques for virtual environments it is important to be able to thoroughly analyse how effectively they can be utilised. The ability to record users interacting in a virtual environment can support such an analysis, as the recordings can be analysed to identify potential strengths and weaknesses of interaction techniques. While screen recordings can be used for such an analysis, they allow only for a limited analysis in the sense that the viewing perspective cannot be changed once the recording has been created, which prevents the analysis of interactions that were not visible on the screen. The ability to record and replay changes and interactions in virtual environments, in contrast, allows for inspecting the interactions in the recording from any viewing perspective during analysis and thus can allow for an exhaustive analysis. We refer to such recordings, which can be explored in immersive virtual reality when played back, as *immersive recordings*. Such immersive recordings are used in several different domains, including asynchronous collaboration [1,2], virtual observation and analysis [3–6], the creation of recorded characters for games, movies and educational content [7–10], development of mixed reality applications [11], entertainment [12,13] and interaction in virtual environments [14,15].

Since immersive recordings are used in a variety of different domains, we would like to briefly introduce how they are used in each of these domains. In asynchronous collaboration, immersive recordings of messages from users to collaborators can be created which are then made available to the collaborators for playback [2]. To study how users act and interact in virtual reality applications, virtual observation and analysis can involve the creation of immersive recordings, as those can allow for a more detailed study of user behaviour. Recordings of a part of the virtual environment, e.g. a character, have been used for 3D games, movies and educational content [7,10]. In software development, immersive recordings can help identify software bugs that result in visual changes in the virtual environment that could not be detected during runtime but that can be identified during playback of the recording [11]. In entertainment, immersive recordings can also be found, e.g. in the game *Fortnite*, where they can be explored by users to re-experience exciting moments of the game [12]. In addition, immersive recordings are tightly coupled to a variety of VR interaction techniques, especially if they provide references in time. One notable example for this is the *Photoportal* technique [15], which allows users to create a immersive recording and interact with it through the use of a *Photoportal*.

Spatio-Temporal Exploration of Immersive Recordings

In contrast to the exploration of static 3D scenes, exploring and understanding spatio-temporal recordings of virtual environments requires both *spatial* and *temporal* navigation. While spatial navigation in VR has been thoroughly investigated in the literature, there is only limited publicly available research on temporal navigation in VR.

The ability to track asynchronous changes within a workspace such as a virtual environment, also known as *change awareness* [16], is required to understand the process that resulted in the current state of the workspace. Because rapid temporal navigation can cause the user to miss relevant events, the user's change awareness can be impaired, causing him or her to become confused. One can see that in order to avoid this, techniques for improving change awareness during temporal navigation are required, which have not, to the best of our knowledge, been studied in the context of immersive recordings so far.

Investigating the exploration of immersive recordings in a multi-user context, we found that there are further research questions that have not been investigated in the literature. Those research questions focus on (1) *how several users can collaborate across time and space*, (2) *how to best navigate to a user which is located at a different point in time* and (3) *how to best inform the user of the temporal positions and navigation intentions of other users*.

In this thesis, we explore, implement and evaluate temporal navigation techniques for immersive recordings, propose techniques to improve change awareness during temporal navigation and investigate simple techniques to support collaborative exploration of immersive recordings. In order to be able to work with immersive recordings on different devices and to investigate techniques for temporal navigation, we introduce a device-independent recording plugin for virtual environments that we integrate into Unity, a game engine that is widely used in the VR community and enables us to share our experiences with a larger audience.

In the following, we will shortly summarise the content of the remaining chapters of this thesis.
Chapter 2 provides an overview of scientific articles which are relevant for this thesis, which deal with immersive recording tools and temporal navigation techniques for 2D, omnidirectional and immersive recordings.
In **Chapter 3** the reader is introduced to the immersive recording plugin designed and developed throughout this thesis, as well as its integration into the Unity game engine.
Chapter 4 analyses different techniques for temporal navigation and exploration of immersive recordings in more detail. Based on this analysis, novel techniques for exploration and temporal navigation of immersive recordings are proposed.
In **Chapter 5** the usability of a set of techniques proposed in the previous chapter is evaluated through a pilot user study.
Chapter 6 summarises the results of this thesis and reflects on the limitations of different techniques. In addition, different techniques for future work are proposed.

2 Related Work

Developing a tool for recording states of a virtual environment is a non-trivial task, as the tool must be able to record and play back potentially significantly large amounts of data efficiently and without stalling. In order to make meaningful design decisions, we analyse existing recording tools from the related literature. To efficiently explore recordings of unknown content and arbitrary length, efficient temporal navigation techniques are required. There has been plenty of research on temporal navigation in the context of 2D and omnidirectional recordings, therefore we analyse existing efficient temporal navigation techniques for 2D and omnidirectional recordings in order to make reasonable decisions for techniques that allow for temporal navigation of immersive recordings in virtual reality.

2.1 Recording of Virtual Environments

To record changes in virtual environments Luttermann et al. [17] propose the *VRML History* scripting language that enables storage and use of spatio-temporal virtual worlds. For recording, a so-called *temporal scene graph* is used that extends the standard scene graph by storing the time intervals during which nodes change. Luttermann et al. discuss reducing the amount of recorded data by detecting data that does not have to be recorded because it can be derived from previous and subsequent recorded data through linear interpolation. By reducing the number of recorded elements, performance during replay of the recording can be improved as less data has to be loaded.

The concept of *temporal links* for immersive recordings was introduced by Greenhalgh et al. [18]. Temporal links essentially relate recorded data to *locales* which represent local coordinate systems similar to scene graph nodes. The immersive recording is created by processing the sequence of all events applied to each locale and writing the events together with a timestamp out to a file. In addition to recording changes in the virtual environment the system of Greenhalgh et al. allows for the recording of audio streams. Since only events, such as state changes are recorded, states are not necessarily recorded for every object and every frame. For event-driven motion, such as a train travelling at a constant speed, only the start and end events may be recorded based on this system.

In studying multimodal data capture in virtual environments, Steptoe and Steed [19] note that it is important to capture all data sources to enable qualitative analysis of the recording. As an example, they mention that, if the user's voice was not captured, a user investigating the recording may not be able to understand why a user behaved in a certain way. To allow for later processing and analysis of the data, the authors suggest that the output files should be in human readable file format.

Fominykh et al. [8] record transformations, text, and voice chat messages, as well as the content of virtual whiteboards for the educational platform *vAcademia*. Their system allows multiple remote users to experience a recording together and provides users with basic editing capabilities. To reduce loading time during random access of the recording checkpoints are created at specific intervals.

Watson et al. [20] propose a hardware and software suite called *Unified Suite for Experiments* (USE) to reliably record and evaluate psychophysical experiments. Their suite is developed for the Unity game engine and uses an Arduino to synchronise multiple data streams. During recording, position, scale, rotation, and additional properties of interest of each object in the scene are recorded. The tool from Watson et al. is also capable of recording the user's gaze position and visualising it during replay. While the recording tool is publicly available, additional hardware including an Arduino is required to utilise it as designed.

Steed et al. [11] outline simple methods for recording and playing back changes in virtual environments using the Unity game engine. They suggest adjusting the recording and playback rates to match the screen refresh rate, as boolean values cannot be meaningfully interpolated during playback when the replay rate is higher than the recording rate and resulting game states may result in visual artefacts. Although the code associated with the publication is publicly available, it is not optimised for efficient recording and playback of scenes with a large number of objects and is for this reason not well suited for the investigation of temporal navigation techniques in which fast temporal navigation can occur.

Nebeling et al. [6] identified a lack of tools for analysing mixed reality experiences and developed the *Mixed Reality Analytics Toolkit* (MRAT). Their toolkit consists of a Unity package, a server that stores recorded sessions and a web interface which includes visualisations of the recorded data that can be used for analysis. To achieve performant recording, Nebeling et al. limit the number of recorded items per frame by introducing different tracking scopes; they distinguish between per frame, per event, and per interval scopes. Although the code for MRAT is publicly available, it is designed for use with the *HoloLens*[1] and although recorded sessions can be analysed using a web interface, they cannot be played back in the virtual environment.

Fanini and Cinque [5] focus on compact recording of user states in virtual environments and interactive remote inspection of recordings by introducing the idea of a *Quantized User Session Volume* (QUSV). A QUSV defines a part of the virtual environment where user sessions can be recorded. The main idea is that within the QUSV, each point in the space is mapped to a specific RGB colour that can be stored instead of the position itself. This allows image-based analysis & compression of the recorded transformations. QUSV represent information strongly quantized and result in less overall precision for larger virtual environments. We argue that this is not appropriate in general because the precision required for analysis might not be known at the time of recording.

Recently, NVIDIA has developed a recording tool that differs from the previously mentioned approaches in that it only records and plays back user input [21]. Although it seems that the performance of this approach is largely independent of the number of elements in the virtual environment, we argue that this approach is not appropriate in general because playing back the original input may not always result in an accurate playback of the original changes in the virtual environment.

---

[1] `https://www.microsoft.com/de-de/hololens` (Accessed on 09/27/2022)

2.2 Exploration of Recordings

For the interactive exploration of a recording, interaction techniques are required that allow users to navigate in time. During rapid temporal navigation, it may happen that a user is not be able to perceive relevant events in the recording. For this reason, it is necessary to provide users with techniques that support their *change awareness* during temporal navigation. Since there is only limited research on the aforementioned techniques for the interactive exploration of immersive recordings, we also examine related techniques for 2D digital video and omnidirectional video that have been published in the related scientific literature.

2.2.1 Exploration of 2D Recordings

Many different exploration and browsing techniques for efficient exploration and browsing have been proposed for digitally recorded 2D videos [22–27]. Because we focus on temporal navigation and techniques to support change awareness, we limit ourselves to the investigation of relevant related video interaction techniques and refer the interested reader to Borgo et al. [23] and Schoeffmann et al. [22] for an extensive overview of video interaction techniques.

Temporal navigation tools for 2D digital video often provide the user with a time slider that allows for an interactive selection of the time in the recording that is of interest [23,25,27–29]. A time slider interface visualises the current point in time of the recording and often provides the user with basic controls that include the actions *play*, *pause*, *fast forward*, *seek*, *skip to the beginning*, and *skip to the end* of a video [23]. Since the length of a time slider is often fixed, but recordings are virtually not limited in length, difficulties can occur when recordings are extensively long, as accurate navigation using a time slider can be difficult. To allow for precise temporal navigation using time slider based interfaces different techniques have been proposed [27,29–32]. The proposed techniques use different approaches to support precise temporal navigation, such as using a non-linear time distribution on the time slider [27,30], refining the visualised time interval [29,31,32], or by supporting precise movements of the handle on the slider [27]. Since time sliders are also commonly used for temporal navigation of immersive recordings (compare **Subsection 2.2.3**), we are going to introduce some of the techniques mentioned above, which we will analyse in more detail in **Chapter 4** when we investigate suitable techniques for temporal navigation of immersive recordings in VR. Hürst et al. [27] proposes the *NLslider* interface, which uses a non-linear time distribution on the time slider such that points in time are distributed further apart from each other around the slider handle, allowing for more precise navigation to points in time near the handle. The *ZoomSlider* interface proposed by Hürst et al. [29] allows the time interval visualised on the time slider to be dynamically modified by vertical mouse movements, refining the temporal navigation range of the user. Hürst et al. [27] propose the *Elastic Skimming* interface, in which the time slider handle can accurately be manipulated using an interaction metaphor which is inspired by pulling a virtual elastic rubber band. Khandelwal et al. [28] proposes an extension to the time slider interface that to some extent supports the change awareness of the user. Their technique visualises a preview of the video content above the slider handle as it is dragged. During this process, the video in the background remains static, giving the user the ability to compare the preview and the current video content, thus supporting the user in identifying changes.

In addition to the aforementioned techniques, adaptive speed based techniques can be used to improve video navigation. Adaptive speed methods often aim to increase the playback speed in places where the recording does not contain relevant information [23,25]. Although such methods can allow for effective temporal navigation and exploration, relevant information in the recording must be identifiable.

Direct object manipulation approaches are an alternative to time slider based interfaces for temporal navigation [26, 33]. In such approaches, objects and their spatio-temporal movements are extracted from the recording. Temporal navigation is made possible by selecting and dragging an object. This is done by using the extracted movements of the selected object to determine the time for which the position of the object in the recording best corresponds to the position to which the user wanted to drag the object.

In this work, we propose techniques for temporal navigation of immersive recordings in VR inspired by time slider based techniques for precise temporal navigation of 2D digital video recordings.

2.2.2 Exploration of Omnidirectional Recordings in VR

Similarly to the exploration of digital 2D video recordings, plenty approaches have been proposed for the exploration of omnidirectional recordings in VR.

In omnidirectional recordings, HMD users can freely change their head orientation while viewing the recording, allowing for restricted spatial navigation in addition to temporal navigation of the recording. Since users are able to freely choose their viewing perspective, it is important to ensure that they are able to perceive all events in the recording that might be relevant to them. Rodriguez [34] found that users watching omnidirectional videos on the commercial platform YouTube[2] spent about 75% of the time within a 90-degree area of the video. To guide users to relevant content that may not be in the user's current field of view, Rodriguez suggests the use of markers and animations. Rothe and Hußmann [35] investigate methods to direct the attention of users towards relevant content. They make use of the notions of *diegetic* and *non-diegetic* cues used in film theory. A diegetic cue is part of the scene, e.g. a singing bird, while a non-diegetic cue is not part of the scene, e.g. a forced rotation. Analysing combinations of diegetic cues from sound, motion, and light changes, Rothe and Hußmann found that sound can efficiently guide the user to points of interests, even if the origin of the sound is not initially in the user's field of view. Rothe et al. [36] examine existing methods for guidance in omnidirectional video, classifying them accordingly to a proposed taxonomy, and analyse the challenges of effective guidance. As a user can only see a fraction of the environment at one point in time, a so-called *fear of missing out* on something was found to often be evoked. Rothe et al. argue that guidance could be used to reduce the user's fear of missing out that may occur when omnidirectional recordings are explored. Another approach to reduce the fear of missing out which does not use explicit guidance was introduced by Yamaguchi et al. [37]. The authors propose a technique that provides the user with a panoramic thumbnail view of the environment so that they can get an overview of the entire environment at the current point in time. Yamaguchi et al. found that their technique alleviated the fear of missing out and improved the quality of the viewing experience.

Nguyen et al. [38] developed a tool for omnidirectional video editing in VR called *Vremiere* and observed that with rapid continuous temporal navigation, which can occur when users drag the handle along the time slider, symptoms similar to motion sickness can

---

[2] `https://www.youtube.com/` (Accessed on 09/27/2022)

occur. To reduce this effect, *Vremiere* uses adaptive vignetting based on the findings of Fernandes and Feiner [39]. Adaptive vignetting dynamically reduces the user's field of view during fast continuous navigation and was found to reduce the effect of motion sickness.

Nguyen et al. [40] present *CollaVR*, a tool that allows for the collaborative review of omnidirectional videos in VR. Their tool enables collaborative exploration and annotation of omnidirectional videos and builds on a time slider interface for temporal navigation. Nguyen et al. allow users to navigate independently in time and visualise each user's temporal position by displaying icons with the user ID on the time slider. In order to support collaboration temporal navigation to a user is made possible via a graphical user interface.

In this work, we propose techniques to support the change awareness of users during temporal navigation inspired by the research on guidance in omnidirectional video. For multi-user exploration of immersive recordings, we provide users with the ability to navigate independently in time and visualise their temporal positions on a time slider similar to Nguyen et al. [40]. To support collaboration in such scenarios, we propose a technique that allows for temporal navigation to a user that, in contrast to the technique used in *CollaVR*, does not require the use of a graphical user interface.

2.2.3 Exploration of Immersive Recordings in VR

Exploring a recording captured in an immersive virtual environment in VR may lead to a more active exploration of the recorded session by users than, for example, exploring the recording on a desktop system. Wang et al. [41] showed that users who re-experience their own recorded session were more likely to explore the recorded virtual environment when the recording was replayed on an HMD compared to users who explored the recording on a desktop system.

Exploring and navigating an immersive recording in VR is different to exploring and navigating a digital 2D video or a omnidirectional video in VR, because in addition to navigation in time, an unrestricted navigation in space is possible [42]. This means that crucial events in the recording may not be visible from the initial point of view and initial point in time of the user who explores the recording, requiring the user to use spatial and temporal navigation in order to discover these events. Since those crucial events may not be visible from a user's current viewing perspective, the *fear of missing out* might be even more pronounced during the exploration of immersive recordings than during the exploration of omnidirectional videos.

For temporal navigation in an immersive recording, many systems provide the user with a conventional time slider and actions such as *playback*, *pause*, *seek*, *rewind* and *fast forward* playback [2, 17, 18, 41, 43–48]. Jackson and Fagan [49] enable users to navigate through time using a time portal. Users can select the time they wanted to navigate to and, after confirming their selection, are teleported to the selected time by flying through the center of the portal. A tangible interface for temporal navigation was proposed by Mahieux et al. [50]. Their interface *SABLIER*, resembles an hourglass and enables users to change the playback speed by tilting the hourglass. Building upon the idea of direct manipulation for browsing digital 2D video proposed by Dragicevic et al. [26], direct manipulation techniques for immersive recordings have been proposed by Wolter et al. [51, 52] and Lilija et al. [42]. Their proposed systems visualize the spatio-temporal trajectories of objects in the scene and allow navigation through time by moving objects along their trajectories.

To support users in identifying events of interest in an immersive recording different techniques have been proposed. Lilija et al. [42] and Wolter et al. [51] provide users with a tool to visualise the spatio-temporal trajectories of all objects that pass through a user defined part of the virtual environment. This technique allows users to interactively discover events in the immersive recording that took place in a specific part of the virtual environment and that can be identified by their spatio-temporal trajectories. In addition, Lilija et al. provide users with a tool that visually highlights all objects in the virtual environment that change in the immersive recording. A different technique to support the identification of interesting events was presented by Kloiber et al. [48], who proposes a system that automatically identifies interesting movements of recorded users and visualises them in the virtual environment. The identification of movements of interest is based on a cluster analysis of the spatial user positions over the duration of the entire recording. In addition, Kloiber et al. visualise the spatio-temporal trajectories of recorded user movement and enable a simplification of the virtual environment that reduces visual clutter that may be caused by the visualisation of the trajectories and the virtual environment. For this, virtual environment textures are replaced by uniform white textures, which allows for a better visual identification of the spatio-temporal trajectories.

In order to share events of interest in an immersive recording with collaborators tools that provide users with the ability to mark points in space and time can be used. Chow et al. [2] propose a technique that allows users to mark spatio-temporal events and navigate to the point in time and space of a marked event. For this purpose, they visualise bookmarks on a time slider interface that correspond to marked events and that can be selected for navigation to those events Wolter et al. [51] introduce the concept of a so-called *time buoy* as a marking mechanism for events at a particular point in space and time. A time buoy is a 3D object that can be placed in space and changes visually when the time point for which it was created is reached in the recording.

Chow et al. [2] investigate asynchronous collaboration in VR and found that users tend to humanise recorded avatars, observing that users exhibited proxemic behaviour toward recorded avatars during a study. When users come into spatial proximity of each other, it can happen that the personal space of a user is violated, which can cause a negative reaction of the user [53]. To avoid potential collision between a recorded user and a user spatially navigating through the recording, Chow et al. [2] developed a prototypical spatial navigation technique that first checks whether a collision could occur at the desired teleportation point within the next few seconds. If a collision would occur, the user is warned and the teleportation destination is adjusted to avoid a collision.

Chow et al. [2] note, that a significant knowledge gap exists when it comes to consuming immersive recordings. Especially in the area of multi-user exploration of immersive recordings in virtual reality, there is, to the best of our knowledge, no research in the related literature. In this work, we propose novel techniques to support multi-user exploration of immersive recordings and temporal navigation techniques to enable precise navigation. Inspired by techniques that use spatio-temporal trajectories to assist users in identifying events of interest, we propose a novel technique that aims to improve users' change awareness during temporal navigation.

3 An Immersive Recording Plugin

In order to investigate techniques for the exploration and temporal navigation of immersive recordings, a way to efficiently record and replay changes in the virtual environment is required. In this chapter we discuss and motivate requirements we identified for an immersive recording tool. We then introduce the immersive recording plugin, which we designed to meet our requirements. To be able to explore suitable techniques for the exploration and temporal navigation of immersive recordings, the recording plugin is integrated into Unity, a game engine that is widely used in the VR community. We detail how the immersive recording plugin is integrated into Unity to enable efficient recording and playback of changes in the virtual environment. To evaluate in which scenarios the recording tool allows for efficient recording and playback, we test the performance of our recording tool independently as well as the performance of the Unity integration.

3.1 Requirements

A plugin that records and plays back changes in a virtual environment must be able to do so efficiently, i.e. without noticeable delay, even when a significant number of changes have to be recorded or played back. To record changes in the scene, the plugin must be able to record transformation data and information about the scene graph. Steptoe and Steed [19] point out the importance of the ability to record and play back audio sources for immersive recordings. In some application scenarios, other information, such as keystrokes, may be of interest and should be recorded. To allow users to record and play back such information without having to modify the plugin itself, a extensible data type supported by the plugin is required. In order to allow for unrestricted temporal navigation, the ability to change the playback speed, to replay the recording backwards and to randomly access the recording is required.

Often recording systems are implemented for use with a single engine such as Unity (compare [2, 11, 20, 41, 47]). This allows for a close integration with the engine but can lead to potential problems when using a new engine. In the worst case dependencies to the original engine must be manually removed and in the case of different programming languages between engines a substantial part of the code may need to be rewritten. One can imagine that this can introduce a tedious amount of work that should be avoided, if possible.

As the development and maintenance of a recording system requires thorough testing, automated testing of the recording plugin should be used to reduce the need for time intensive manual testing and allow for faster development.

In the following, we provide a concise list of the requirements we motivated for our recording plugin:

- Functional requirements
  - Record & replay scene graph information and transformation data
  - Record & replay microphone and application audio
  - Record & replay an extensible data type that can be modified by the user
  - Record & replay without stalling
  - Efficient random-access of the recording during playback
- Non-functional requirements
  - Portability of the software
  - Ability for automated testing of the software

3.2 Recording Plugin

In the following, the architecture of the recording plugin designed and developed throughout the course of this thesis is described and motivated on the basis of the previously established requirements.

Portability & Automated Testing

To make the recording plugin usable independently of the game engine or the operating system used, we developed a plugin that can be used on Android, Linux or Windows with different programming languages. The plugin supports cross-compilation to create a library for the target operating system that exposes endpoints in C and was developed test-driven to reduce the need for manual testing during development.

Multi-Threaded Recording & Playback

In order to allow for efficient creation and replay of recordings, the plugin relies on multi-threading for both recording and playback. A separate thread is used to write the already recorded data to the file system during recording while a different thread is used to store new recorded data in RAM. For the replay of a recording, a separate thread is tasked with loading new data from the corresponding recording files while the main thread is used to allow for the retrieval of the recorded and already loaded data. A more detailed visualisation of the multi-threaded behaviour can be found in the appendix in **Figure A.1**.

To enable asynchronous loading of recordings, a ring buffer data structure is used to store the loaded data. The ring buffer stores two or more buffers, which each store data loaded from a recording file. Each buffer is associated with a time interval and only stores data recorded for point in time within that interval. The associated time intervals are always mutually disjoint and their union is a single continuous interval. If the time for which data is requested during forward playback is greater than the times contained in the time interval of a buffer, the buffer contains no relevant data and can be used to load new data, while a different buffer whose interval contains the point in time for which data is requested is used to retrieve data. This behaviour is illustrated in **Figure 3.1**.

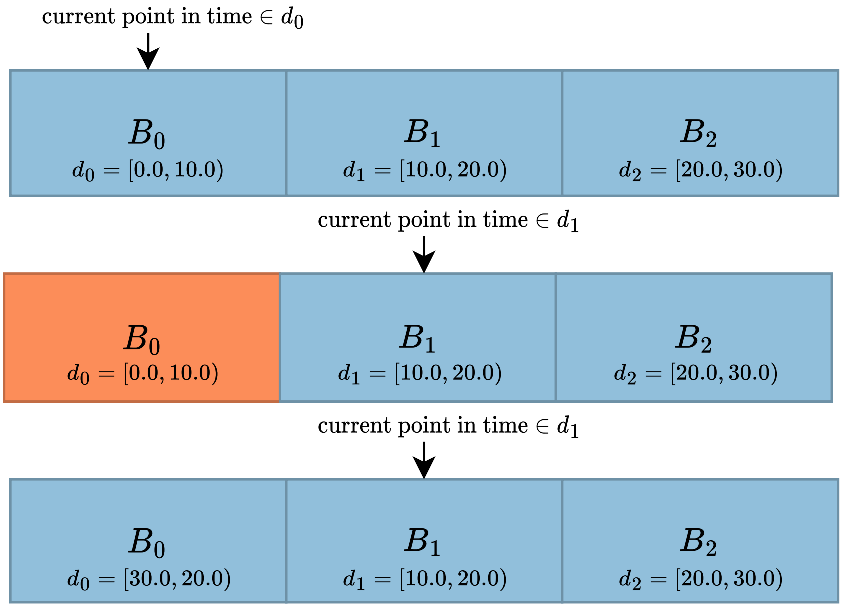


Figure 3.1: Simplified illustration of the loading of data associated with time intervals based on ring buffering. The interval associated with a buffer is denoted by d_i. If the time interval boundaries values are both less than the current time value, the buffer is marked in orange and else in blue. Since the current point in time used for forward replay increases continuously, the buffer marked in orange contains no relevant data and can be filled with new data as illustrated in the last step at the bottom of the figure.

Recording Data Types

The plugin uses inheritance to allow for efficient code reuse and provides an abstract base recording data type and three derived recording data types that meet our previously defined requirements. A simplified UML class diagram including the recording data types can be seen in **Figure 3.2**. To record object transformations as well as information about each object, such as the current parent in the scene graph and whether an object is currently active in the scene graph, the transformation data type is used. The transformation data type consists of the identifier of the object associated with the data, the time for which the data is recorded, the position, rotation and scale information of the object, and the identifier of the current parent object in the scene graph. By recording information about the current parent in the scene graph, this data type can be used to record changes in the scene graph. If transformation data is requested for a point in time for which no data has been recorded, interpolation between the recorded data of the most recent previous point in time and the recorded data of the next subsequent point in time is used to approximate the data for the requested time. To record audio, we introduce an audio data type that consists of the identifier of the audio source associated with the data, the time for which the data is recorded and a buffer of 4,800 floats that contains the sound samples to be recorded. It can be used for recording the microphone and system audio, which is often necessary in order to allow for an informed analysis of a recording [19]. To provide users with a tool for recording more generic data, we introduce a generic data type that allows the recording of integer, float, and character arrays of a predefined size to support the extension of the recording routine to some degree. The generic data type can be used to record additional data that may be relevant to the analysis of the recording, for example, latency information, keystrokes, eye-tracking data. If a more specific data type is required, the existing hierarchy can easily be extended.

Recording Files

During recording, data is written to the file system in a binary format, as this allows for efficient writing to the file system and for efficient loading of data to memory during playback of a recording. A single binary file is created for each type of data, allowing efficient access to the stored data. This approach focuses on efficient storing and loading of data, but not on reducing the memory size of the recording files. Depending on the available file system space, a modified approach using compression may be required. Compression does not only affect the space required to store a recording, but can also decrease loading and writing times at the cost of additional compression and decompression time. To reduce the amount of recorded data, Luttermann et al. [17] introduce a recording strategy in which a data point is recorded only if it differs by a certain threshold from the previously recorded data point and the following data point. A similar approach could be integrated into the proposed recording tool as well in addition to a compression of the created recording files.

In addition to the files that are created for the recording data types, a metafile is created that stores information about all recorded objects in the scene graph. This is done in order

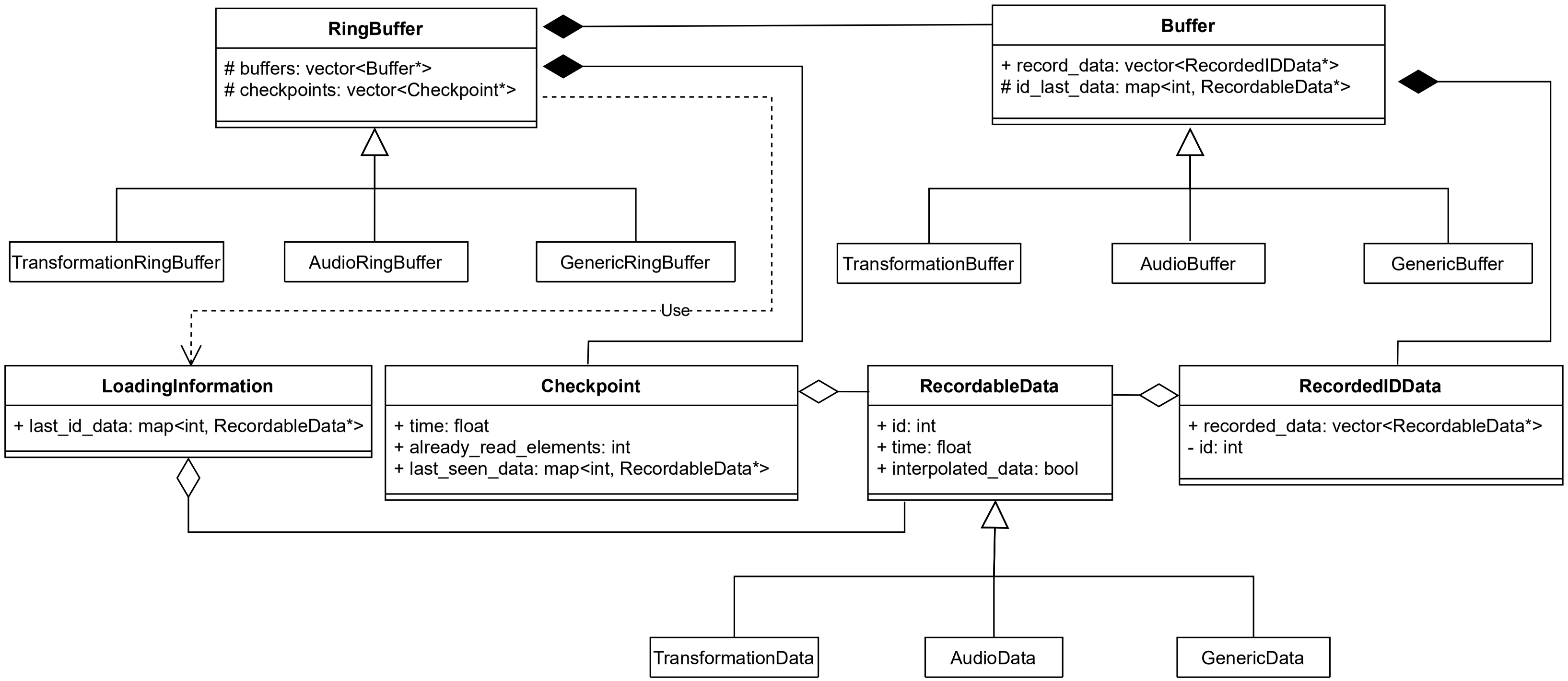


Figure 3.2: Simplified UML class diagram of the class and object hierarchy.

to be able to check whether all objects present in a recording are also present in the virtual environment in which we want to replay the recording.

In order to allow for easier post-processing of the recorded data, the transformation data is automatically written to a CSV file after the recording is finished, following the proposal of Steptoe and Steed [19] to store recorded data in a human readable format. In addition, a WAV file is created for each recorded audio source so that the recorded audio can be played back individually without having to replay the immersive recording in a virtual environment.

Efficient Random-Access of Immersive Recordings

Checkpoints enable more efficient loading of recorded data and have been used by Fominykh et al. [8] and the Unreal game engine replay system [54] to improve replay performance. We create and use checkpoints for more efficient loading at runtime. A checkpoint is defined by a timestamp, the byte position within the recording file associated with this timestamp, and the most recent recorded transformation, audio and generic data up to this timestamp, as is illustrated in **Figure 3.2**.

To reduce the number of binary read operations used to load data from a recording file, the recording data is read in chunks of several data elements. The chunked data is then processed such that the contained data is loaded correctly.

Simultaneous Recording & Playback

For asynchronous collaboration and editing of recordings it is useful to be able to create a new recording while exploring an immersive recording which was previously created [8]. One can imagine a recording of a lecture where certain parts do not contain relevant information and could be removed while other parts lack necessary information. Instead of creating a completely new recording of the lecture, a modified version can be created by recording changes to the original version. If a part does not contain relevant information the user can fast forward until a part containing relevant information is reached. As the current scene is being recorded, this means that the part without relevant information is not contained in the modified recording. For parts where relevant information is missing, the replay can be paused and the relevant information can be added. The plugin allows for multiple recording and playback devices to be used at the same time, making it possible to simultaneously play back different recordings and to create new recordings while playing back existing recordings. We refer the interested reader to Fominykh et al. [8] for an overview of further use cases and scenarios in which simultaneous recording and replay of immersive recordings can be used.

3.3 Network Distribution & Post-Processing

To support the creation of immersive recordings in scenarios where multiple remote distributed users are present in the virtual environment, we use a web service as a central data repository to facilitate the distribution of the recording files. This setup is in general similar to the setup used by Nebeling et al. [6]. The recording files are automatically uploaded to the web service and can be downloaded by users for playback. This allows for a fast distribution of recording files where a user can create a recording on an HMD which can afterwards directly be played back on a desktop PC after the recording file was downloaded from the web service.

The web service can further be used for automated post-processing of the data. As a prototype for such automatic post-processing, we have implemented a functionality that generates text from recorded audio data. The created WAV files can be uploaded to the web service, where a timestamped transcript is automatically generated by a pretrained *DeepSpeech* machine learning model [55]. The generated transcript can later be retrieved and allows for an enrichment of the recording, for example similar to the transcript visualisation used by DeCamp et al. [56].

3.4 Unity Integration

To integrate the recording plugin into the Unity game engine the endpoints exposed by the plugin have to be declared inside of a C# script with C linkage. The plugin functions can then be called from inside the C# script. To enable functions defined in a C# script to be called from a native plugin, delegates can be used. Delegates allow for passing of methods as arguments to other methods. This functionality is used to support the development of Unity applications by displaying debug output that has been created by the plugin in the Unity console.

Data Recording

To record changes in a virtual environment, information about the transformations of all objects one wants to record is required. For this purpose, a recording script is attached to all objects that one wants to record and for each object it is checked whether new information can be recorded. The reason for this is the fact that usually only a small part of all objects in the scene changes at the same time. For such scenes, recording transformations with and without new information would create an overhead that can be avoided by considering only data for which the relevant attributes changes. When new information is available, the plugin is called and the transformation data of the corresponding object is recorded.

The user is able to specify the maximum rate at which the transformation data is being recorded. This allows for a control over the performance required for recording as well as of the file system space required to store the recording, but comes at the cost of reducing the number of available data points and the risk of potentially not recording relevant data, as further discussed in Steptoe and Steed [19].

Replay Preparation

Before playback of a recording begins, a list of all objects present in the recording is fetched from the metafile through the plugin and used to ensure that all objects required for a playback are present in the current scene. Since the instance identifiers of Unity objects are not persistent across restarts of the software, it is necessary to perform this check independently of the object instance identifiers. For this purpose, a string that stores the initial location of the object in the scene graph is used to identify the object in the scene.

It would be possible to instantiate all objects that are missing from the current scene but are present in the recording file to be played back. However, this may require information about the components of each object that would have to be explicitly recorded increasing the manual overhead for creating a recording. For this reason, we assume that all objects present in the recording are present in the scene before a playback of the recording starts.

3.5 Performance Evaluation

In order to assess for which scenarios the proposed plugin enables performant creation and playback of immersive recordings, it is necessary to investigate the performance of the plugin as well as the performance of the Unity integration. During playback, transformation data requested for a point in time for which no data was recorded can be interpolated if data was recorded for a previous and subsequent point in time. This allows for continuous replay of transformation data at a rate that can be higher than the rate at which the data was recorded. Although Steptoe and Steed [19] argue that the rate at which data is recorded should match the refresh rate of the display, we argue that this might not be appropriate in general for virtual reality applications where refresh rates of 120 Hz or more are not uncommon, as for example an immersive recording of a virtual lecture might be unlikely to have any important high-frequency motions that must be recorded. To evaluate the performance of our plugin, we record the virtual environment at a rate of 30 steps per second, a recording rate commonly supported by many commercially available video recorders.

For the measurements in the following performance evaluations, a desktop PC with 32 GB RAM, a AMD Ryzen 7 3800X CPU, a NVIDIA GeForce GTX 1080 graphics card and a 500 GB NVMe SSD was used.

3.5.1 Recording Plugin Performance

To evaluate the performance of the plugin in terms of recording and playback of transformation data, we created a situation, in which a certain number of objects that are constantly moving are recorded and played back at a rate of 30 steps per second for 5 minutes. The time required for recording and playback of a single object at a single point in time is then calculated by averaging the time required for all recording and playback operations. From the time required for recording and playback of a single object, the highest achievable frame rate can then be estimated and later be used for comparison with the measured frame rate of the Unity integration.

To investigate the influence of the write buffer size, the recording performance is measured for different sizes. As is illustrated in **Figure 3.3**, the size of the write buffer influences the recording performance. Depending on the number of objects that are recorded, different write buffer sizes perform best. It can be seen that performance decreases as soon as more objects are recorded than the write buffer is able to store. As this means that the write and read buffer are swapped and data is written to the file system before another swap between the write and read buffer can be done, it is possible that stalling can occur if writing data to the file system requires more time than time elapses before the newly emptied write buffer is filled again. The time required for recording a single object with a write buffer size of 1,000 increases by a factor of 27.75 when 1,250 objects instead of 1,000 objects are being recorded. This sudden increase might be caused by an overhead that is due to the repeated subsequent write operations to the file system, in which only 1,000 transformation data objects are written to the file system. In general, one can see that a write buffer size of 7,500 performs well for our use case as long as no more than this number of objects need to be recorded at once.

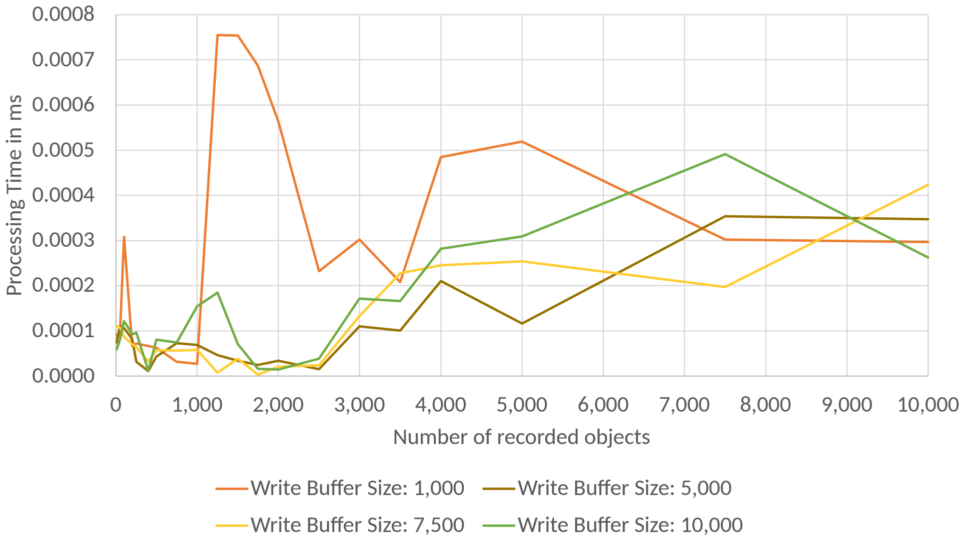
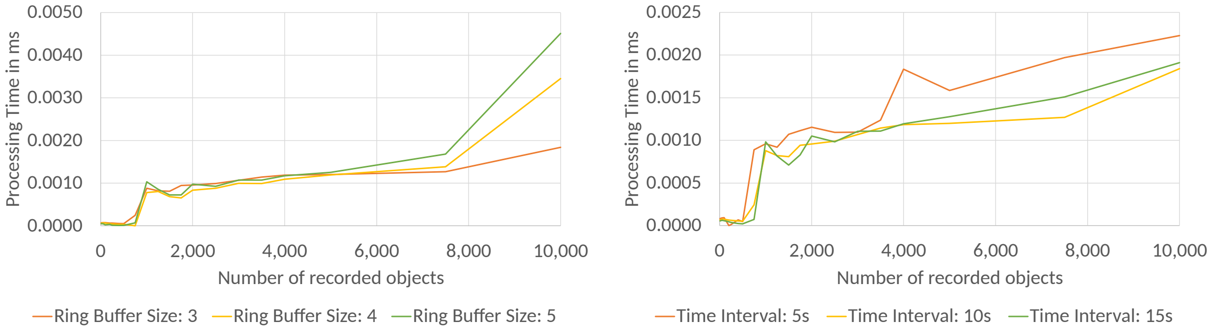


Figure 3.3: Average time required to record transformation data of a single object for a given point in time across different write buffer sizes.

When examining the playback performance, the influence of the ring buffer size and the influence of the time interval length for which data is stored in each buffer are best examined separately. As is shown in **Figure 3.4a**, playback performance is in general similar for ring buffers of size 3, 4 and 5 when each buffer stores 10 seconds of data. However, as more objects are recorded, the performance decreases with increasing ring buffer size. A possible reason for this may be that asynchronous loading is not yet completely independent of the ring buffer size. In general, it can be said that a ring buffer consisting of 3 buffers performs best overall for our use case.

When examining the influence of the time for which data is stored in each buffer, it is shown in **Figure 3.4b** that storing 10 or 15 seconds of data for each buffer results in better playback performance than storing 5 seconds of data. One reason for this could be the fact that data is loaded more frequently when 5 seconds of data are stored in each buffer compared to when 10 seconds or 15 seconds of data are stored.



(a) Note, that each buffer used is associated with a 10 second long time interval.

(b) Note, that each ring buffer used is of size 3.

Figure 3.4: Average time required to replay the transformation data of a single object for a point in time. In **(a)** the performance is measured for ring buffer of different size. In **(b)** the performance is measured for ring buffer of a fixed size but with different time interval lengths associated.

Comparing the time required for recording and playback given in **Figure 3.5**, one can see that data can be recorded more efficiently than played back and that the average time required for recording and playback increases as the number of objects to be recorded or played back increases.

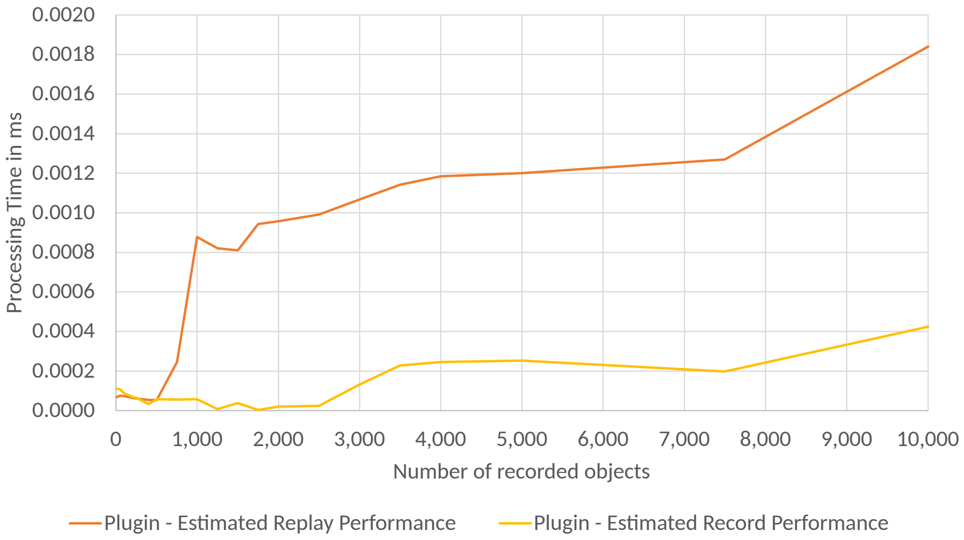


Figure 3.5: Comparison of the average time required to record and replay transformation data of a single object for a single point in time. For replay a ring buffer of size 3 with time intervals of 10 seconds is used. For recording a write buffer of size 7,500 is used. As can be seen more time is required for replaying than for recording.

Based on the average time required to record and playback an object at a given point in time, it is possible to estimate the number of FPS that can be achieved during recording and playback. As is shown in **Figure 3.6**, the number of estimated FPS for recording scenes with up to 10,000 constantly moving objects is above 200 FPS. When playing back recordings with up to 7,500 constantly moving objects, the estimated number of FPS is above 100 FPS. Although the number of estimated FPS suggests that efficient recording and playback is possible, in practice the performance when playing back a recording may be lower because the data for playback is usually requested more than 30 times per second. In general, it can be assumed that data for playback is requested for each frame in order to avoid inconsistent states [19]. For this reason, it is necessary to also examine the performance of the Unity integration, which will be done in the following subsection.

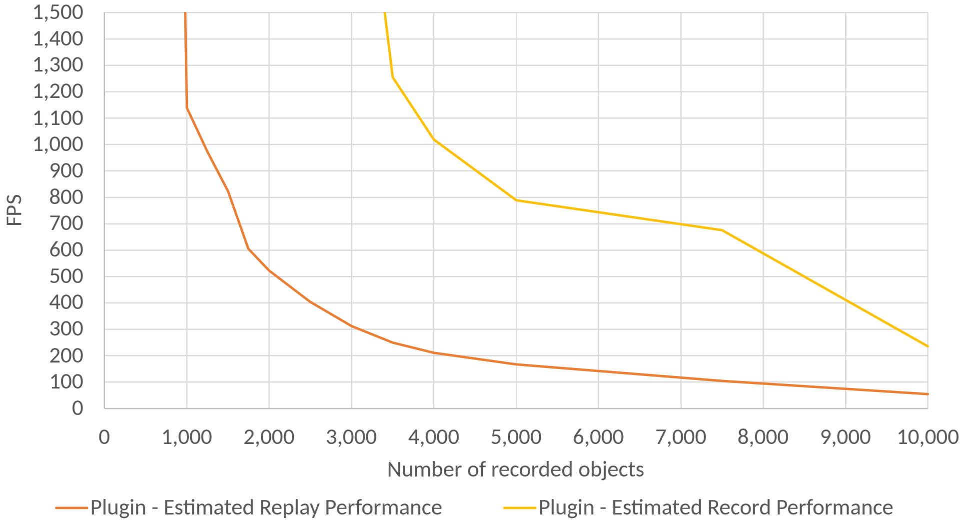


Figure 3.6: Comparison of the estimated number of FPS that can be achieved during recording and play back. To estimate the FPS for replay the measurements taken for a ring buffer of size 3 with time intervals of 10 seconds are used. To estimate the FPS for recording the measurements taken for a write buffer of size 7,500 are used. For presentation purposes, the maximum number of FPS shown is limited to 1,500.

In order to compute the memory required to store an immersive recording the memory sizes required to store transformation data objects, audio data objects and generic data objects can be used. To store a single transformation data object 96 bytes and to store a single generic data object 100 bytes are required. The amount of memory required to store a single audio data object is roughly 19.3 kilobyte. More memory is required to store an audio data object compared to a transformation data object due to the fact that the audio data type consists of a float array that can store up to 4,800 audio samples.

The memory required to record n transformations at a rate of a steps per second for t seconds can be computed by $m = n \cdot a \cdot t \cdot 96$ Byte. Recording 500 constantly moving transformations at a rate of 30 steps per second for 60 minutes thus roughly produces 5.18 GB of data.

Although the memory required to record a single audio data object is much larger than the memory required to store a single transformation data object, to record an audio source that has a sampling rate of at most 48 kHz requires at most 10 recording steps per second as each audio data object can store up to 4,800 audio samples. Thus, the memory required for recording t seconds of audio data for n audio sources is given by $m = n \cdot t \cdot 10 \cdot 19.3$ Kilobyte. Recording microphone and world audio for a recording of 60 minutes thus approximately produces 1.38 GB of data.

For a recording of one hour for which the transformation data of 500 constantly moving objects are recorded at a rate of 30 steps per second and the microphone and world audio is being recorded, thus roughly 6.56 GB of data are produced. Although the amount of data produced could further be reduced by using compression algorithms, a recording of 6.56 GB can already be stored on modern HMD's such as the Oculus Quest 2 which comes with either 128 GB or 256 GB of file system storage. However, reducing the amount of data recorded, e.g. by not recording data that can be interpolated by previous and subsequent data, leads to less file system space required to store a recording and may also improve playback performance as less data needs to be loaded at the cost of additional compression and decompression overheads for recording and playback during runtime.

3.5.2 Unity Integration Performance

During the playback performance evaluation of the Unity integration of the plugin, data is requested for each frame instead of the 30 times per second in our preliminary evaluation to allow for playback without visual stuttering. In addition to the increased number of data requests during playback, the Unity scripts required for recording and playback create an additional overhead that affects overall performance. It is shown in **Figure 3.7** that the estimated number of FPS for recording and playback is always higher than the measured number of FPS for recording and playback for the Unity integration.

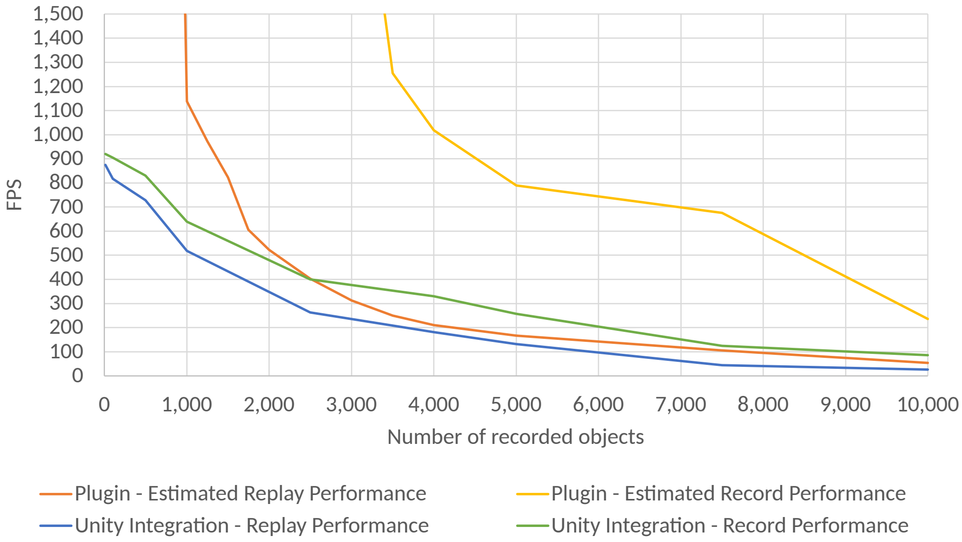


Figure 3.7: Comparison of the estimated number of FPS for recording and replaying and the measured number of FPS of the Unity integration.

For many virtual reality applications, a frame rate of at least 120 FPS is required to provide an immersive experience. Although the measured number of FPS was lower than the estimated number of FPS, the Unity integration allows for efficient recording and playback of scenes with up to 5,000 constantly moving objects with more than 120 FPS and could be used for virtual reality applications in such scenarios.

As it is difficult to estimate what kind of scenes contain 5,000 constantly moving objects, we try to illustrate this by an example scenario. The users' avatars we used to develop temporal navigation techniques (compare **Chapter 4**) are composed of 25 scene graph nodes. It would therefore be possible to create and replay an immersive recording with more than 200 simultaneously moving avatars at more than 120 FPS.

4 Temporal Navigation & Exploration of Immersive Recordings in VR

Working with spatio-temporal recordings requires appropriate temporal navigation techniques that allow for precise navigation while being comfortable to use. In the following, we analyse several existing temporal navigation techniques and propose a set of techniques for temporal navigation of immersive recordings in virtual reality. Since immersive recordings allow unrestricted temporal and spatial navigation, appropriate techniques for exploring recordings are needed to avoid fear of missing out, and to assist the user in discovering and investigating events of interest in the recording. For this purpose, we examine existing techniques for guidance in omnidirectional video and awareness in virtual reality, and based on our findings, propose techniques which aim at supporting the user in discovering events of interest in immersive recordings.

In many cases, e.g. when analysing study data, it might be helpful to analyse immersive recordings together with a collaborator. However, as users do not necessarily have to be at the same point in time within the recording, the question arises as to how collaboration can be best supported in such cases, as users may be observing completely different states of the virtual environment and may therefore have no consistent common workspace, which is often a prerequisite for collaboration. For this reason, we analyse existing techniques in the literature that focus on supporting collaboration during exploration of recordings in a multi-user context. Based on our findings, we then propose techniques to support collaboration during the exploration of immersive recordings.

4.1 Temporal Navigation

In order to investigate techniques for temporal navigation of immersive recordings in VR it is worthwhile to investigate existing temporal navigation techniques for digital 2D and omnidirectional recordings. After thorough analysis of different temporal navigation techniques, we propose a set of different techniques for temporal navigation of immersive recording in virtual reality.

4.1.1 Analysis of Temporal Navigation Techniques

Due to the ability to freely choose a viewing perspective into the virtual environment, not all temporal navigation techniques from 2D videos may be directly applicable to immersive recordings. Video techniques such as *EgoScanning* [25] or *TwistLens* [57] which make use of 2D image information may not be directly applicable because the required information might have to be generated first from the immersive recording.

As a large proportion of temporal navigation techniques for 2D recordings are based on a time slider interface, as found by Schoeffman et al. [22], it is worthwhile to analyse different time slider based techniques to identify their strengths and weaknesses.

Since the length of a time slider is often limited, but recordings are virtually not limited in length, difficulties can occur when recordings are very long, as accurate navigation using a time slider can be difficult. For this reason, Koike et al. [30] proposed the *TimeSlider* interface, in which the time steps on the slider are non-linearly distributed. Near the slider handle, the scale is fine and allows for precise navigation, while the scale is coarser further away from the slider handle as illustrated in **Figure 4.1**.

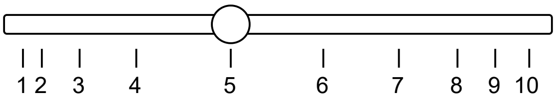


Figure 4.1: Illustration of a non-linear distribution of points in time on the time slider. Note, that around the slider handle points in time are distributed such that a more precise temporal navigation to a point in time close to the current point in time is made possible.

When the slider is moved, the scales are automatically updated. In this way a specific value can be selected precisely even if the recording is long. However, a drawback of this technique is that points in time are mapped to different positions on the time slider depending on the current position of the slider handle. A user who has memorised the position of a point in time on the *TimeSlider* may be confused when later trying to navigate to the same point in time by selecting the memorised position on the *TimeSlider*, as the position may now correspond to a different point in time.

To enable more precise temporal navigation with a time slider based interface, Hürst et al. [27] propose *ElasticSkimming*, an approach that dynamically changes the speed at which temporal navigation is performed. This is done by allowing for accurate manipulation of the time slider handle using an interaction metaphor which is inspired by pulling a virtual elastic rubber band. When the user moves the cursor away from the time handle, the tension of the virtual elastic rubber band increases and the time handle moves in the direction of the cursor. If the distance between the cursor and the time handle is small, the tension is lower and the time handle moves more slowly. The behaviour is illustrated in **Figure 4.2**. Although this technique allows for more precise temporal navigation, it does not solve the problem that a single point on the time slider can potentially correspond to a larger time interval of the recording.

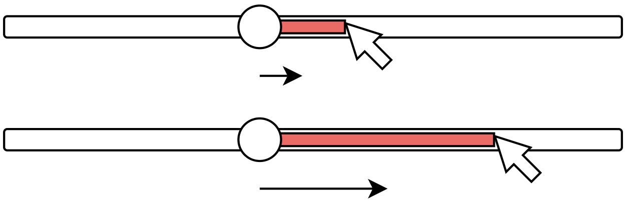


Figure 4.2: Illustration of the virtual rubber band metaphor used by the ElasticSkimming interface. Note, that the movement of the slider handle depends on the distance between the cursor and the slider handle.

Hürst et al. [58] present the *ZoomSlider* interface, which makes it possible to change the scale of the slider by moving the cursor in a vertical direction. While moving the cursor vertically changes the time scale of the slider as is illustrated in **Figure 4.7**, moving the cursor horizontally allows for temporal navigation. An advantage of this technique is that the user is not required to select the time slider itself to be able to navigate through time, but can position the cursor anywhere on the screen. In principle, adjustments of the time slider scale can allow for precise temporal navigation. However, as horizontal and vertical cursor movements are typically not independent, movements of the cursor can potentially lead to unintentional changes of the scale or temporal navigation.

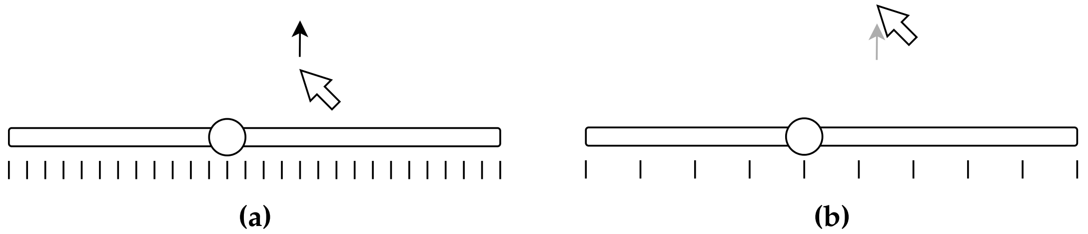


Figure 4.3: Illustration of the modification of the time slider scale by vertical cursor movements. In **(a)** the distribution of keypoints on the time slider before vertical movement of the cursor is shown. After vertical cursor movement the scale and distribution of keypoints on the time slider is modified as shown in **(b)**.

Time slider based techniques are also commonly used for temporal navigation of immersive recordings in VR. However, often only a basic time slider with simple controls is used [2,17,18,41–48]. Selecting a point in time on a time slider for temporal navigation in VR is often done through raycasting, where the ray is cast from the user's hand. Because the time slider often occupies only a small portion of the user's field of view and the width of the time slider is often much greater than its height, selecting the time slider through raycasting can be cumbersome. To some extent, selecting a point on the time slider in VR through raycasting can be compared to bivariate target acquisition in 2D, for which a modification of *Fitt's law* [59] can be used to predict the time required to make a selection [60]. Accot and Zhai [60] found that the acquisition performance of a target defined by width and height in 2D can be modelled by equation **Equation 4.1** where T is the time needed to acquire the target, D is the distance of the cursor from the target, W is the width of the target, H is the height of the target, $a \in [-50, 200], b \in [100, 170]$, and $\eta \in [1/7, 1/3]$.

$$T = a + b \cdot \log_2 \left(\sqrt{\left(\frac{D}{W}\right)^2 + \eta \cdot \left(\frac{D}{H}\right)^2} + 1 \right) \tag{4.1}$$

If the height is only a fraction of the width $H = \frac{W}{\alpha}$ then the equation can be reformulated to see how α influences the target acquisition performance.

$$T \geq a + b \cdot \log_2 \left(\sqrt{\eta} \cdot \alpha \cdot \left(\frac{D}{W}\right) + 1 \right) \tag{4.2}$$

As can be seen in **Equation 4.2** the target acquisition performance increases logarithmic with increasing α value.

Dragging a time handle along a time slider through raycasting often requires the user to stay within the boundaries of the time slider. This interaction is to some extend similar to steering in 2 dimensions for which performance can be estimated by the *Steering law* [61]. As width and height of a time slider are in most cases constant moving the time handle can be described as a tunnel travelling task for which the required time can be estimated by **Equation 4.3**, where a and b are constants, W corresponds to the height and A corresponds with the width of the time slider.

$$T = a + b \cdot \frac{A}{W} \tag{4.3}$$

If the height is only a fraction of the width $W = \frac{A}{\alpha}$ then $T = a + b \cdot \alpha$ showing that the time required for steering grows linear with respect to α.

As can be seen, both the time required for selecting the time slider and the time for dragging the handle along the time slider increase with increasing α value. This means that the height of the time slider should ideally be as close as possible to the width of the time slider. However, since increasing the height may result in occlusion of relevant content, and decreasing the width may result in compromising the ability to clearly associate a point in time with a point on the time slider, a suitable compromise has to be found. Alternatively, a time slider based interface similar to the *ZoomSlider* interface could be used where users are able to navigate in time without being required to select the time slider or the time slider handle.

Investigating factors that affect the selection of 3D objects, Argelaguet and Andujar [62] note that when the input of a selection tool is not filtered, hand fixation may be required to reduce hand tremor and to keep the selection tool stable in order to enable accurate selection. However, if manual hand fixation is required the user may become fatigued and possibly discomforted. We argue that for time slider based temporal navigation through raycasting, input filtering should be used to support precise temporal navigation, as already small unintentional movements of the time slider handle can result in unintentional temporal navigation over a considerable temporal distance.

Lilija et al. [42] propose the *Who Put That There* system, which builds on the idea of direct manipulation for temporal navigation and visualises the spatio-temporal movements of objects that change in a immersive recording. For temporal navigation, objects can be grabbed and moved along their spatio-temporal trajectories by the user. The authors argue that trajectories in space can be much longer, while the length of time sliders is limited. Participants found the *Who Put That There* system to feel more interactive than temporal navigation using a basic time slider interface which was additionally provided for the participants. However, as the position and orientation of the working space in which a user operates plays an important role in the user's comfort [62], discomfort may be experienced by the user, as the technique of Lilija et al. does not restrict the working space of their trajectory scrubbing method. One can imagine a recording in which a mouse moves across the floor. To navigate to the corresponding point in time at which the mouse was at a specific position on the floor, the user would potentially be forced to crouch in order to touch the trajectory on the floor, which can lead to fatigue and discomfort. Lilija et al. note that their system requires active engagement and can potentially become exhausting or cumbersome.

As the *Who Put That There* system is based on trajectories it is important to identify potential weaknesses of trajectories. Lilija et al. [42] found that entangled trajectories can potentially lead to confusion as noted by a study participant. In an expert study of Kloiber et al. [48] a participant expressed discomfort when trajectories were passing through his virtual body. Furthermore, the spatio-temporal trajectories of objects that did not move continuously during recording may cause confusion for the user during trajectory scrubbing, as time in the scene might not change continuously when the object is moved continuously along its trajectory. While trajectory based temporal navigation can potentially provide a more interactive experience than temporal navigation techniques based on time sliders, the development of trajectory-based techniques is faced with challenges such as providing a suitable working space, dealing with entangled trajectories and trajectories that violate the personal space of users.

The *rubber slider* proposed by Cheymol et al. [63] extends the time slider interface by allowing the user to manipulate the slider by pushing and pulling it using a virtual rubber band metaphor. By pulling the time slider, it is bent towards the user in a similar way to a rubber band fixed at both end points, and the displayed time range is reduced, allowing more precise temporal navigation. As it can prove difficult to manipulate both the scale of the time slider and the time simultaneously, hooks metaphor is used to define hooks that hold the bent time slider in position. Although the rubber slider allows for intuitive interaction and scale modification, we argue that temporal navigation at the end points of the rubber slider can potentially become difficult, as the selection of points in time at the end points is different from the selection of points in time in the middle, as precise movement in two dimensions is required of the user.

Petry and Huber [64] propose a gesture based temporal navigation technique for omnidirectional videos. Users are able to navigate in time by performing arm movements that are detected using a Leap Motion controller. Performing a push gesture corresponds to the play command, holding the arm pauses playback, and moving the arm horizontally enables forward or backward temporal navigation. Since the gestures are detected using a Leap Motion Controller that is attached to the HMD, the arm of the user must always be in the field of view of the Leap Motion Controller, which restricts the working space of the user.

We argue that a temporal navigation technique should provide for an appropriate working space in order to allow for comfortable use over extended periods of time. As the working space of the *Who Put That There* system is not constrained temporal navigation might become uncomfortable and potentially exhausting for the user. Time slider based temporal navigation techniques which use raycasting often provide a more restricted working space but can be uncomfortable to use as the selection of the time slider and the time slider handle can be cumbersome. We argue that the working space of such techniques can be improved, e.g. similar to the *ZoomSlider* technique, to allow for more comfortable temporal navigation in VR. Although the gesture based temporal navigation technique proposed by Petry and Huber [64] might be uncomfortable when used over a prolonged period of time, as the user has to position their arm such that it is visible to the Leap Motion controller, a similar gesture based technique without this limitation could allow comfortable temporal navigation in virtual reality.

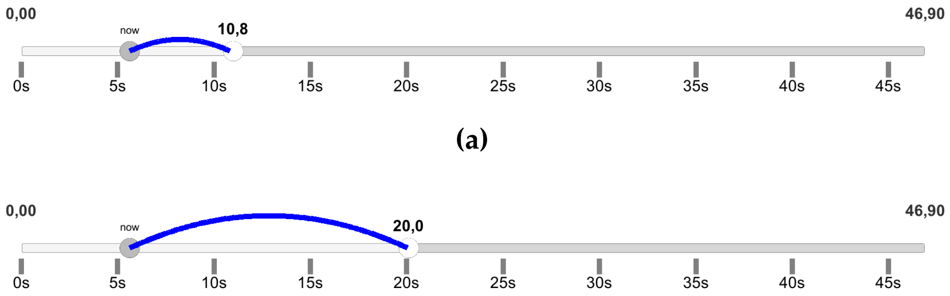
4.1.2 Proposed Temporal Navigation Techniques

In the following we focus on temporal navigation techniques based on time slider interfaces which provide the user with an overview of the temporal position in the recording at which he or she is currently located. Inspired by the gesture based temporal navigation technique proposed by Petry and Huber [64], we propose a gesture based temporal navigation technique that aims to provide the user with a more appropriate working space.

Visualisation of Temporal Distances Covered by Temporal Navigation

Time slider based interfaces for temporal navigation inform the user about his or her temporal position in the recording. During temporal navigation, it might be useful for the user to be aware of the temporal distance they have travelled. One can imagine a user navigating in time, but realising that they have navigated beyond the point in time of interest to them and subsequently deciding to return to their original position in time. In such a scenario, it may be important to support the user in identifying their original position in time, e.g. by visualising the temporal distance they have travelled during temporal navigation. To the best of our knowledge, there has been no research carried out that discusses techniques for the visualisation of the distance covered during temporal navigation.

To inform the user of the temporal distance travelled during temporal navigation, we visualise the distance using a preview arc that connects the initial point in time in the recording at which the user begins temporal navigation and the current point in time that the user is navigating to. The preview arc is defined by a quadratic function that connects the two points in time on the time slider, and is defined such that the height of the arc grows with the temporal distance travelled, as is shown in **Figure 4.4**.



(a)

(b)

Figure 4.4: Temporal distances covered by temporal navigation visualised through the preview arc. Note, that the height of the preview arc changes with the temporal distance travelled as can be seen by comparing **(a)** and **(b)**.

Reference Thumbstick Technique

As a basic reference technique for temporal navigation, we use a simple time slider interface in which temporal navigation is made possible by using raycasting and a thumbstick, an elastic input device used for two-dimensional input. The displacement of the thumbstick is mapped non-linearly to determine how the current time in the recording will be modified. Moving the thumbstick to the left essentially results in backward navigation in time, whereas moving the thumbstick to the right results in forward navigation in time. Temporal navigation is activated as soon as the user touches the thumbstick to indicate to the user that time can now be navigated.

Let $\vec{d} = \binom{x}{y}$, $x, y \in [-1, 1]$ be the displacement of the thumbstick, Δr be the time for which continuously $|\vec{d}| \geq 0.95$ up to this moment, then the modification to the current time Δt is computed as shown in equation **Equation 4.4**.

$$\begin{cases} \Delta t = \frac{\log\left(\frac{\Delta r}{5}+10\right)}{1+e^{\left(0.5-|\vec{d}|\right)\cdot 10}} & \text{if } x \geq 0 \\ \Delta t = \frac{-\log\left(\frac{\Delta r}{5}+10\right)}{1+e^{\left(0.5-|\vec{d}|\right)\cdot 10}} & \text{else} \end{cases} \tag{4.4}$$

The nonlinear mapping allows the user to navigate precisely in time through small movements of the thumbstick, but also allows for efficient navigation over larger temporal distances through continuous larger displacements of the thumbstick. The modification of the current time Δt is related to Δr allowing for further temporal navigation when the thumbstick displacement is above a threshold for more than a specified amount of time. The influence of a displacement in the x dimension on Δt for $\Delta r = 0$ is visualised in **Figure 4.5**.

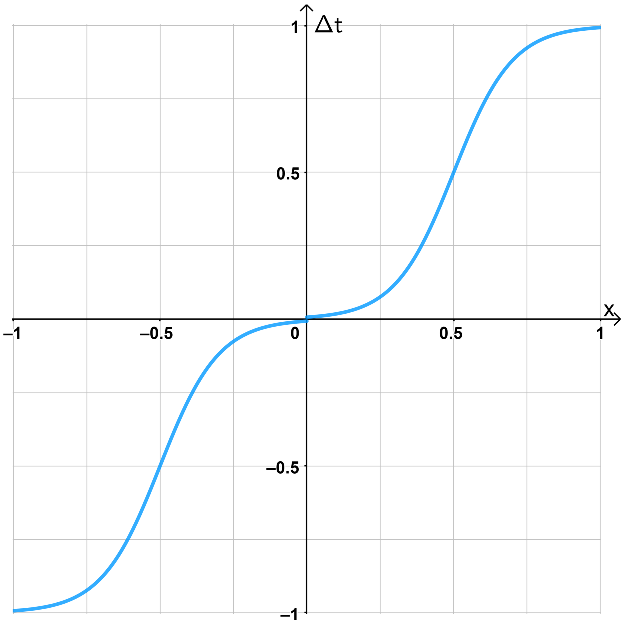
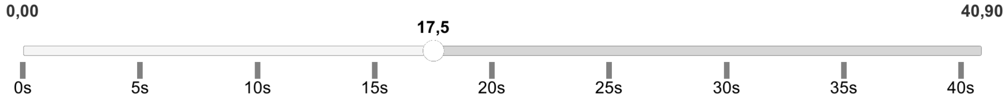


Figure 4.5: Transfer function for nonlinear input mapping of thumbstick displacement. Note that for this plot $\Delta r = 0$ and $y = 0$.

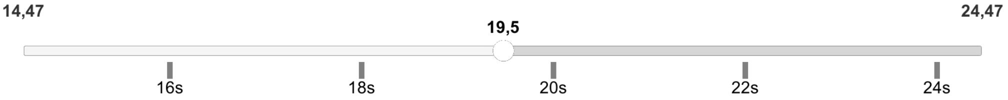
To counteract unwanted positional deviations that can occur because the thumbstick is an elastic device and the user must apply a certain force against an elastic resistance, the resulting values are filtered by a *one-euro filter* [65] before being used for temporal navigation.

Thumbstick Zoom Technique

Based on the idea of the *Orthozoom* scroller of Appert et al. [32], we propose a technique with which the user can dynamically refine the time interval displayed on the time slider around his or her current point in time. To refine the time interval displayed on the time slider, the user can use the right thumbstick as input. By moving the right thumbstick up, the user zooms in on the time slider, effectively reducing the time interval visualised in the time slider. By moving the thumbstick down, the user zooms out, effectively increasing the time interval being displayed on the time slider. The left thumbstick can be used for temporal navigation, where the speed of the temporal navigation is dynamically adjusted according to the current time interval displayed on the time slider. As the speed of the temporal navigation is dynamically adjusted to the current time interval displayed on the time slider, the user can navigate more precisely in time by adjusting the time interval. The modification of the time interval is visualised on the timeline by marking specific points in time on the timeline that are updated automatically, as can be seen in **Figure 4.6**. Similar to the reference thumbstick technique a *one-euro filter* is applied to reduce jitter.



(a) Complete time interval visualised on the time slider.



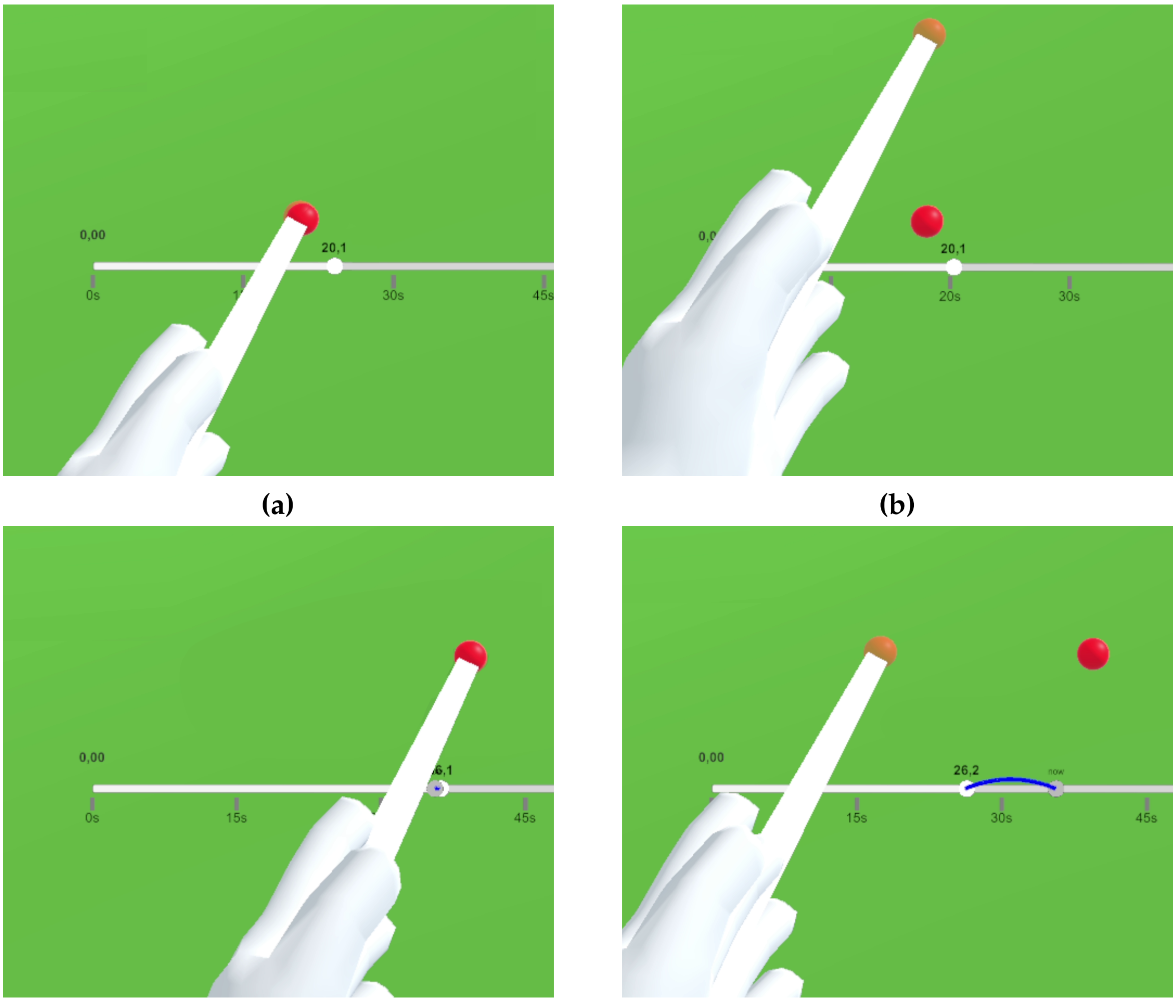
(b) Refined time interval visualised on the time slider.

Figure 4.6: Update of the time interval displayed on the time slider. Note, that depending on the interval range different points in time on the time slider are visualised.

Although this technique allows precise temporal navigation around the current point in time, a modified time interval may not allow users to understand how long the recording is and how their temporal positions relate to the temporal length of the recording.

Raycasting Zoom Technique

Building upon the idea of the *ZoomSlider* proposed by Hürst et al. [29], we propose a raycasting based technique that allows users to navigate in time and to modify the time interval of the time slider through raycasting without the need to select the time slider. To do this, the user can select a point on the invisible plane on which the time slider is positioned in world space through raycasting. Depending on the vertical distance of the point from the time slider, the time interval displayed on the time slider is dynamically updated. The greater the vertical distance of the selected point to the time slider is, the smaller is the time interval that is displayed on the time slider. By dragging the selected point horizontally on the plane, the user is able to navigate forward and backward in time. To reduce the effect of possible hand tremor on the temporal navigation, the resulting change in time is filtered by a *one-euro filter* before it is applied. The modification of the time interval shown on the time slider and the temporal navigation through horizontal dragging is visualised in **Figure 4.7**.



(a) **(b)**

(c) **(d)**

Figure 4.7: Temporal navigation and time scale modification through raycasting. By moving the ray vertically the time interval displayed is modified as can be seen comparing **(a)** and **(b)**. Horizontal movement results in temporal navigation as can be seen by comparing **(c)** and **(d)**.

Time Grab

Similar to the gesture based temporal navigation technique proposed by Petry et al. [64], we propose a novel temporal navigation technique in which the user essentially grabs the handle on the time slider and manipulates it, enabling her or him to navigate through time. By pulling the trigger button of a controller, the temporal navigation is activated as the user essentially grabs the slider handle. Moving the hand to the right while holding the trigger button results in forward temporal navigation in time, while moving the hand to the left corresponds to backward temporal navigation. Here, the temporal distance travelled is defined by a nonlinear mapping of the angle $\theta \in [-90, 90]$ between the user's initial head and hand position upon activation of the time grab and the current position of the hand. A visualisation of the definition of the angle θ is illustrated in **Figure 4.8** and the nonlinear mapping used is given in **Equation 4.5**, where t corresponds to the initial time at which the time grab was activated and t' corresponds to the time to which the user is navigating. To reduce unwanted hand movements caused by hand tremor, the value t' is filtered by a *one-euro filter* before being used for time navigation.

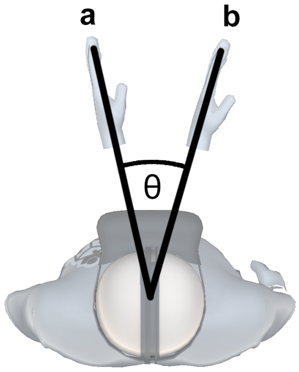
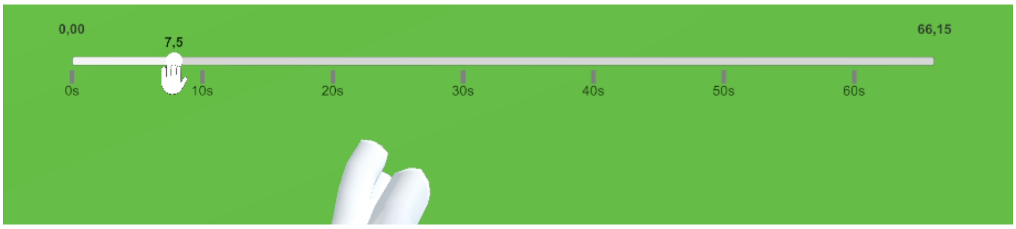


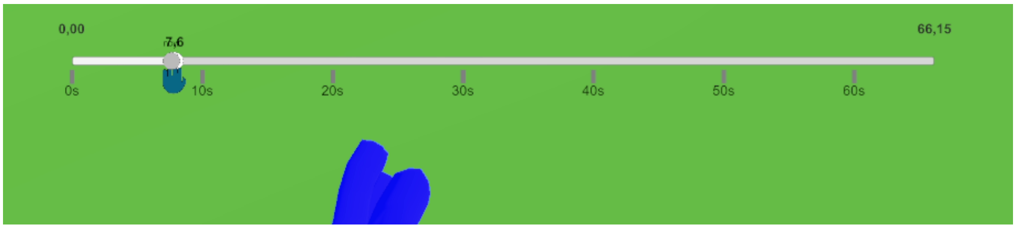
Figure 4.8: Illustration of the definition of the angle θ. The initial hand position is labelled **a**, the current hand position is labelled **b**. Note, that the initial head position, the initial hand position and the current hand position are used to compute θ.

$$t' = \begin{cases} t + \frac{\theta^3}{30^2} & \|\theta\| < 30 \\ t + \theta & \text{else} \end{cases} \tag{4.5}$$

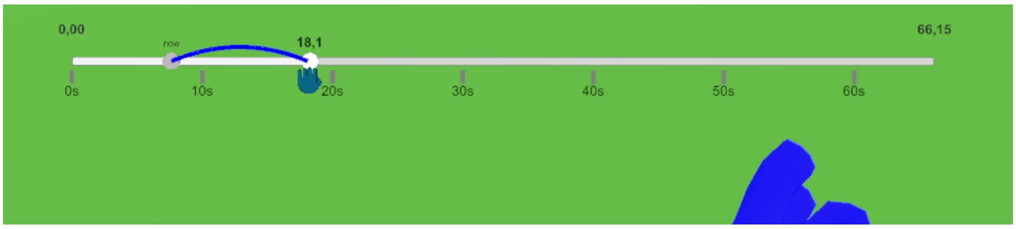
To visualise the influence of the time grab, the difference between the current time and the initial time when the time grab was initialised is visualised on the time slider by a preview arc, as illustrated in **Figure 4.9c**. As the user moves their hand to the right while the time grab is active, the preview arc on the time slider also extends to the right as the time on the time slider increases from left to right. Vice versa, moving the hand to the left results in a preview arc extending to the left on the time slider. In addition, an icon on the time slider and a colouring of the virtual hand of the user is used to inform the user that a time grab is currently active, as is shown in **Figure 4.9b**.



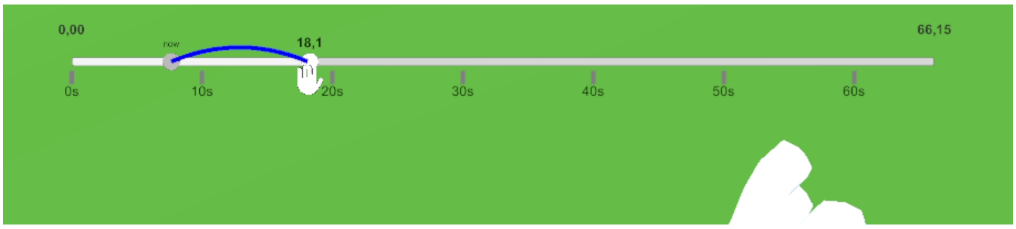
(a) Initial state before the time grab is activated.



(b) The time grab is activated by the user.



(c) Temporal navigation through the time grab.



(d) State after temporal navigation was performed.

Figure 4.9: Sequence of steps during temporal navigation using the time grab. Note, that the hand of the users' avatar is only partially visible at the bottom of the images.

As one can see, the time grab only allows for a restricted temporal navigation in the sense that a single grab can result in a time difference of at most $\frac{90^3}{30^2} = 810$ seconds. For navigation over a greater temporal distance clutching can be used [66].

One advantage of the time grab is that the working space can be defined by the user, which can make temporal navigation more comfortable. Since the angle θ is used to compute the temporal distance travelled, only small hand movements close to the user are sufficient for navigation, and the user is not forced to extend her or his arm. However, if more precise control is required, moving the hand away from the body allows more precise control of the angle θ and thus more precise temporal navigation. A potential disadvantage of this technique is that the user's hand might move when the user moves their head during temporal navigation with the time grab, which can result in unintended temporal navigation.

4.2 Change Awareness

The ability to track asynchronous changes to a workspace, the so-called change awareness, is often a prerequisite for understanding the process that led to the current state of the workspace [16]. During playback of an immersive recording, the speed at which changes of elements in the virtual environment are played back can be controlled by the user through temporal navigation techniques. Rapid continuous temporal navigation, or instantaneous temporal navigation to a different point in time, may cause the user to miss important changes in the scene, affecting the user's change awareness and potentially confusing the user.

Not only temporal navigation but also spatial navigation can affect the user's change awareness in immersive recordings. Since changes in the immersive recording can occur at any position in the virtual environment, changes can occur at points in the virtual environment that are not visible to the user from their current position. When events are not visible to the user his or her change awareness can be affected and similar to omnidirectional video a fear of missing out might be evoked [37].

4.2.1 Analysis of Awareness Techniques

Chow et al. [2] investigated asynchronous collaboration in virtual reality using recordings of virtual environments, and identified a particular challenge for spatial and temporal navigation in guiding the user to locations where important events are taking place. For this reason, Chow et al. argue that temporal navigation should be combined with automatic spatial navigation such that the user can optimally see events of interest to them. Since relevant events are generally not known in advance, we argue that such an approach cannot be applied without prior identification or specification of potentially relevant events. Furthermore, automatic spatial navigation of the user could break immersion.

Guiding the user to events of interest in recordings is part of research related to omnidirectional recordings [35,36,67–69], where the terms *diegetic* and *non-diegetic* are often used to categorise cues used for guidance. Diegetic cues are cues that to some extent are part of the scene, such as a flashing traffic light, which may attract the user's attention and lead the user to a region where events of interest are taking place. In contrast, non-diegetic cues are not part of the scene, such as a forced rotation of the user that leads the user to interesting events, or arrows that point to events of interest.

Investigating diegetic cues in omnidirectional recordings, Rothe et al. [35] found that sound can guide users to objects of interest even if the object is not initially in the user's field of view. Although guidance with spatialized sound worked better than sound without spatialization, Rothe et al. found that non spatialized sound also allowed for guidance to objects that were not initially in the user's field of view.

Studying techniques to alert HMD users of bystanders in order to prevent collisions, Medeiros et al. [70] found that most study participants preferred a combination of visual and auditory cues as a warning of bystanders over combinations of other cues. Moreover, study participants preferred less distracting combinations of cues.

As the virtual environment can abruptly change during direct temporal navigation to a different point in time Chow et al. [2] propose to provide the user with visual indicators of how the environment will change. For a proof of concept, the authors implemented an animated scene transition technique inspired by *Mnemonic Rendering* [71] that identifies objects that changed during temporal navigation and animates a transparent double of the object that is moved from the original state to the new state. This technique visually displays changes in the scene that have occurred over the period of time over which the user has navigated, potentially improving the change awareness of the user. However, in scenes where multiple objects are changing, this technique might lead to visual clutter that could make it difficult for the user to identify individual changes.

Another technique that combines two temporal states of a immersive recording into a single representation was used by Craven et al. [9] for their non-linear drama *Avatar Farm*. During playback of the drama, users can trigger a *time rift* that creates a replay of a previous part of the recording, which is visualised by combining the past state of the scene with the current state of the virtual environment. The two different states are combined by making all objects from the past state transparent, while the objects from the current state are not modified. This technique also provides the user with the ability to compare two different states of the virtual environment with each other and allows for the detection of changes between both states.

4.2.2 Proposed Change Awareness Techniques

Based on the findings of Medeiros et al. [70] that participants prefer a combination of auditory and visual cues as a warning of bystanders over other combinations of different cues and the findings of Rothe et al. [35] that auditory cues allowed for guidance to objects of interest that are not in the field of view of the user, we propose a combination of auditory and visual cues to notify the user of changes in the virtual environment during temporal navigation.

Outlines

Similar to the technique proposed by Lilija et al. [42], which visually highlights all objects in the virtual environment that change in a recording, we propose a technique that visually accentuates all objects that change during temporal navigation. Objects whose position, rotation, or scale change during temporal navigation are highlighted through outlines. As recordings can contain multiple moving objects, using outlines to highlight changes can lead to visual clutter. To alleviate this effect, overlapping outlines are merged such that they are represented by a single outline. An example of this can be seen in **Figure 4.10**. To some extent, the proposed technique for highlighting changes in the scene is similar to a video cut-out technique in which foreground or background objects are extracted from a video and used for further processing e.g. for compositing [23].

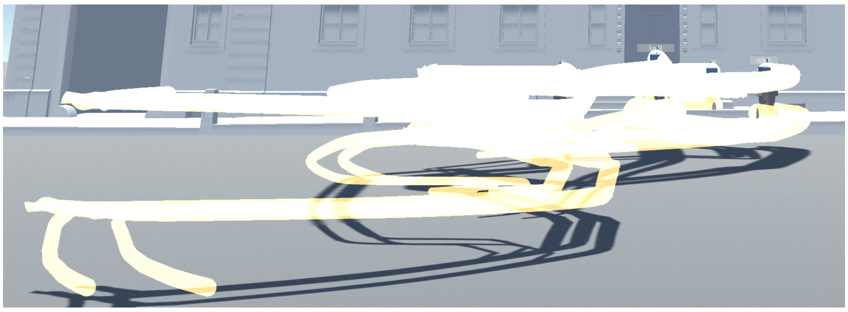
(a) Individual outlines.

(b) Combined outlines.

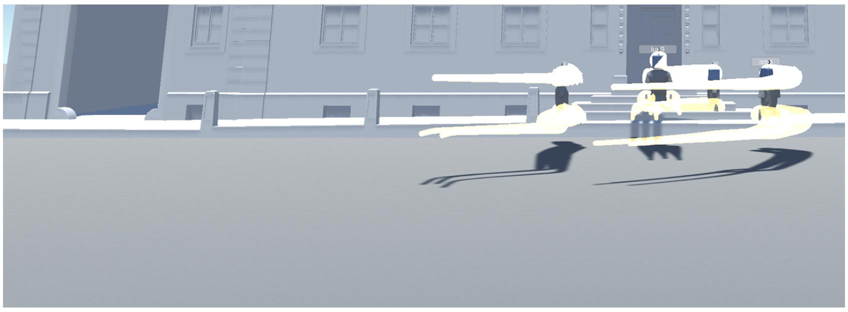
Figure 4.10: Illustration of the difference between individual outlines as can be seen in **(a)** and combined outlines as can be seen in **(b)**.

Spatio-Temporal Trails

Rapid continuous temporal navigation can cause the user to miss events even if they occur within the field of view of the user, e.g. a passing car that is missed because of rapid temporal navigation, which is why we propose the use of spatio-temporal trails to inform the user of such events. As objects move during temporal navigation, spatio-temporal trails that follow the object movements are created. One can see that this potentially results in visual clutter caused by long spatio-temporal trails when navigating through time over a longer duration. For this reason, a trail visualises only the object movements that occurred within the last second of temporal navigation. The effect of limiting the trail length can be seen in **Figure 4.11**.



(a) Unrestricted trail length.



(b) Restricted trail length.

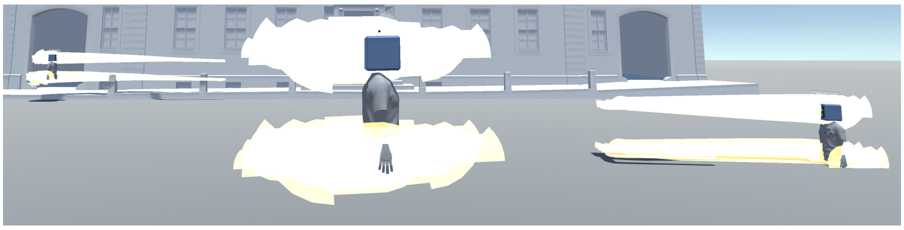
Figure 4.11: Comparison of spatio-temporal trails with no time constraint as seen in **(a)** and trails with time constraint as seen in **(b)** during temporal navigation over a duration of 7 seconds.

Because objects that change during temporal navigation can be close to or far away from the user, spatio-temporal trails with a fixed width can cause important events to be occluded when the trails are close to the user, or not be highlighted to the user at all when the trails are far away. To improve the visibility of distant spatio-temporal trails and reduce the occlusion of events by spatio-temporal trails close to the user, the size of each trail is calculated such that the width of the trail always approximately occupies the same portion of a user's horizontal field of view.

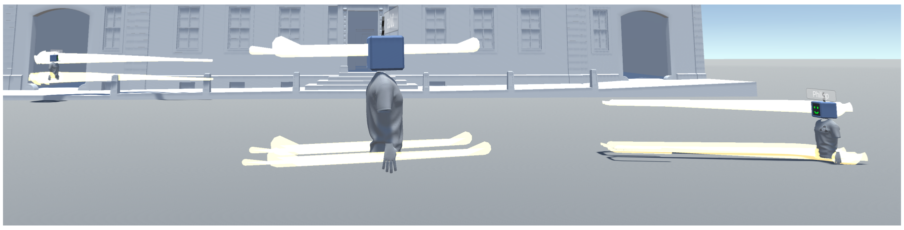
If d is the distance of an object that changes during temporal navigation from the camera, and α is the angle of the user's field of view that the trail width is allowed to occupy, then the trail width w in world space is computed using **Equation 4.6**.

$$w = \left|\tan\left(\frac{\alpha}{2}\right) \cdot d \cdot 2\right| \tag{4.6}$$

The effect of the dynamic trail width modification can be seen in **Figure 4.12**.



(a) Fixed trail width.



(b) Dynamic trail width.

Figure 4.12: Comparison of temporal trails with fixed width as seen in **(a)** and trails with dynamic width as seen in **(b)** during temporal navigation.

Combined Temporal Virtual Environment States

To provide the user with a context of how the virtual environment changed during temporal navigation, we propose a technique which visualises two different temporal states of the virtual environment simultaneously. Our technique builds upon the idea of the *time rift* of Craven et al. [9] and the *preview sphere* proposed by Lilija et al. [42], which to some extend combine two different temporal states of a virtual environment. During temporal navigation, the initial state of the virtual environment is effectively frozen and made transparent, while the state of the virtual environment at the time the user is navigating to is rendered normally. An example of this visualisation is shown in **Figure 4.13**.

Figure 4.13: Combined representation of the two distinct temporal virtual environment states during temporal navigation. The initial state of the virtual environment is visualised transparent.

Combination of Visual Change Cues

To support users in detecting objects that have changed during temporal navigation and to guide them to events that they may have missed during fast temporal navigation, we propose a combination of the proposed outline and trail techniques with the proposed technique for the visualisation of two different temporal states of the virtual environment. An example of this combination of visual change cues is shown in **Figure 4.14**.

Figure 4.14: Combination of the previously introduced visual change cues.

It can be seen that the trails visualise the movement of the objects that have changed in the last few seconds during temporal navigation, whereas the transparent initial state of the virtual environment allows for a comparison of how objects have changed during temporal navigation. The outlines allow for a visual separation between the objects that have changed during temporal navigation and the objects that have not changed.

Auditiory Change Cues

Based on the findings of Rothe et al. [35] that auditory cues are able to guide the user to objects of interest in an omnidirectional video we propose the use of spatialized sounds to inform the user about objects that change during temporal navigation. For this purpose, a generic spatial sound is assigned to all objects which change during temporal navigation complementing the visual change cues. Because potentially many objects around the user can change during temporal navigation, the pitch of each sound is modified to allow the different sounds to be distinguishable.

4.3 Collaborative Exploration of Immersive Recordings

Given the increasing popularity of collaborative and social virtual environments, as reflected for example in the *RecRoom* application with up to three million monthly users [72], the question arises as to how a collaborative exploration of a immersive recording should best be designed. It is clear that in collaborative virtual environments, network synchronicity and workplace coherence, among other things, play an important role in allowing for seamless collaboration. In the following, we explore research on collaborative exploration of immersive recordings and propose techniques to support collaborative exploration of immersive recordings in virtual reality.

4.3.1 Analysis of Collaborative Exploration Techniques

For their non-linear drama *Avatar Farm*, which could be explored by multiple users at the same time, Craven et al. [9] provided users with the ability to make other users invisible if they felt their presence interfered with viewing the recording. Craven et al. identified four aspects influencing user interaction during exploration of immersive recordings, which are time synchronisation, user awareness, communication and shared decision making. For time synchronisation, it is important to determine whether the time of all users in the recording is always synchronised. Regarding user awareness is important to determine the extent to which users are aware of each other and whether they are to some extend able of perceiving each other's viewing experience. As users may be at different points in time within the recording, the extent to which communication between users is made possible is important, as communication across time although helpful for collaboration may lead to confusion as the virtual environment states may be different between users. For negotiation purposes, Craven et al. note that it is important to establish whether users are able to make joint decisions or are able to engage in shared control.

Rothe et al. [68] identified communication, field-of-view awareness, togetherness, accessibility, interaction techniques, synchronisation and multi-user scenarios with more than two users as challenges for social viewing of omnidirectional recordings in virtual reality. To address these challenges, Rothe et al. investigate the usage of a voice chat, a video chat, sending emotes to other users and field of view indication techniques. The results of their study show that the ability to send emotes and the use of the voice chat were found to score the highest for togetherness, whereas the field-of-view indication was found to be the least relevant.

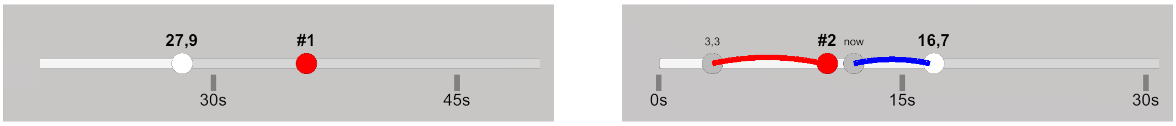
The system of Nguyen et al. [40] for collaborative review of omnidirectional video in virtual reality allows users to navigate independently in time. Users are made aware of their temporal positions by icons that are displayed on the time slider indicating the points in time of all users. Interaction with collaborators is made possible through the use of icons such as *thumbs up*, *thumbs down* and speech, which are displayed on the time slider below icon of the user. Nguyen et al. also enable users to navigate to another user's point in time by selecting the user icon on the time slider and clicking a *Follow in time* button. While the button is visible, a preview of the current part of the recording that the user to be navigated to is seeing is displayed. In addition, a *slave tool* (sic!) is available to the user that visualises the content of the recording that a selected user is seeing on top of the omnidirectional video, so that the user can see the same content as the selected user. To facilitate collaboration, users can create notes associated with time stamps for each other.

4.3.2 Proposed Collaborative Exploration Techniques

In multi-user virtual environments, tools for individual spatial navigation are often made available to users. For collaborative exploration of immersive recordings, we provide each user with the techniques for individual temporal navigation presented earlier.

Temporal Awareness of Collaborators

As individual temporal navigation can lead to users being spread out across time, it is important to ensure that each user knows where in time collaborators are located. To support this, we visualise the points in time for each user on the time slider with icons, similar to Nguyen et al. [40], as can be seen in **Figure 4.15a**. To help the user understand the temporal navigation intentions of their collaborators, the temporal distance covered during temporal navigation of each collaborator is visualised by a preview arc on the time slider, as seen in **Figure 4.15b**.



(a) Without temporal navigation. **(b)** With individual temporal navigation.

Figure 4.15: Time slider in a multi user scenario in which multiple user explore a recording. Note, that in **(b)** the user and the collaborator are navigating through time.

When a collaborator is located at a different point in time than the user, the virtual representation of the collaborator is gradually becoming more transparent depending on the temporal distance between the user and the collaborator. In addition, the collaborator's point in time is visualised above the collaborator's head, as is shown in **Figure 4.16**. The reason for this visualisation method is that a collaborator who is at a different point in time can potentially occlude events of interest from the user if his or her representation is not made transparent. It furthermore allows the user to visually distinguish whether a collaborator is temporally close or far away from the user, without requiring the user to necessarily focus on the time slider.

Figure 4.16: Collaborator becoming gradually more transparent as their temporal distance to the user increases. For this visualisation the reference user was temporally positioned at 27.9 seconds on the time slider. The label of the collaborator shows their temporal position by the numeric value below their nametag as soon as the users do not reside at the same point in time.

Let Δt be the time difference between the user and a collaborator. Then the opacity α is computed using the derivative of the *Sigmoid* function according to **Equation 4.7** such that the collaborator becomes gradually more transparent with increasing time difference but is never completely transparent.

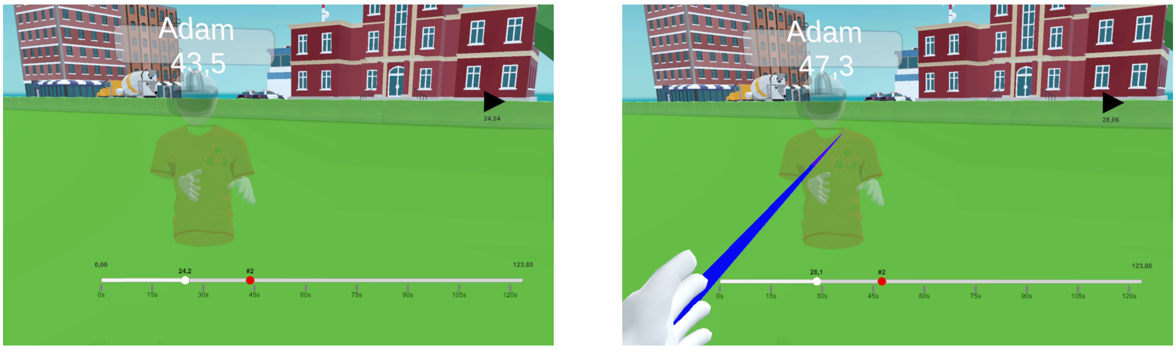
$$\alpha = 3 \cdot \left(\frac{1}{1 + e^{-\Delta t}} \cdot \left(1 - \frac{1}{1 + e^{-\Delta t}} \right) \right) + 0.25 \tag{4.7}$$

Communication via Voice Channel

Rothe et al. [68] identified the importance of audio communication for social viewing of omnidirectional recordings. To support communication between users, we provide users with a spatialized audio chat. Audio communication between users located at different points in time in the recording is made possible in order to support collaboration across time. It can be seen that the extent to which communication is enabled across time, as identified by Craven et al. [9], needs to be taken into consideration as communication across time can potentially result in confusion for users as their virtual environment states may be different. Two users who are at different points in time could be talking about what they like about a car parked on a street corner, without realising that they are talking about two different cars because the original car left at some point in time and a new car parked at the same spot.

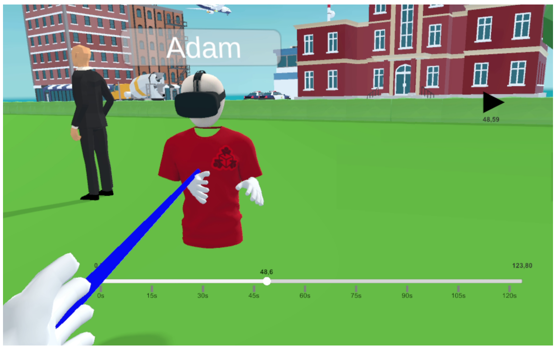
Temporal Navigation to a Collaborator

To enable collaboration in a multi-user environment, it is often important for users to be able to navigate to each other in order to collaborate effectively. To assist users in navigating to the time of a collaborator, we propose a raycasting based direct temporal teleportation technique, in which the user can select a collaborator via raycasting and, after confirming the selection, is teleported to the point in time at which the collaborator resides. The steps for temporal navigation using the proposed technique are illustrated in **Figure 4.17**. As temporal navigation to a collaborator requires a selection of the collaborator via raycasting, this technique requires that the collaborator is visible to the user.



(a) A collaborator with whom the user wants to explore the recording together is at a different point in time.

(b) Through raycasting the user selects the collaborator.



(c) After confirmation of the selection the user is directly teleported to the time of the collaborator.

Figure 4.17: Direct temporal navigation to the point in time of a collaborator using raycasting. Note, that the virtual environment state at the point in time of the collaborator is different to the initial virtual environment state before temporal navigation.

5 Evaluation of Temporal Navigation Techniques

In order to evaluate the usability of a set of the proposed techniques for temporal navigation, a pilot study was conducted. The system investigated in the study includes the introduced visual and auditory change cues, as well as the introduced thumbstick technique for temporal navigation. The aim of the pilot study is to investigate whether the chosen techniques are suitable for temporal navigation. The main focus of the study is on the visual and auditory cues used to support the exploration of immersive recordings, as the artificial change cues may potentially confuse the user and influence the exploration negatively.

5.1 Study Setup

For the study, an Oculus Quest 2 head-mounted display (HMD) and Oculus Quest controller were used. The HMD was connected via an Oculus Link cable to a computer on which an immersive recording was played back. Participants had to answer three questions about events in the immersive recording which happen roughly at the same time but at different spatial locations requiring the participants to use temporal and spatial navigation in order to answer the questions correctly. The immersive recording used in the study was a recording of a city scene that is approximately 2 minutes long and contains over 200 moving objects. The three events that participants had to investigate were (1) a car crash caused by a malfunction of the traffic light system, (2) a present, which is dropped by a helicopter, and (3) the abduction of a sheep by an UFO. A map of the virtual environment indicating the position of the events of interest is illustrated in **Figure 5.1**.

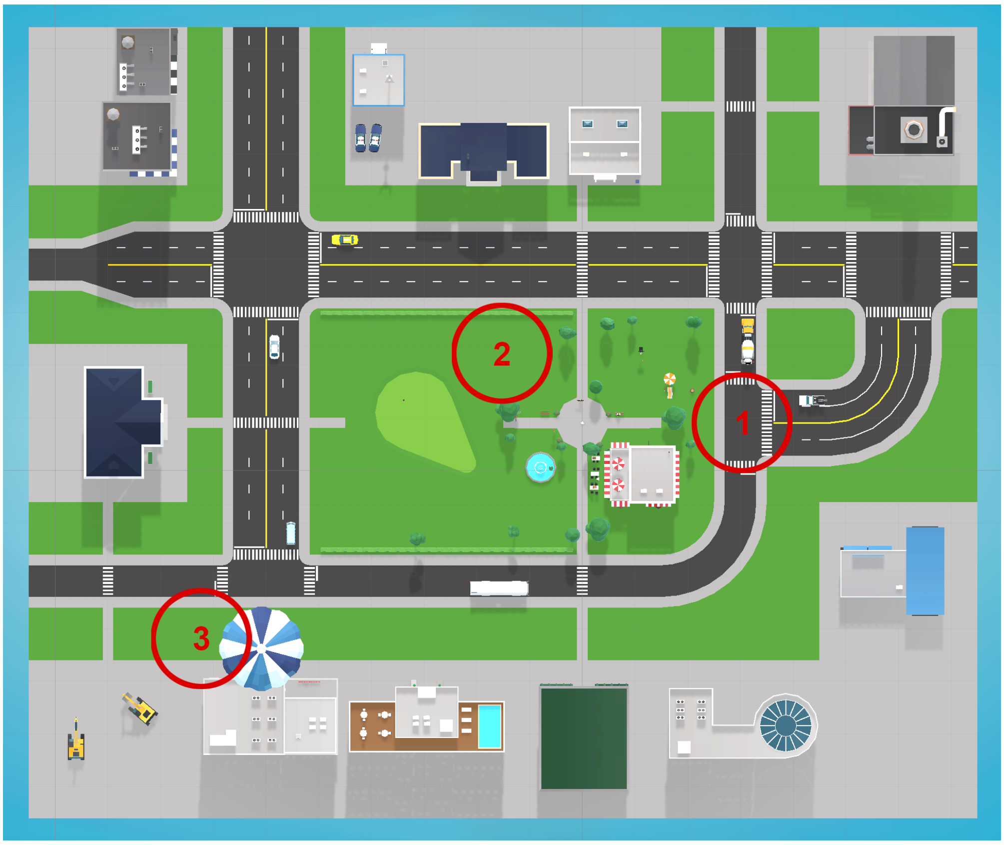


Figure 5.1: Orthographic view from above onto the virtual environment used throughout the study. Red circle indicate the spatial position of the three relevant events. Circle 1 indicates the position of the car crash, circle 2 indicates the position of the present and circle 3 indicates the position of the sheep.

To verify that the events shown in **Figure 5.2** had been explored participants had to identify the cause of the car crash, the place from which the present came from, as well as what happened to the sheep.

(a) Crash seen from the direction of the car. **(b)** Crash seen from the direction of the bus.

(c) Helicopter dropping the present. **(d)** UFO abducting the sheep.

Figure 5.2: Events in the immersive recording that occur roughly at the same time. Note, that in **(a)** and **(b)** both traffic lights are green indicating the malfunction of the traffic light system. The spatio-temporal trails in **(c)** and **(d)** indicate the movement of the sheep and the present.

Participants were able to use steering and teleportation for spatial navigation and the proposed non-linear thumbstick technique for temporal navigation. The number of temporal navigation techniques was limited for the pilot study in order to not overwhelm users without prior experience with virtual reality and to limit the number of potential factors influencing the results of the study. To support change awareness, the visual and auditory change cues proposed throughout **Chapter 4** were used.

Participants were visitors of the *summaery2022* at the Bauhaus-Universität Weimar and received a brief introduction to spatial navigation prior to participating in the study to ensure that the influence of spatial navigation on the results of the study was reduced. For their participation in the study participants were compensated with sweets. Participants were asked to rate the usability of the system using a system usability questionnaire (SUS) and had to answer the three questions about the events happening in the recording. On average, each session lasted 25 minutes, including introduction and filling in the questionnaire.

5.2 Evaluation

A total of 15 participants (male=9, female=6, diverse=0) took part in the study, including 4 participants without prior virtual reality experience, 6 participants with little prior virtual reality experience, and 5 participants with extensive virtual reality experience. 13 of the 15 participants were between 18 and 35 years old, one participant between 35 and 60 years old and one participant over 60 years old.

Of the 15 participants, 12 participants correctly identified the traffic light as the cause of the car accident, while 3 participants identified either the bus or the car as the cause of the accident. All 15 participants were able to correctly identify what happened to the sheep. The question of where the gift in the park came from was answered correctly by 13 participants. One participant incorrectly identified a plane as the place the present came from, while a second participant incorrectly identified a hot air balloon as the place the present came from.

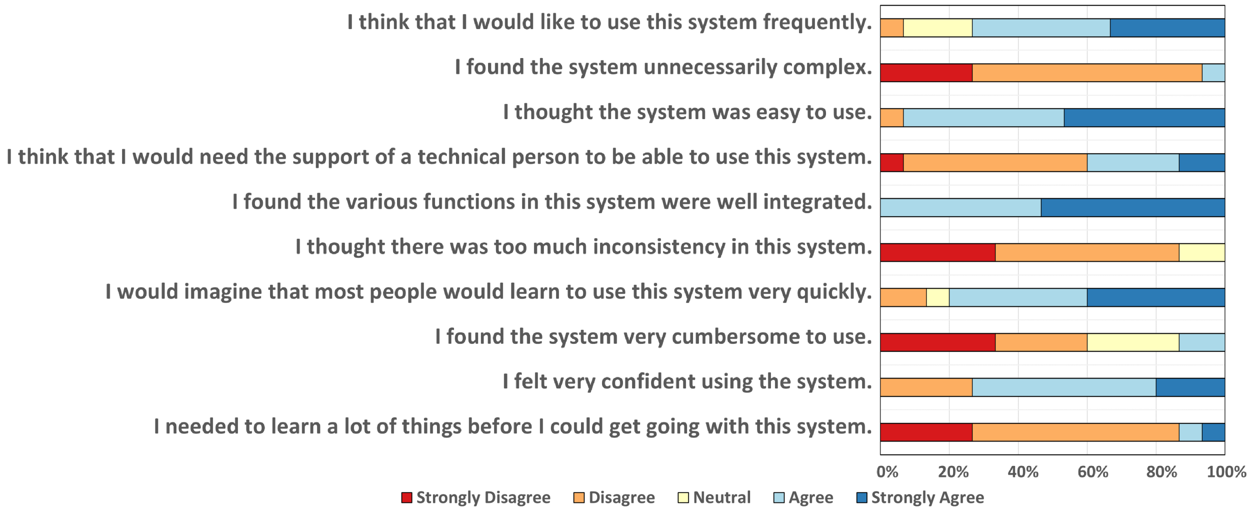


Figure 5.3: Results of the system usability questionnaire.

The average SUS score across all participants is 74.5 ($\sigma = 13,63$), which corresponds to a B grade according to the *Sauro–Lewis curved grading scale* [73]. As illustrated in **Figure 5.3**, the functions of the system were considered to be well integrated. In addition, all but one participant found the system easy to use. However, 6 of the 15 participants felt that they would need the support of a technical person to use the system. As 3 of these 6 people had no previous experience with virtual reality, it may be that the combination of spatial and temporal navigation was potentially too demanding.

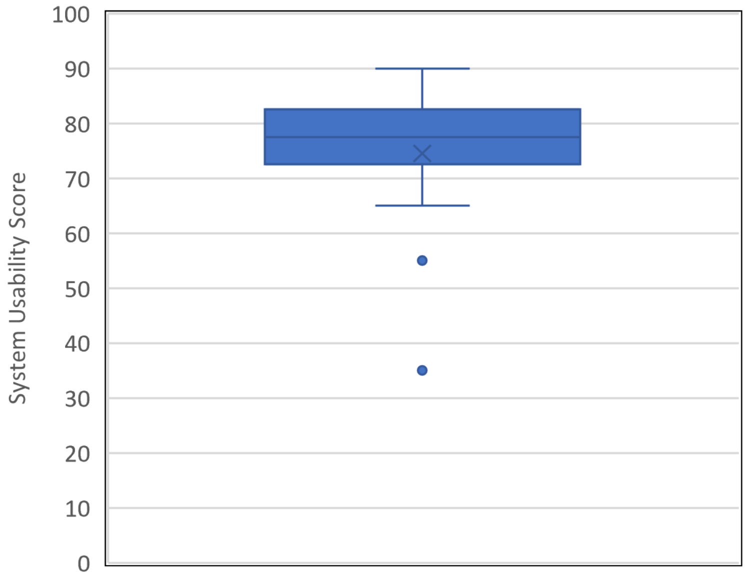


Figure 5.4: Box plot of the individual system usability scores.

As illustrated in **Figure 5.4** most participants rated the system usability with a score between 65 and 90 which corresponds with a grade between C and A+. However, 2 participants rated the system usability with 35 and 55 showing that the system was not rated as well usable by all participants.

When asked what participants liked about the system, several participants noted that they found the system to be intuitive (P4, P6, P12, P14) and easy to use (P2, P3, P6, P12, P13). Furthermore, the visualisation of temporal changes was positively mentioned by participants (P5, P7, P10, P15). One user mentioned that he liked the fun factor of the system (P2).

Being asked what the participants did not like about the system, several participants noted that they had trouble using the spatial navigation technique (P8, P10, P15). One participant commented that the interaction with the time slider was not accurate enough (P5). Regarding temporal navigation one participant commented that the activation of the temporal navigation as soon as the thumbstick of the controller was touched by the user did not work well as time was instead turned back (P14).

Although participants were introduced to spatial navigation prior to the study, it became evident that spatial navigation was a source of problems for some users. It is not possible to determine the extent to which the usability of the spatial navigation technique influenced the usability rating of the system, but although the spatial navigation was a point of critique for some users, the overall usability rating of the system shows that the system is well usable.

Auditory change cues were not mentioned by participants as either positive or negative, indicating that further research is needed on the usefulness of auditory change cues in immersive recordings. In contrast, visual change cues were mentioned positively by several participants.

As only one fixed-length recording was used throughout the conducted study, the results may not be generalisable, but may suggest that visual change cues in the context of immersive recordings can support exploration and potentially improve change awareness. However, further studies would need to be conducted to verify our findings and more precisely determine the factors that positively influenced the usability of our system.

6 Conclusion

Throughout this thesis, we have investigated techniques for temporal navigation and exploration of different types of recordings and proposed novel techniques aimed at assisting the user during temporal navigation and exploration of immersive recordings in single and multi-user virtual reality.

We suggest a combination of visual and auditory change cues to inform the user of changes in the virtual environment that occur in the immersive recording during temporal navigation in order to support their change awareness. Based on existing techniques for temporal navigation of 2D digital video and omnidirectional video, we propose techniques for precise temporal navigation of immersive recordings that are designed to be comfortable to use over an extended period of time by providing an appropriate working space.

To support the collaborative exploration of immersive recordings by multiple users, we propose a set of different techniques. In order to inform the user of the temporal navigation intentions of other users, we propose a technique that visualises the temporal distances covered during temporal navigation on the time slider. For enabling temporal navigation to the point in time of a collaborator, through which a consistent common workspace is created, we propose a raycasting-based direct temporal teleportation technique. To make the user aware of the temporal difference between him and collaborators as well as to limit potential occlusions of relevant events by collaborators located at a different point in time, we propose a technique that makes collaborators gradually more transparent the further away they are temporally located from the user.

We found that a system using a simple technique for precise temporal navigation in combination with visual and auditory change cues was considered to be well usable by the study participants. Although the results of our study are only indicative, as only one immersive recording was used, several study participants rated the proposed visual change cues as a positive addition to the temporal navigation.

In order to develop techniques to support temporal navigation and exploration of immersive recordings, we designed and developed a plugin that allows for performant recording and playback of changes in virtual environments with up to 5,000 simultaneously moving elements. The usefulness of our plugin is indicated by its use in a variety of other projects, e.g. for recording and analysing studies carried out in virtual reality and recording immersive virtual museum tours.

Future Work

As the proposed thumbstick zoom, raycasting zoom and time grab technique offer alternative approaches to temporal navigation, it might be of interested to investigate whether they allow for precise temporal navigation and are well usable. In addition, investigating the usability of the proposed techniques to support collaboration in immersive recordings could provide interesting information on whether additional techniques for efficient collaboration are needed. As symptoms similar to motion sickness can occur during rapid continuous temporal navigation of omnidirectional recordings [38], it might be of interest to investigate the extent to which these symptoms may occur during exploration of immersive recordings and which techniques might be best suited to reduce such symptoms.

When multiple users who want to explore a recording together are able to individually navigate in time, side effects, such as a coordination overhead and the risk of users being spread out across time, might occur. Similar to techniques supporting spatial group navigation, which allow a group of users to stay together during spatial navigation [74], techniques for temporal group navigation might be required in such scenarios to reduce the risk of group users losing each other across time.

In scenarios in which one is working with immersive recordings, it is often desirable to edit those, e.g. by combining different recordings or by removing parts of a recording. Not all techniques used for editing digital 2D video or omnidirectional recordings may be appropriate for editing immersive recordings. For example, a cut in an immersive recording that causes a rapid change in the virtual environment might confuse the user exploring the recording. However, it might be possible to edit the content of immersive recordings with less effort than the content of digital 2D videos. Since immersive recordings contain information on all objects which are present in the recording, one can imagine editing techniques that allow, for example, for the modification of individual objects in the recording. To the best of our knowledge, suitable interactive editing techniques for immersive recordings have only scarcely been explored in the related literature, which is why it is worthwhile to focus on this topic for further research.

Acknowledgements

I would like to show my deep appreciation to my supervisors Adrian Kreskowski and Gareth Rendle who helped me finalize this thesis.

Bibliography

[1] Tomoko Imai, Andrew E. Johnson, Jason Leigh, David E. Pape, and Thomas A. DeFanti. The virtual mail system. In *Proceedings IEEE Virtual Reality (Cat. No. 99CB36316)*, page 78. IEEE Comput. Soc, 1999.

[2] Kevin Chow, Caitlin Coyiuto, Cuong Nguyen, and Dongwook Yoon. Challenges and Design Considerations for Multimodal Asynchronous Collaboration in VR. *Proceedings of the ACM on Human-Computer Interaction*, 3(CSCW):1–24, 2019.

[3] Scott Howie and Marco Gilardi. Virtual Observations: a software tool for contextual observation and assessment of user's actions in virtual reality. *Virtual Reality*, 25(2):447–460, 2021.

[4] Paul McIlvenny. The future of 'video' in video-based qualitative research is not 'dumb' flat pixels! Exploring volumetric performance capture and immersive performative replay. *Qualitative Research*, 20(6):800–818, 2020.

[5] Bruno Fanini and Luigi Cinque. Encoding immersive sessions for online, interactive VR analytics. *Virtual Reality*, 24(3):423–438, 2020.

[6] Michael Nebeling, Maximilian Speicher, Xizi Wang, Shwetha Rajaram, Brian D. Hall, Zijian Xie, Alexander R. E. Raistrick, Michelle Aebersold, Edward G. Happ, Jiayin Wang, Yanan Sun, Lotus Zhang, Leah E. Ramsier, and Rhea Kulkarni. MRAT: The Mixed Reality Analytics Toolkit. In Pierre Dragicevic, Gonzalo Ramos, Jacobo Bibliowitcz, Derek Nowrouzezahrai, Ravin Balakrishnan, and Karan Singh, editors, *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, ACM Digital Library, pages 1–12, New York,NY,United States, 2020. Association for Computing Machinery.

[7] Chris Greenhalgh, Martin Flintham, Jim Purbrick, and Steve Benford. Applications of Temporal Links: Recording and Replaying Virtual Environments. In *Proceedings IEEE Virtual Reality 2002*, pages 101–108. IEEE Comput. Soc, 2002.

[8] Mikhail Fominykh, Andrey Smorkalov, Mikhail Morozov, and Ekaterina Prasolova-Førland. 3D Virtual Worlds as a Fusion of Immersing, Visualizing, Recording, and Replaying Technologies. In Dharmendra Sharma, Margarita Favorskaya, Lakhmi C. Jain, and Robert J. Howlett, editors, *Fusion of Smart, Multimedia and Computer Gaming Technologies*, volume 84 of *Intelligent Systems Reference Library*, pages 137–171. Springer International Publishing, Cham, 2015.

[9] Mike Craven, Bernd Lintermann, Michael Hoch, Ian Taylor, Adam Drozd, Jim Purbrick, Chris Greenhalgh, Steve Benford, Mike Fraser, John Bowers, and Kai-Mikael Jää-Aro. Exploiting interactivity, influence, space and time to explore non-linear drama in virtual worlds. In Julie Jacko, editor, *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ACM Conferences, pages 30–37, New York, NY, 2001. ACM.

[10] Vojtěch Brůža, Jan Byška, Jan Mičan, and Barbora Kozlíková. VRdeo: Creating engaging educational material for asynchronous student-teacher exchange using virtual reality. *Computers & Graphics*, 98:280–292, 2021.

[11] Anthony Steed, Mingqian Wang, and Jason Drummond. Recording and Replaying Virtual Environments for Development and Diagnosis. In William R. Sherman, editor, *VR programing gems*, pages 349–361. CRC Press, Boca Raton and London and New York, 2019.

[12] Fortnite Battle Royale - Replay System. `https://www.epicgames.com/fortnite/en-US/news/fortnite-battle-royale-replay-system`, 2018. (Accessed on 06/09/2022).

[13] Jeff Gipson, Lauren Brown, Ed Robbins, Jose Gomez, Mike Anderson, Jose Velasquez, Jorge Ruiz, and Dan Cooper. VR story production on Disney animation's "cycles". In *ACM SIGGRAPH 2018 Talks*, ACM Conferences, pages 1–2, New York, NY, 2018. ACM.

[14] Haijun Xia, Sebastian Herscher, Ken Perlin, and Daniel Wigdor. Spacetime: Enabling Fluid Individual and Collaborative Editing in Virtual Reality. In Patrick Baudisch, Albrecht Schmidt, and Andy Wilson, editors, *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology*, pages 853–866, New York, NY, USA, 2018. ACM.

[15] André Kunert, Alexander Kulik, Stephan Beck, and Bernd Froehlich. Photoportals: Shared References in Space and Time. In Susan Fussell, editor, *Proceedings of the 17th ACM conference on Computer supported cooperative work & social computing*, ACM Digital Library, pages 1388–1399, New York, NY, 2014. ACM.

[16] James Tam and Saul Greenberg. A Framework for Asynchronous Change Awareness in Collaboratively-Constructed Documents. In David Hutchison, Takeo Kanade, Josef Kittler, Jon M. Kleinberg, Friedemann Mattern, John C. Mitchell, Moni Naor, Oscar Nierstrasz, C. Pandu Rangan, Bernhard Steffen, Madhu Sudan, Demetri Terzopoulos, Dough Tygar, Moshe Y. Vardi, Gerhard Weikum, Gert-Jan de Vreede, Luis A. Guerrero, and Gabriela Marín Raventós, editors, *Groupware: Design, Implementation, and Use*, volume 3198 of *Lecture Notes in Computer Science*, pages 67–83. Springer Berlin Heidelberg, Berlin, Heidelberg, 2004.

[17] Hartmut Luttermann and Manfred Grauer. VRML history: storing and browsing temporal 3D-worlds. In Christian Bauer, Stephan Diehl, and Jean-Francis Balaguer, editors, *Proceedings of the fourth symposium on Virtual reality modeling language*, ACM Conferences, pages 153–160, New York, NY, 1999. ACM.

[18] Chris Greenhalgh, Jim Purbrick, Steve Benford, Mike Craven, Adam Drozd, and Ian Taylor. *Temporal links: recording and replaying virtual environments: Proceedings*. Addison Wesley Longman, Reading, Mass., 2000.

[19] William Steptoe and Anthony Steed. Multimodal Data Capture and Analysis of Interaction in Immersive Collaborative Virtual Environments. *Presence: Teleoperators and Virtual Environments*, 21(4):388–405, 2012.

[20] Marcus. R. Watson, Voloh Benjamin, Thomas Christopher, Hasan Asif, and Womelsdorf Thilo. *USE: An integrative suite for temporally-precise psychophysical experiments in virtual environments for human, nonhuman, and artificially intelligent agents*. 2018.

[21] Ingo Esser. Record, Edit, and Rewind in Virtual Reality with NVIDIA VR Capture and Replay | NVIDIA Technical Blog. `https://developer.nvidia.com/blog/record-edit-and-rewind-in-virtual-reality-with-nvidia-vr-capture-and-replay/`, 2022. (Accessed on 06/13/2022).

[22] Klaus Schoeffmann, Marco A. Hudelist, and Jochen Huber. Video Interaction Tools: A Survey of Recent Work. *ACM Computing Surveys*, 48(1):1–34, 2015.

[23] R. Borgo, M. Chen, B. Daubney, E. Grundy, G. Heidemann, B. Höferlin, M. Höferlin, H. Leitte, D. Weiskopf, and X. Xie. State of the Art Report on Video-Based Graphics and Video Visualization. *Computer Graphics Forum*, 31(8):2450–2477, 2012.

[24] Francis C. Li, Anoop Gupta, Elizabeth Sanocki, Li-wei He, and Yong Rui. Browsing digital video. In Thea Turner and Gerd Szwillus, editors, *Proceedings of the SIGCHI conference on Human factors in computing systems - CHI '00*, pages 169–176, New York, New York, USA, 2000. ACM Press.

[25] Keita Higuchi, Ryo Yonetani, and Yoichi Sato. Egoscanning: quickly scanning first-person videos with egocentric elastic timelines. In Takuji Narumi and Borom Tunwattanapong, editors, *SIGGRAPH Asia 2017 Emerging Technologies*, ACM Digital Library, pages 1–2, New York, NY, 2017. ACM.

[26] Pierre Dragicevic, Gonzalo Ramos, Jacobo Bibliowitcz, Derek Nowrouzezahrai, Ravin Balakrishnan, and Karan Singh. *Video Browsing by Direct Manipulation: The 26th Annual CHI Conference on Human Factors in Computing Systems, April 5-10, 2008 in Florence, Italy conference proceedings*. ACM Conferences. ACM, New York, NY, 2008.

[27] Wolfgang Hürst, Georg Götz, and Philipp Jarvers. Advanced user interfaces for dynamic video browsing. In Henning Schulzrinne, Nevenka Dimitrova, Angela Sasse, Sue Moon, and Rainer Lienhart, editors, *Proceedings of the 12th annual ACM international conference on Multimedia - MULTIMEDIA '04*, page 742, New York, New York, USA, 2004. ACM Press.

[28] Madhur Khandelwal, Andruid Kerne, and J. Michael Mistrot. Manipulating history in generative hypermedia. In Jim Whitehead and David de Roure, editors, *Proceedings of the fifteenth ACM conference on Hypertext & hypermedia - HYPERTEXT '04*, page 139, New York, New York, USA, 2004. ACM Press.

[29] Wolfgang Hürst. Interactive audio-visual video browsing. In Klara Nahrstedt, Matthew Turk, Yong Rui, Wolfgang Klas, and Ketan Mayer-Patel, editors, *Proceedings of the 14th annual ACM international conference on Multimedia - MULTIMEDIA '06*, page 675, New York, New York, USA, 2006. ACM Press.

[30] Yuichi Koike, Atsushi Sugiura, and Yoshiyuki Koseki. TimeSlider: An Interface to Specify Time Point. In George Robertson and Chris Schmandt, editors, *Proceedings of the 10th annual ACM symposium on User interface software and technology - UIST '97*, pages 43–44, New York, New York, USA, 1997. ACM Press.

[31] Heather Richter, Jason A. Brotherton, Gregory D. Abowd, and Khai N. Truong. A Multi-Scale Timeline Slider for Stream Visualization and Control. 1999.

[32] Caroline Appert and Jean-Daniel Fekete. OrthoZoom Scroller: 1D Multi-Scale Navigation. In Rebecca Grinter, Thomas Rodden, Paul Aoki, Ed Cutrell, Robin Jeffries, and Gary Olson, editors, *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 21–30, New York, NY, USA, 2006. ACM.

[33] Don Kimber, Tony Dunnigan, Andreas Girgensohn, Frank Shipman, Thea Turner, and Tao Yang. Trailblazing: Video Playback Control by Direct Object Manipulation. In *Multimedia and Expo, 2007 IEEE International Conference on*, pages 1015–1018. IEEE, 2007.

[34] Frank Rodriguez. Hot and Cold: Heatmaps in VR. `https://blog.youtube/news-and-events/hot-and-cold-heatmaps-in-vr/`, 2017. (Accessed on 05/19/2022).

[35] Sylvia Rothe and Heinrich Hußmann. Guiding the Viewer in Cinematic Virtual Reality by Diegetic Cues. In Lucio Tommaso de Paolis and Patrick Bourdot, editors, *Augmented Reality, Virtual Reality, and Computer Graphics*, volume 10850 of *Lecture Notes in Computer Science*, pages 101–117. Springer International Publishing, Cham, 2018.

[36] Sylvia Rothe, Daniel Buschek, and Heinrich Hußmann. Guidance in Cinematic Virtual Reality-Taxonomy, Research Status and Challenges. *Multimodal Technologies and Interaction*, 3(1):19, 2019.

[37] Shoma Yamaguchi, Nami Ogawa, and Takuji Narumi. Now I'm Not Afraid: Reducing Fear of Missing Out in 360 degree Videos on a Head-Mounted Display using a Panoramic Thumbnail. In *2021 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 176–183. IEEE, 2021.

[38] Cuong Nguyen, Stephen DiVerdi, Aaron Hertzmann, and Feng Liu. Vremiere: In-Headset Virtual Reality Video Editing. In Gloria Mark, editor, *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, ACM Digital Library, pages 5428–5438, New York, NY, 2017. ACM.

[39] Ajoy S. Fernandes and Steven K. Feiner. Combating VR sickness through subtle dynamic field-of-view modification. In *2016 IEEE Symposium on 3D User Interfaces (3DUI)*, pages 201–210. IEEE, 2016.

[40] Cuong Nguyen, Stephen DiVerdi, Aaron Hertzmann, and Feng Liu. CollaVR: Collaborative In-Headset Review for VR Video. In Krzysztof Gajos, editor, *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology*, ACM Digital Library, pages 267–277, New York, NY, 2017. ACM.

[41] Cheng Yao Wang, Mose Sakashita, Upol Ehsan, Jingjin Li, and Andrea Stevenson Won. Again, Together: Socially Reliving Virtual Reality Experiences When Separated. In Pierre Dragicevic, Gonzalo Ramos, Jacobo Bibliowitcz, Derek Nowrouzezahrai, Ravin Balakrishnan, and Karan Singh, editors, *Proceedings*

of the 2020 CHI Conference on Human Factors in Computing Systems, ACM Digital Library, pages 1–12, New York,NY,United States, 2020. Association for Computing Machinery.

[42] Klemen Lilija, Henning Pohl, and Kasper Hornbæk. Who Put That There? Temporal Navigation of Spatial Recordings by Direct Manipulation. In Pierre Dragicevic, Gonzalo Ramos, Jacobo Bibliowitcz, Derek Nowrouzezahrai, Ravin Balakrishnan, and Karan Singh, editors, *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, ACM Digital Library, pages 1–11, New York,NY,United States, 2020. Association for Computing Machinery.

[43] Sebastian Hubenschmid, Jonathan Wieland, Daniel Immanuel Fink, Andrea Batch, Johannes Zagermann, Niklas Elmqvist, and Harald Reiterer. ReLive: Bridging In-Situ and Ex-Situ Visual Analytics for Analyzing Mixed Reality User Studies. In Simone Barbosa, editor, *CHI Conference on Human Factors in Computing Systems*, ACM Digital Library, pages 1–20, New York,NY,United States, 2022. Association for Computing Machinery.

[44] Hunter Quant, Sean Banerjee, and Natasha Kholgade Banerjee. A Virtual Reality Interface for Interactions with Spatiotemporal 3D Data. In Klaus Schoeffmann, Thanarat H. Chalidabhongse, Chong Wah Ngo, Supavadee Aramvith, Noel E. O'Connor, and Yo-Sung Ho, editors, *MultiMedia modeling*, volume 10705 of *Lecture notes in computer science Information systems and applications, incl. internet/web, and HCI*, pages 343–347. Springer, Cham, 2018.

[45] Wolfgang Büschel, Anke Lehmann, and Raimund Dachselt. MIRIA: A Mixed Reality Toolkit for the In-Situ Visualization and Analysis of Spatio-Temporal Interaction Data. In Yoshifumi Kitamura, editor, *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, ACM Digital Library, pages 1–15, New York,NY,United States, 2021. Association for Computing Machinery.

[46] Balasaravanan Thoravi Kumaravel, Cuong Nguyen, Stephen DiVerdi, and Björn Hartmann. TutoriVR: A Video-Based Tutorial System for Design Applications in Virtual Reality. In Stephen Brewster, Geraldine Fitzpatrick, Anna Cox, and Vassilis Kostakos, editors, *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, pages 1–12, New York, NY, USA, 2019. ACM.

[47] Thomas Lopez, Olivier Dumas, Fabien Danieau, Bertrand Leroy, Nicolas Mollet, and Jean-François Vial. A playback tool for reviewing VR experiences. In Morten Fjeld, editor, *Proceedings of the 23rd ACM Symposium on Virtual Reality Software and Technology*, ACM Digital Library, pages 1–2, New York, NY, 2017. ACM.

[48] Simon Kloiber, Volker Settgast, Christoph Schinko, Martin Weinzerl, Johannes Fritz, Tobias Schreck, and Reinhold Preiner. Immersive analysis of user motion in VR applications. *The Visual Computer*, 36(10-12):1937–1949, 2020.

[49] Randolph L. Jackson and Eileen Fagan. *Collaboration and Learning within Immersive Virtual Reality: September 10-12, 2000, San Francisco, CA proceedings of the third International Conference on Collaborative Virtual Environments*. ACM Conferences. ACM, New York, NY, 2000.

[50] Pierre Mahieux, Romain Biannic, Sébastien Kubicki, and Ronan Querrec. SABLIER : a Tangible Interactor to Navigate through Space and Time. In Simone Barbosa, editor, *CHI Conference on Human Factors in Computing Systems*, ACM Digital Library, pages 1–12, New York,NY,United States, 2022. Association for Computing Machinery.

[51] Marc Wolter, Irene Tedjo-Palczynski, Bernd Hentschel, and Torsten Kuhlen. Spatial Input for Temporal Navigation in Scientific Visualizations. *IEEE Computer Graphics and Applications*, 29(6):54–64, 2009.

[52] Marc Wolter, Bernd Hentschel, Irene Tedjo-Palczynski, and Torsten Kuhlen. A direct manipulation interface for time navigation in scientific visualizations. In *2009 IEEE Symposium on 3D User Interfaces*, pages 11–18. IEEE, 2009.

[53] Laurie M. Wilcox, Robert S. Allison, Samuel Elfassy, and Cynthia Grelik. Personal space in virtual reality. *ACM Transactions on Applied Perception*, 3(4):412–428, 2006.

[54] Replay System. `https://docs.unrealengine.com/4.27/en-US/TestingAndOptimization/ReplaySystem/`, 02.06.2022. (Accessed on 06/02/2022).

[55] Awni Hannun, Carl Case, Jared Casper, Bryan Catanzaro, Greg Diamos, Erich Elsen, Ryan Prenger, Sanjeev Satheesh, Shubho Sengupta, Adam Coates, and Andrew Y. Ng. Deep Speech: Scaling up end-to-end speech recognition.

[56] Philip DeCamp, George Shaw, Rony Kubat, and Deb Roy. An Immersive System for Browsing and Visualizing Surveillance Video. In Alberto del Bimbo, editor, *Proceedings of the international conference on Multimedia*, ACM Conferences, page 371, New York, NY, 2010. ACM.

[57] Gonzalo Ramos and Ravin Balakrishnan. Zliding: Fluid Zooming and Sliding for High Precision Parameter Manipulation. In Patrick Baudisch, Mary Czerwinski, and Dan Olsen, editors, *Proceedings of the 18th annual ACM symposium on User interface software and technology*, pages 143–152, New York, NY, USA, 2005. ACM.

[58] Wolfgang Hurst and Philipp Jarvers. Interactive, Dynamic Video Browsing with the Zoomslider Interface. In *2005 IEEE International Conference on Multimedia and Expo*, pages 558–561. IEEE, 2005.

[59] Paul M. Fitts. The information capacity of the human motor system in controlling the amplitude of movement. 1954. *Journal of experimental psychology. General*, 121 3:262–269, 1992.

[60] Johnny Accot and Shumin Zhai. Refining Fitts' law models for bivariate pointing. In Gilbert Cockton and Panu Korhonen, editors, *Proceedings of the conference on Human factors in computing systems - CHI '03*, page 193, New York, New York, USA, 2003. ACM Press.

[61] Johnny Accot and Shumin Zhai. Beyond Fitts' law: Models for Trajectory-Based HCI Tasks. In Steven Pemberton, editor, *Proceedings of the ACM SIGCHI Conference on Human factors in computing systems*, pages 295–302, New York, NY, USA, 1997. ACM.

[62] Ferran Argelaguet and Carlos Andujar. A survey of 3D object selection techniques for virtual environments. *Computers & Graphics*, 37(3):121–136, 2013.

[63] Antonin Cheymol, Gwendal Fouche, Lysa Gramoli, Yutaro Hirao, Emilie Hummel, Mae Mavromatis, Yann Moullec, Ferran Argelaguet, and Florian Nouviale. The Rubber Slider Metaphor: Visualisation of Temporal and Geolocated Data. In *2022 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW)*, pages 904–905. IEEE, 2022.

[64] Benjamin Petry and Jochen Huber. Towards effective interaction with omnidirectional videos using immersive virtual reality headsets. In Suranga Nanayakkara, Ellen Yi-Luen Do, Jun Rekimoto, Jochen Huber, and Bing-Yu Chen, editors, *Proceedings of the 6th Augmented Human International Conference*, pages 217–218, New York, NY, USA, 2015. ACM.

[65] Géry Casiez, Nicolas Roussel, and Daniel Vogel. 1euro Filter: A Simple Speed-based Low-pass Filter for Noisy Input in Interactive Systems. In Joseph A. Konstan, Ed H. Chi, and Kristina Höök, editors, *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 2527–2530, New York, NY, USA, 2012. ACM.

[66] Mathieu Nancel, Daniel Vogel, and Edward Lank. Clutching Is Not (Necessarily) the Enemy. In Bo Begole, Jinwoo Kim, Kori Inkpen, and Woontack Woo, editors, *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pages 4199–4202, New York, NY, USA, 2015. ACM.

[67] Chong Cao, Zhaowei Shi, and Miao Yu. Automatic Generation of Diegetic Guidance in Cinematic Virtual Reality. In *2020 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 600–607. IEEE, 2020.

[68] Sylvia Rothe, Alexander Schmidt, Mario Montagud, Daniel Buschek, and Heinrich Hußmann. Social viewing in cinematic virtual reality: a design space for social movie applications. *Virtual Reality*, 25(3):613–630, 2021.

[69] Nahal Norouzi, Gerd Bruder, Austin Erickson, Kangsoo Kim, Jeremy Bailenson, Pamela Wisniewski, Charlie Hughes, and Greg Welch. Virtual Animals as Diegetic Attention Guidance Mechanisms in 360-Degree Experiences. *IEEE Transactions on Visualization and Computer Graphics*, 27(11):4321–4331, 2021.

[70] Daniel Medeiros, Rafael dos Anjos, Nadia Pantidi, Kun Huang, Mauricio Sousa, Craig Anslow, and Joaquim Jorge. Promoting Reality Awareness in Virtual Reality through Proxemics. In *2021 IEEE Virtual Reality and 3D User Interfaces (VR)*, pages 21–30. IEEE, 2021.

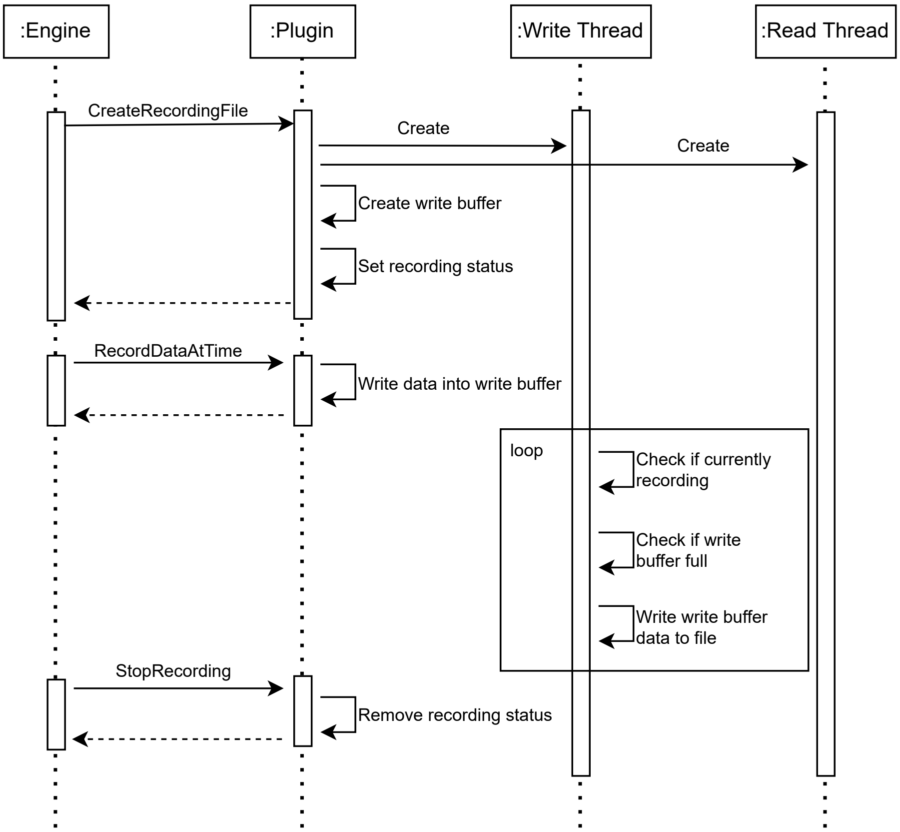
[71] Anastasia Bezerianos, Pierre Dragicevic, and Ravin Balakrishnan. Mnemonic Rendering: An Image-Based Approach for Exposing Hidden Changes in Dynamic Displays. In Pierre Wellner and Ken Hinckley, editors, *Proceedings of the 19th annual ACM symposium on User interface software and technology - UIST '06*, page 159, New York, New York, USA, 2006. ACM Press.

[72] Jamie Feltham. Rec Room Passes 3 Million Monthly Active VR Users. `https://uploadvr.com/rec-room-3-million-vr-users/`, 2022. (Accessed on 04/14/2022).

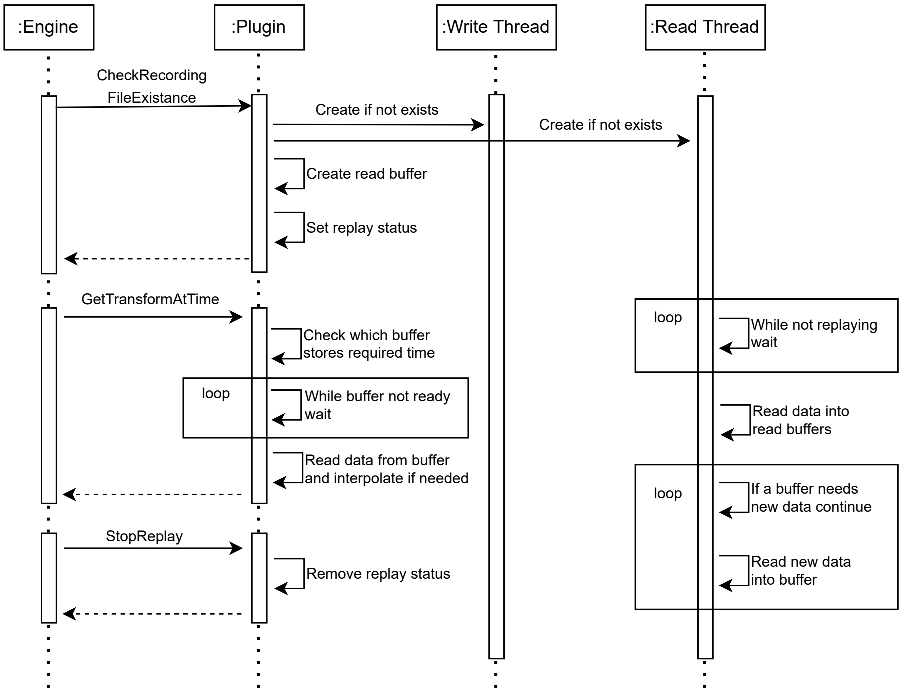
[73] James R. Lewis. The System Usability Scale: Past, Present, and Future. *International Journal of Human–Computer Interaction*, 34(7):577–590, 2018.

[74] Tim Weissker, Pauline Bimberg, and Bernd Froehlich. An Overview of Group Navigation in Multi-User Virtual Reality. In *2021 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW)*, pages 363–369. IEEE, 2021.

A Appendices



(a) Multi-threaded writing.



(b) Multi-threaded reading.

Figure A.1: Simplified UML sequence diagram of the multi-threaded recording and playback behaviour.