Causality-preserving Asynchronous Reality

Andreas Rene Fender & Christian Holz
Department of Computer Science, ETH Zürich
Zurich, Switzerland

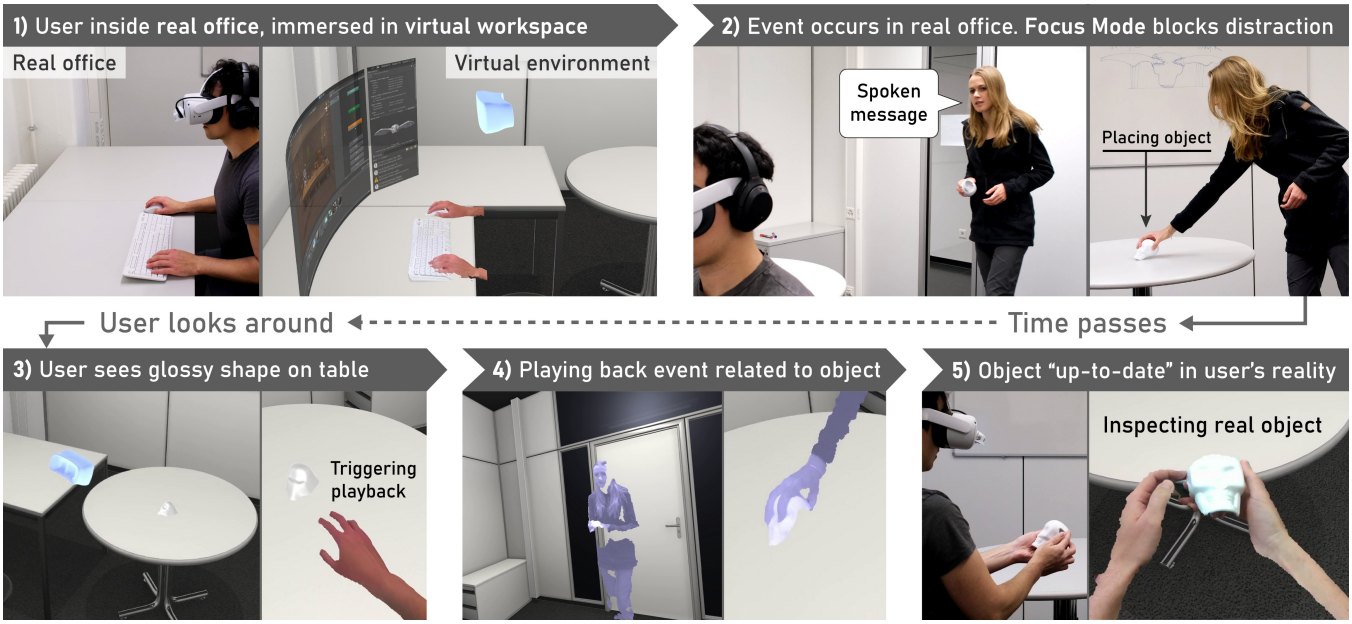

Figure 1: Our AsyncReality system volumetrically captures physical events and allows immersed users to experience those events in an asynchronous yet causally-accurate manner. (1) The user is immersed inside a virtual replica of his real office and uses his physical mouse and keyboard to interact with virtual displays. (2) A coworker enters the room and leaves a spoken message while placing an object on the table, but the user had activated *Focus Mode*, so AsyncReality conceals any visual and auditory sensations from him. (3) After he completes his task, he discovers an artifact on the table and approaches it, which (4) triggers the playback of the captured event. (5) When the playback finishes, the user can interact with the real object.

ABSTRACT

Mixed Reality is gaining interest as a platform for collaboration and focused work to a point where it may supersede current office settings in future workplaces. At the same time, we expect that interaction with physical objects and face-to-face communication will remain crucial for future work environments, which is a particular challenge in fully immersive Virtual Reality. In this work, we reconcile those requirements through a user's individual *Asynchronous Reality*, which enables seamless physical interaction across time. When a user is unavailable, e.g., focused on a task or in a call, our approach captures co-located or remote physical events in real-time, constructs a causality graph of co-dependent events, and lets immersed users revisit them at a suitable time in a causally accurate way. Enabled by our system AsyncReality, we present a workplace scenario that includes walk-in interruptions during a person's focused work, physical deliveries, and transient spoken messages. We then generalize our approach to a use-case agnostic concept and system architecture. We conclude by discussing the implications of an Asynchronous Reality for future offices.

CCS CONCEPTS

• **Human-centered computing** → **Interactive systems and tools**; **Mixed / augmented reality**.

KEYWORDS

Asynchronous communication, Mixed Reality, Collaboration, Immersive workspaces, Camera networks, Interruption in workplaces

ACM Reference Format:
Andreas Rene Fender & Christian Holz. 2022. Causality-preserving Asynchronous Reality. In *CHI Conference on Human Factors in Computing Systems (CHI '22), April 29-May 5, 2022, New Orleans, LA, USA.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3491102.3501836

CHI '22, April 29-May 5, 2022, New Orleans, LA, USA

ACM ISBN 978-1-4503-9157-3/22/04…$15.00
https://doi.org/10.1145/3491102.3501836

1 INTRODUCTION

Mixed Reality (**MR**) has reached the mainstream market and is capable of creating immersive worlds and a strong sense of presence. Today, such technologies can also be found outside controlled laboratories, i.e., in users' homes and offices [13] and even in collaborative multi-user settings [14]. Collaborative interfaces based on Augmented Reality (**AR**) work particularly well when sharing physical space while interacting with virtual contents, e.g., in the same space and time [1] or in the same space, but at a different time [34]. Virtual Reality (**VR**) enables collaborators to interact in a shared virtual environment (**VE**), e.g., to communicate across space and time [5]. Combining such VR technology with RGB-D camera networks [15, 33, 66] creates intriguing opportunities for incorporating the physical space around the user [23] or even perceptually modifying it [37]—enabling new types of reality.

In this paper, we explore the *Asynchronous Reality* concept, a physically-accurate perception of the real world that modifies the user's perceived flow of time. In such a reality, we can control *when* a *real* event in the physical world happens. This way, users can mute real events (e.g., when being immersed in a task) and delay them to a suitable time in their workflow. Figure 1.1 shows an example of an Asynchronous Reality. Here, an immersed user is working on a task. He activates *Focus Mode*, which causes our system to block all external sensations and thus potential distractions—including events such as a person delivering an object (Figure 1.2). For the immersed user, such an asynchronous event only unfolds when approaching the object (Figure 1.3). Importantly, by playing back the event (Figure 1.4), we preserve all the transient information tied to the delivered object, e.g., spoken comments or instructions by the person delivering the object. After experiencing the past event, the object fully enters the user's subjective reality, thus his reality is synchronized now (Figure 1.5).

From asynchronous *messages* to an asynchronous *reality*

Much of human communication has already been asynchronous before modern technological advancements. Outside face-to-face conversations, we have exchanged letters or used other message transportation services. Only with the advent of technology, such as the telegraph and later the telephone, have synchronous means of remote communication increasingly replaced asynchronous messages. From that point on, asynchronous communication has been a fallback option—not because of technical limitations, but rather due to the unavailability of the communication partner at the moment of initiating communication. A prominent example in early modern society is the phone answering machine. Developments of *synchronous* communication technologies over a distance have flourished since. Group calls, video teleconferencing, or even immersive VR solutions all aim to preserve as many characteristics from face-to-face communication as possible, creating the feeling of co-located presence and communication (i.e., the peer is 'here'). In contrast, *asynchronous* communication has not become more natural in comparison. Even in the digital age, asynchronous communication is entirely *explicit*. That is, the sender explicitly creates and submits an e-mail, a video message, or a voice recording. However, an advantage is that receivers may choose *when* to process messages by explicitly opening them or—to avoid being disturbed—they can choose to mute notifications altogether. Utilizing this advantage of asynchronous messages, we argue that the key challenge is to make such asynchronous communication—no matter whether over a distance or in the same space—feel contemporaneous (i.e., the peer is 'now'). During daily interaction in the physical world, 'muting' events or delaying their reception until a later point is not possible. Conversations in an office are inevitably synchronous, leaving little room for implementing the tools we have grown used to in the digital counterparts. While co-located social interaction will always be important (likely also in future office environments), unintended interruptions between collaborators can be counterproductive and are hence worth mitigating [6, 7, 49, 60]. For the purpose of physical-digital realities, we thus see much potential in immersive technologies to bring the advantages of asynchronous communication to physical space.

With an Asynchronous Reality, we address the unavailability of the communication peer (e.g., user in a call or focused on a task) by seamlessly switching to asynchronous communication without changing the communication modality. We explore a future in which immersive technology will have matured enough for productivity tasks, while interaction with physical objects and face-to-face communication will remain as relevant as today. We investigate how to turn the seeming contradiction between immersion and physical interaction into an opportunity that enables seamless interaction across space and time. By creating a reality that incorporates the physical space, but is *asynchronous* we bring the capabilities of digital communication tools to real-world situations. We achieve this through a multi-RGB-D camera network system and several image processing steps to detect events as well as to trace and segment changes in the room between detected events.

The contributions of our work are two-fold. First, we present *Asynchronous Reality* as a general **concept**. Second, we implemented a **system** called *AsyncReality* as an instance of our concept. We illustrate our immersive system in a scenario that takes place in a holistically captured office room. We then present the key part of our system's architecture and implementation—a camera-based method that automatically detects events inside the space, records spatial modifications, and derives their causal relationships. Our method thereby generalizes beyond the illustrated use-cases. In the remainder of this paper, *Asynchronous Reality* refers to the general concept and *AsyncReality* to our system (including architecture, implementation and specific scenario).

2 RELATED WORK

Our work builds upon previous research in RGB-D camera networks, Mixed Reality and the combination of those technologies for creating collaborative interfaces as well as new types of reality.

2.1 Collaboration in MR

MR interfaces are gaining interest as future technologies for productivity ranging from low-level input tasks [41] to collaboration [5, 26]. General CSCW research has long been concerned with classifying collaborative systems [54]. For discussing collaboration and general multi-user interaction in MR, we will coarsely follow the Time-Space Matrix [12, 29, 51], which contains the two orthogonal dimensions *place* and *time* (each either same or different), i.e., combinations of co-located/remote and synchronous/asynchronous.

2.1.1 Synchronous interaction. Several previous systems were designed for synchronous collaboration in MR [35, 52, 67]. An early example of a *synchronous—co-located* MR interface is *Shared space* by Billinghurst et al. [1] with which two AR users could interact with the same virtual content in the same physical space. *ShareVR* [22] allowed two users to interact synchronously but asymmetrically in the same VE, with one user experiencing it in VR and the other via projection mapping. *One Reality* [55] combined various display technologies (hand-held AR displays, HMDs) to seamlessly transition between physical and virtual interaction while collaborating in a shared physical space. In *Slice of Light* [62], realities of different co-located VR users were spatially separated, such that one of the VR users (e.g., with a teacher role) could physically walk in and out of the realities of the other VR users (e.g., student roles).

2.1.2 Asynchronous interaction. A common basis for asynchronous systems is the ability to record and play back users' actions. Lopez et al. [38] presented research on recording and (immersive) playback of VR sessions. Knierim et al. [32] present a system that visually slows down time for the user and then speeds up time again to catch up with reality. Lilija et al. [36] presented an approach that allows to use spatial information to navigate immersive animations. Using similar recording and playback capabilities (and/or the ability to annotate physical objects [26]), researchers previously demonstrated the potential of asynchronous collaboration in MR. Chow et al. [5] presented a VR system that enables recording of messages that include audio together with deictic gestures (e.g., to point to virtual objects) within a purely virtual space so that it can be played back at a later point by a collaborator. Systems for authoring MR instructions use similar concepts [25, 63]. Most related are systems that record instructions ad-hoc without any post production. For example, Lee et al. [34] enabled users to record first-person videos and create spatial cues so as to be played back by a different AR user in the same space but at a different time.

In our implementation, we record real-time point clouds using RGB-D cameras to play them back in an immersive setup following causality-based events. While our motivation for recording and playback is similar to previous work, the main challenge in Asynchronous Reality is not the playback itself, but *how* and *when* to trigger event playback and in which order to play back the events.

2.2 MR and the physical environment

3D reconstruction in a broader sense has been used for a long time to generate virtual representations of real scenes. An early example is *Virtualized Reality* by Kanade et al. [30]. With the increasing availability and quality of RGB-D cameras, many researchers have investigated high-quality reconstruction approaches [8, 9, 28, 72]. Previous works also utilized 3D reconstruction techniques to render parts [50] or all of the physical environment, e.g., for telepresence [19]. With *Mixed Voxel Reality* [53], Regenbrecht et al. investigated the combination of VR devices and live point cloud rendering in terms of user experience. Lindlbauer et al. presented *RemixedReality* [37], which reconstructs the whole physical environment live. The immersed user sees a version of the whole room with several space-time modifications applied (e.g., moved or copied objects). The system features immersive recording and playback of point clouds, yet is oblivious to individual people, objects, their interaction, or events. *SpaceState* [16] uses point clouds to identify pre-defined static states of the room to adapt projection mapping contents to the current usage of the space. However, the system has no knowledge about causal relationships between those states.

With *Substitutional Reality* (**SR**), VEs match the layout of the physical environment, but with different textures and details to match the virtual setting stylistically (and/or particular physical objects are replaced with similarly shaped virtual objects). Simeone et al. [57] presented the SR concept and tested the effect of different virtual alterations of physical props on the VR user experience. Other projects scanned the environment to automatically generate an SR environment, such as a stylistically altered virtual representation for the immersed user [56, 58, 59] also in a mobile context [4, 68]. Hettiarachchi et al.'s concept of *Annexing Reality* [24] is closely related to SR. They scan the user's immediate environment to find geometrically matching physical objects to be used as proxies for tangible interaction in VR. *TransforMR* [31] visually replaces real-world objects and people with animated virtual counterparts with the goal of maintaining semantically correct behavior.

Rather than altering physical space, the goal of our work is preserving individual spatial entities and replaying inter-object behavior at a later point. Therefore, we use a simplified replication of a real office in our examples (Figure 1.1 and Figure 2) and process objects moving inside, but our concept is also compatible with visual alterations as presented in SR or closely related approaches.

2.2.1 Awareness and interruptions. Goerge et al. [21] found out that people in a fully immersive setup feared disengagement from the real world. One goal is therefore to increase *awareness*, so that users retain their ability to interact with physical objects and to communicate with co-located people, possibly without sacrificing immersion. In *A Dose of Reality* [40], McGill et al. rendered portions of the real environment inside the VE (similar to Augmented Virtuality in the taxonomy of Milgram et al. [43]), so that VR users can still interact with physical objects and bystanders. The *RealityCheck* system by Hartmann et al. [23] rendered the physical space around VR users directly into commercial VR games. A goal that is in some ways contrary to increasing awareness has been to *prevent interruptions*, e.g., from collaborators [7] or more generally blocking distractions from the physical environment. For instance, Goerge et al. investigated interruptions of VR users by bystanders [20].

Our Asynchronous Reality concept combines both goals (awareness and preventing interruptions) depending on the situation. The immersed user is *either* fully aware of the physical surroundings (i.e., we render people and objects live inside the VE), *or* we block distractions and prevent interruptions (resorting to asynchronous communication) whenever desired.

2.2.2 Causality detection. The playback component of our system primarily focuses on causality. Nancel et al. [46] utilized causality for improving undo/redo operations in 2D UIs. In MR, the laws of causality can be utilized [3] or intentionally altered [39]. In contrast, our goal is to create a reality that is asynchronous, but still obeys the causal relationships from the real world. There are several vision-based methods for detecting causalities [17, 18, 47, 69]. For instance, Brand et al. [2] analyzed causalities in video feeds to identify key frames of causal events. For our AsyncReality implementation, we use RGB-D cameras to detect events and causalities.

2.3 Summary, positioning, and goals

In contrast to traditional MR systems that focus on adding virtual content to reality or on fully immersing the user, there is a recent trend of creating new realities by incorporating the physical environment in different ways as outlined above. Our work combines those techniques to create the concept of a purely *asynchronous* reality, which conceptually does not add virtuality[1], but only changes the perceived occurrence of real events while obeying real causality. Combining VR technology with RGB-D cameras is currently the most suitable way of demonstrating our concept. However, our time-focused alteration of reality makes it difficult to unequivocally position this concept within classical taxonomies such as the virtuality continuum by Milgram et al. [43]. We thus consult the *RemixedReality* design space by Lindlbauer and Wilson [37], who define four high-level *dimensions of modifications*: Spatial, Temporal, Viewpoint and Appearance. Asynchronous Reality can be primarily positioned within the Temporal modification and in particular within the Temporal.playback modification. By using different color tints (sub-modification Appearance.recolor), AsyncReality explicitly communicates, which events are remote and/or from the past (e.g., blue tint in Figure 1.4). Hence, the goal of an Asynchronous Reality is not to 'deceive' the user by pretending that all interactions happen here and now. Instead, the goal is that those asynchronous events are seamlessly integrated into the user's reality with many qualities of synchronous communication preserved. This can be compared to conventional video conferencing versus voice-only communication: Even though users are aware that they are not co-located, they still use body language and seek eye contact for an overall better communication, i.e., we still act as if we would talk face-to-face. Similarly, Asynchronous Reality is about making communication across *time* seamless.

[1]Note that we also display virtual content in our examples (e.g., the virtual screen in Figure 1.1), but those are for context and illustrative purposes rather than part of the core concept of an Asynchronous Reality.

Figure 2: A virtual replication of the office in our scenario. Instead of the real office, the user sees this reduced virtual representation. Depending on the system state, we augment this static replication with virtual content as well as parts of real-time point clouds (live or recorded).

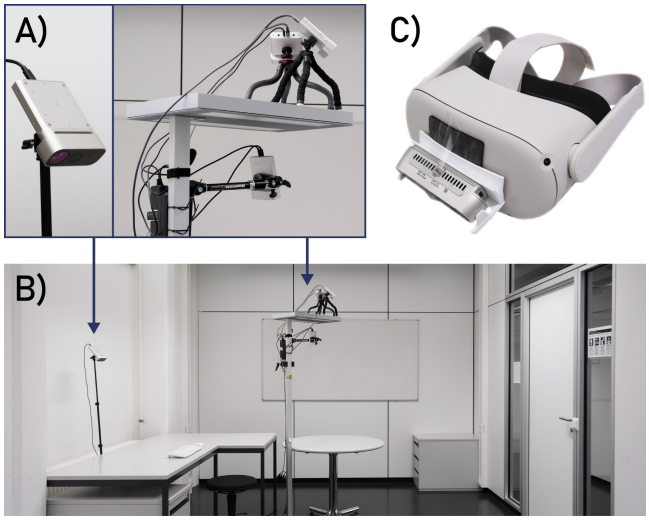

Figure 3: Scenario apparatus. We installed four Azure Kinect cameras (A) in an office (B). Mouse and keyboard serve as user input (see desk). The user wears an Oculus Quest 2 (C) with a RealSense D435 camera mounted to it (slightly angled downwards to capture hands and objects).

3 SCENARIO

Before formalizing the Asynchronous Reality concept, we describe a specific scenario that we acted out with our AsyncReality prototype implementation. The scenario showcases a transition from a synchronous to an asynchronous collaboration style. We encourage the reader to additionally watch the *Scenario* video inside the 'Supplementary Materials' of this publication because the scenario contains many parts that are easier to understand in motion.

3.1 Scenario apparatus

In this subsection, we outline the specific components and virtual assets we used for our scenario. We set up an RGB-D camera network (Figure 3.A) in one of our offices (Figure 3.B). In total, we installed four static Azure Kinect cameras to capture real-time point clouds of different parts of the room: the entrance, the table, the whiteboard and the user's desk. For output, we use an Oculus Quest 2 (Figure 3.C). We attached a RealSense D435 camera to it so as to capture additional live point cloud data from the user's point of view (e.g., when interacting with hand-held objects). As a basis for our VE, we created a visually reduced virtual replication of the office, which contains static furniture items, such as tables and drawers (Figure 2). This way, there is considerably less camera coverage required to create the impression of a self-contained room without gaps along the surfaces. While remapping techniques for navigation would be compatible with our approach [44, 48, 71], we simply use a 1:1 mapping between the real and virtual motion because we envision future office environments that are intrinsically designed around the use of immersive technology. We augment this static replication with virtual content (e.g., Figure 1.1, Figure 4.1). Furthermore, we selectively render live or recorded point clouds for synchronous and asynchronous interaction, respectively.

3.2 Scenario context: A small game studio

Our scenario takes place in a small game studio. The team is in the middle of working on their current game (Figure 4.A). They plan to release a physical toy of one of their game characters (a skull with wings, Figure 4.A right) alongside the game to promote it. Therefore, they already want to start prototyping the toy today. They already have a 3D mesh of the character to be used in-game, but its shape and size make it unsuitable for 3D printing. Two team members, **Joe** and **Anna**, now discuss how to turn it into a 3D printable model. They discuss the 3D model version at the whiteboard (**WB**) in Joe's office (Figure 4.B). They create a sketch of the character seen from behind and conclude that it needs to be split into three pieces (two wings and the skull). Afterwards, Joe puts on VR goggles and starts working on the meshes (Figure 4.C left).

3.3 Synchronous part

At some point, while Joe is still wearing VR goggles, Anna drops by to check how the 3D models are progressing. Joe sees her as a live 3D reconstruction inside the replicated environment (Figure 4.C right). This is a *synchronous—co-located* interaction akin to *Dose of Reality* [40] because they are both in Joe's office and their timelines align. Shortly after, Joe finishes the 3D meshes and calls Anna, who is in her office now. He sees a live 3D reconstruction of Anna (Figure 4.D). This is a *synchronous—remote* interaction [50]. After sending the models to Anna for printing, Joe has no further tasks related to the 3D print for now and wants to continue working on the game. He switches to a programming task (Figure 4.E) and turns on *Focus Mode* so that physical distractions are blocked.

3.4 Asynchronous part

As Joe enters *Focus Mode*, the collaboration transitions to an asynchronous style. Anna occasionally drops by whenever a piece has finished printing and/or she has comments.

3.4.1 First asynchronous event. The skull piece is done first. Figure 1 shows this event in detail. Anna puts the skull on the table and leaves a spoken message, encouraging Joe to take a look when he has time (Figure 1.2). At some point, Joe takes a break from programming, sees a glossy shape on the table and triggers playback (Figure 1.3 and Figure 1.4). After inspecting the skull (Figure 1.5), he returns to his programming task. His reality is up-to-date for now because this first event was fully played back. In contrast, all subsequent events occur while he remains in *Focus Mode*.

3.4.2 Events with causalities. 'Recording' in Figure 5 shows the sequence in which Anna delivers the 3D printed pieces as well as her interactions with the WB sketch. AsyncReality internally generates a Causality graph (Figure 5 bottom-left). Each event in the example graph (numbered circles) is bound to one or more printed pieces or the WB sketch depending on what changed during the event (small circles with images). Arrows indicate how those changes are causal dependencies for later events. Changes that are not a dependency are the final states and act as trigger zones for later event playback. From here, we denote specific events as circled numbers and describe the remaining events.
②**:** The left wing is done. Anna brings it to Joe's office and tries to attach it to the skull while commenting that it does not fit very well.

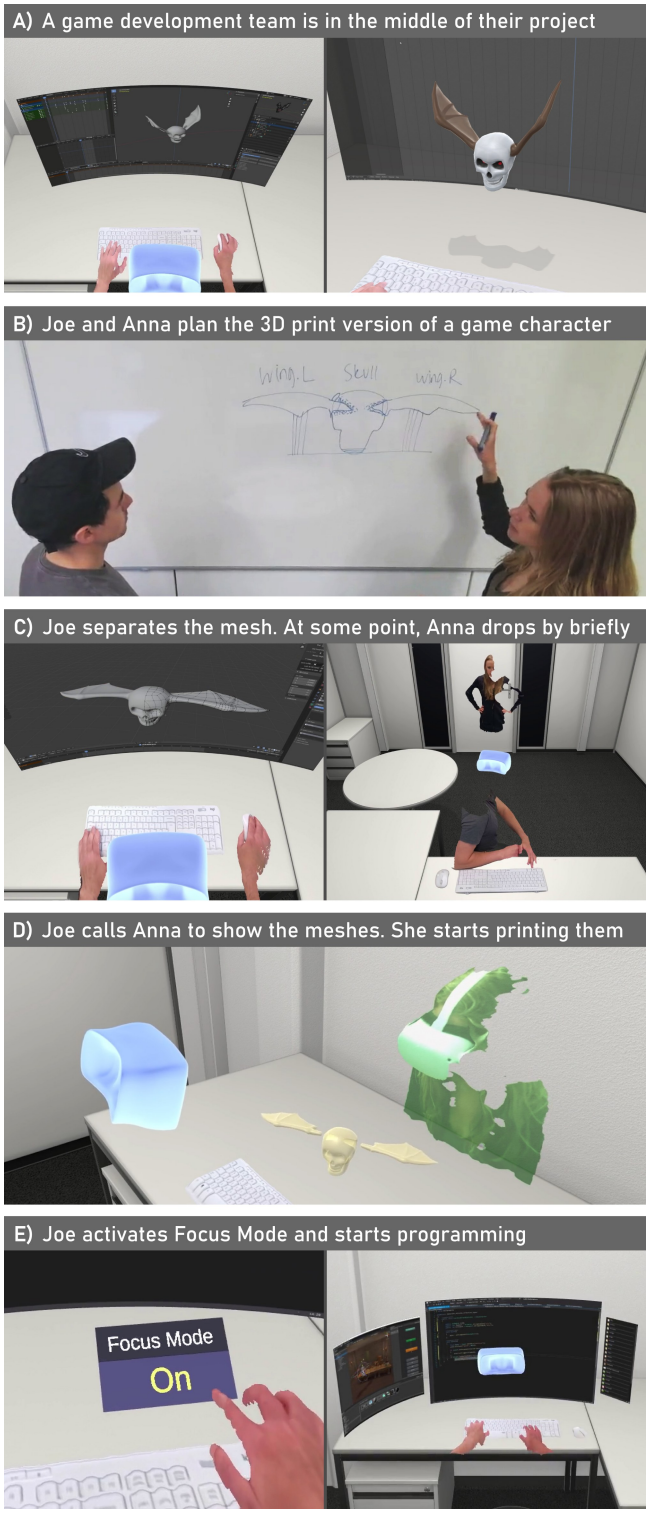

Figure 4: Context (A) and synchronous part (B–E) of our scenario. The team members Joe and Anna plan to work on a 3D print model (B), which includes synchronous co-located (C) and remote (D) interaction. The synchronous part ends when Joe enters *Focus Mode* (E) while Anna prints the pieces.

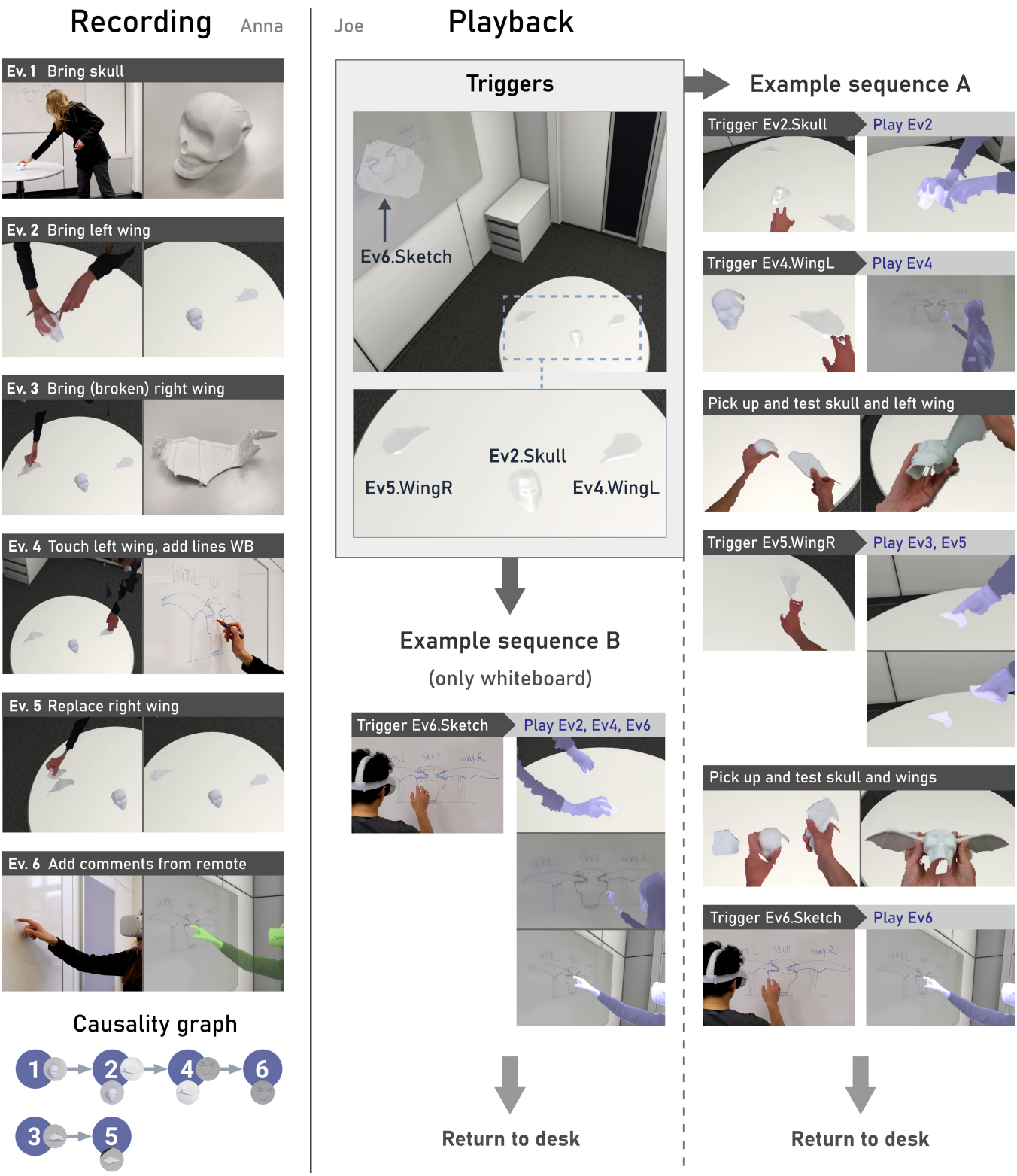

Figure 5: Scenario for event recording and causality-based playback. Left: While Joe is in *Focus Mode*, Anna brings pieces of the 3D print to his office one by one and leaves some comments. This generates a CAUSALITY GRAPH (bottom-left). Right: Objects are rendered as glossy shapes and act as spatial trigger zones to play back events. The playback order differs depending on the order in which Joe approaches the trigger zones (example sequences A and B). In sequence B, Joe is only interested in the updated sketch. Note that Joe already saw the playback of the first event earlier, so it is not part of the playback sequences.

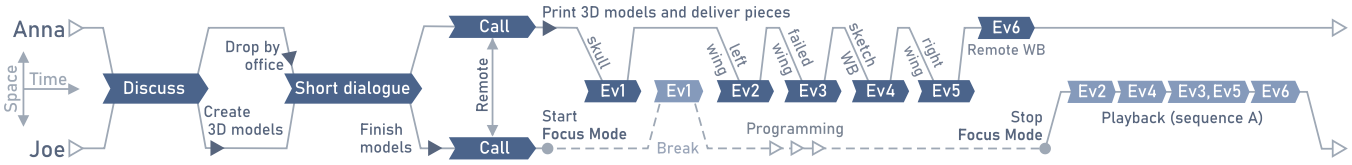

Figure 6: The timelines of Joe and Anna in our scenario. The vertical axis shows how space is shared (middle) or separated (top, bottom). The time axis shows the order of events (horizontal distances do *not* indicate relative time spans between events).

③: A first print of the right wing is done, but the print failed. Anna puts the broken wing on the table while noting that she will print it again. **④:** While the right wing is re-printing, Anna drops by with an idea for improving the 3D model. She explains how it might be better to make a hole that goes all the way through the skull and adds red dashed lines to the sketch on the WB. Importantly, within the same event, she touches the left wing. This makes the annotations causally dependent on the left wing (see arrow from ② to ④ in Causality graph). **⑤:** The re-print of the right wing has finished and Anna replaces the broken right wing. **⑥:** Anna has another idea for improving the print, but this time, she leaves the message from remote. She uses VR goggles, virtually teleports into Joe's office and points to the sketch on the whiteboard while explaining how the sides of the skull model need adjustments.

3.4.3 Joe exits Focus Mode. The 'Triggers' part in Figure 5 shows the four trigger zones that represent the final states of the skull, the left wing, the right wing, and the whiteboard sketch (also see corresponding small circles at the bottom of ②, ④, ⑤ and ⑥ in the Causality graph). The order in which Joe catches up with reality depends on the order in which he approaches the trigger zones. 'Playback' in Figure 5 shows two example sequences.

In **Sequence A**, Joe triggers the 3D print pieces first, starting with the skull. He already knows about the skull (① was already played back when he took a break), but Anna later moved it when trying out the left wing. Therefore, AsyncReality plays back ②. The skull is now at its final state (note how there is no skull after ② in the Causality graph). Therefore, AsyncReality fully reveals and renders its live reconstruction. Afterwards, Joe touches the left wing, which triggers the playback of Anna adding the dashed lines to the WB sketch. Joe then tries out the skull and the left wing to verify what Anna described. Shortly after, he approaches the right wing. ③ and ⑤ (bringing the broken right wing and then the finished right wing) are played back one after another (no matter what the actual time difference between them was), as they are both associated with the final state of the right wing. Finally, Joe walks to the WB and plays back Anna's (remote) remarks about the geometry around the holes in the skull. In the end, he returns to his desk and implements the changes suggested by Anna.

Sequence B is an alternative sequence. Joe directly goes to the WB and thus triggers ⑥ first. As illustrated in the Causality graph, ⑥ is causally dependent on ①, ② and ④. This is because Anna had touched the left wing before altering the WB sketch. Therefore, those events are played back first (except for ①, which Joe already saw), so that ⑥ makes sense for Joe. Joe then directly goes back to his desk without seeing ③ and ⑤, as he does not need to see them to implement the suggestions on the WB.

3.5 Scenario: Summary

The scenario showcases different combinations of synchronous, asynchronous, co-located and remote communication styles. Particularly in the asynchronous part, whenever Anna drops by with objects or comments, there is no need for a separate digital channel to ask for availability or to provide transient information—she simply enters Joe's office without worrying about interrupting him. Figure 6 summarizes how the two timelines of Joe and Anna interact. The purpose of the described scenario is not only to provide a use case, but also to explain the behavior of the concept in different combinations of events and causalities. We used a physical equivalent of a *Focus assist* function [42], but the same ideas apply to other situations in which synchronous communication is not available, e.g., the user being in a remote call or not in the office at all. Even the relatively simple resulting Causality graph from this scenario already covers many types of situations that we envision in an Asynchronous Reality.

4 ASYNCHRONOUS REALITY CONCEPT

We now outline our design rationale and conceptual framework of an Asynchronous Reality. Terms such as *Causality*, *Event graph* etc. have previously been used in different forms (in HCI [46], but predominantly in other fields such as statistics [70]). We use related terms in simplified ways and now define them in the context of an Asynchronous Reality in order to generalize from our scenario.

4.1 Reality constraints

As soon as the user 'freezes' the time around them (e.g., by activating *Focus Mode* or by initiating a remote call), time will perceptually only flow for objects in the user's personal space (mouse and keyboard or objects to pick up like a coffee mug). Yet no external events will be noticeable for the user. A metaphor for this is a 'time bubble' around the user within which time flows normally. Everything outside the bubble becomes 'frozen', so the room remains in the same state. When creating such a reality that is 'out-of-sync' with the physical environment, we are still bound to three *constraints*[2] from reality, which we describe in the following.

Physical intrusion: A spatially mismatched reality 'collapses' as soon as a user approaches physical objects that have been added or moved while the user was immersed. Therefore, we turn this Physical intrusion constraint into a form of user input to update the state of the room locally. By approaching parts of the room that have changed, those parts eventually enter the user's time bubble.

[2]Note that we use the term 'constraint' in a high-level conceptual sense rather than in a mathematical sense as for instance in optimization.

This updates the room parts including objects in the user's reality and they remain at this state, even when they are not in the time bubble anymore. Our implementation of this concept generates **trigger zones** around regions that are out-of-sync with the user's reality. Inspired by unobtrusive ambient notifications in physical spaces [27], trigger zones appear as semi-transparent glossy artifacts that resemble the physical objects (for instance, Figure 1.3 and 'Triggers' in Figure 5).

Causality: The previous constraint alone is not sufficient for a coherent reality. If the objects are simply being displayed inside the user's time bubble as they enter it, there would still be no causal connection in the user's reality for those objects, i.e., no 'reason' for them having changed their state. In addition, the user might need the transient information that the person who manipulated the object provided as well as all transient information from other relevant events leading up to the object's state. Storing those causal relationships is the main purpose of a Causality graph.

Spatial relevance: Because the playback is registered with the physical environment, its usefulness also depends on where exactly the user is situated during playback. A playback achieves high Spatial relevance if it contains events during which the user is at a suitable location while experiencing the playback. For instance, in the scenario sequence A, Joe is standing next to the whiteboard while Anna is pointing to the sketch in ⑥, which yields a *high* Spatial relevance because he can easily follow the message.

4.1.1 Constraint-satisfying playback. A trivial solution for quickly bringing an immersed user up-to-date after 'unfreezing time' (e.g., exiting *Focus Mode*, ending a call) would be to simply play back all events back-to-back in the sequence in which they occurred, no matter where in the room they occurred (*chronological solution*). This would trivially fulfill the aforementioned hard constraints Physical intrusion and Causality. However, this can lead to a low Spatial relevance because the user might be far away from the locations of the events that are played back.

A major difference to purely digital asynchronous messages (e.g., e-mail inbox) is that physical events are physically and semantically tied to a location and/or object in the room. For instance, updates of a whiteboard should appear when the immersed user is standing next to it, so they can directly engage after the playback ends (e.g., adding annotations, picking up the updated object). In this regard, the most suitable option may be playing back the event purely depending on *where* the immersed user is—no matter when the event occurred (*space-only solution*). However, in many cases this option would violate **causality**. For instance, objects would be moved around before they have been brought into the room or annotations would be added to sketches that are not there yet.

This leads to our main goal for event playback: achieving an as high as possible Spatial relevance (a soft constraint) *while always* meeting the Physical intrusion and Causality constraints (hard constraints). Therefore, our approach aims to identify only the minimally needed causal relationships between events (as opposed to assuming that every future event is causally dependent on every past event) and uses physical objects as triggers for starting a playback. This implies that the Causality has priority over the Spatial relevance. For instance, in the **Sequence B** of the asynchronous part of the scenario, the events of the table were played back even though Joe was standing at the whiteboard because the transient information of the physical left wing was required so that ⑥ makes sense. This is an example in which the Causality constraint overrides the Spatial relevance. The Physical intrusion has the highest priority, i.e., if users physically approach or pick up objects too early (before playback finishes), a system should reveal them as a last resort even though Causality is not fulfilled—also for safety.

4.1.2 Physical versus semantic causality. We distinguish between a **Physical causality** (i.e., based on physical manipulations) and a **Semantic causality**. In the context of our concept, a **Semantic causality** occurs when users (verbally) refer to a specific object as part of an event, e.g., by talking about an object on the table. As opposed to Physical causality, Semantic causality would require advanced inference, enabled through methods in artificial intelligence and large amounts of demonstration data to train automatic detection. Instead, our approach inherently supports a simple semi-implicit interaction in which users physically touch objects that are relevant for their message to turn a Semantic causality into a Physical causality. A concrete example is ④ in Figure 5, in which Anna touched the left wing before talking about it at the whiteboard, thereby naturally turning the semantic connection between the sketch annotation and the wing into a Physical causality.

4.2 Causality graph components

To meet the different reality constraints, we generate a Causality graph as exemplified in the scenario (Figure 5 bottom-left). A Causality graph contains Events and their causal relations (Figure 7). The graph is always acyclic (an event cannot depend on itself or future events).

A Causality graph can be constructed successively whenever a new event occurred. As a basis for causality, we first look at the difference of the room between right before and right after an event. Changes like added, moved or removed objects create Manipulations, which we abbreviate with Δ. The simplest example is a person who enters a room (event starts just before door opens), places an object inside the room, and leaves (additional object in the room after the event). In this case, the only Δ is the added object. Furthermore, touching objects or surfaces during the event also counts as a Δ (even though no object is moved between event start

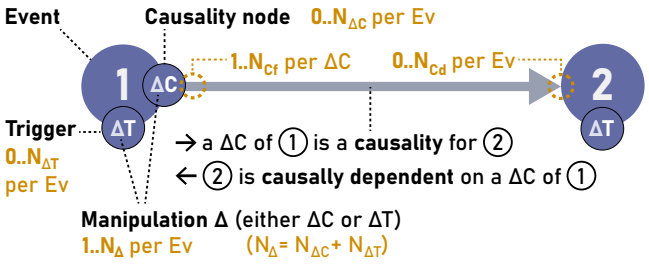

Figure 7: Generalized components of a Causality graph. Every event has one or more Manipulations (Δ). Each Δ is either a Causality node (ΔC) or a Trigger (ΔT). An event can depend on one or more Causality nodes (not necessarily from the same event).

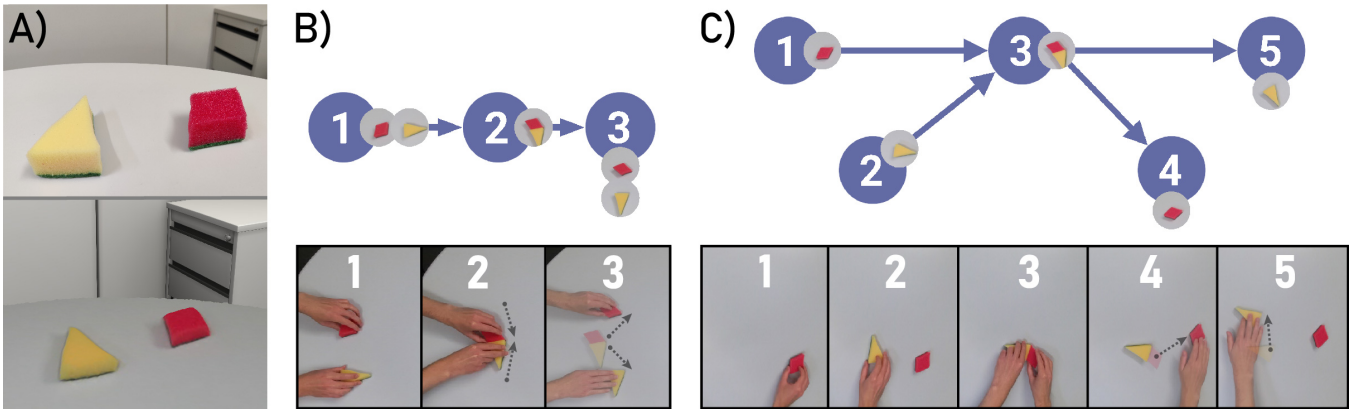

Figure 8: We use two sponges (A) to generate examples for multiple Δ per event (B) as well as graphs with converging and diverging causalities (C). (A) shows a photo (top) and reconstruction inside our VE (bottom). In (B) we put both sponges on the table within the first event (leading to a single event ① with two Causality nodes), whereas in (C), we put them on the table within two separate events (events ① & ② with one Causality node each). We separate the sponges within one event ③ in (B) or two separate events ④ & ⑤ in (C), respectively.

and end). We currently do not consider events without any Manipulations (e.g., a person entering, only leaving a spoken message, and leaving again), i.e., those could simply be treated separately and played back chronologically. This means that in a Causality graph, each event has at least one Manipulation ($N_\Delta \geq 1$). Every Δ is either a Causality node or a Trigger, i.e., $N_\Delta = N_{\Delta C} + N_{\Delta T}$.

A causality originates from an individual Manipulation of an event (not directly from the event itself) and points to the event that depends on that Manipulation, which leads to the concept of **Causality nodes**. A Causality node is a Δ that is also a causal dependency for at least one future event. In our visualizations, a Causality node is a circle (e.g., circled ΔC in Figure 7) with outgoing arrows pointing to the events it is a causality for.

In fact, an event can depend on two or more Causality nodes of a *single* previous event. For instance, in Figure 8.B, we place two objects on the table in ①. Thus, ① generates two Causality nodes (one per object), both of which ② depends on, because we attach them to each other, leading to a single Causality node in ②. Finally, ③ depends on that single 'assembled' object, which we separate again within one event.

Other possible emerging structures are *converging* and *diverging* causalities (Figure 8.C). For instance, an event could depend on the objects of multiple events, which were previously independent (in Figure 8.C, ① and ② *converge* in ③). Conversely, multiple events could depend on the same Causality node, but do not depend on each other (in Figure 8.C, ④ and ⑤ *diverge* after ③).

Causalities are *transitive* (e.g., ⑥ in the scenario is also causally dependent on the Causality nodes of ①, ② and ④). However, we only visualize and process direct causalities because the additional indirect causalities do not change the behavior during playback.

A **Trigger** is a Δ, which is *not* a causal dependency for any future event (circled ΔT in Figure 7), e.g., an object that was not manipulated anymore in subsequent events. In the graph visualizations, Triggers are located below the event and stack up if there are multiple in one event (e.g., ③ in Figure 8.B).

Overall, from an object-based perspective, a Trigger can be seen as the final physical state of an object, whereas Causality nodes can be seen as intermediate states (which always implies that the object was touched and/or moved in a later event). Initially, every Δ of a new event is a Trigger. However, subsequent events can turn Triggers of previous events into Causality nodes if they depend on them. An event can have both, Triggers and Causality nodes (e.g., in Figure 5 the skull is in its final state in ②, but ④ still depends on the wing of ②). There can also be separate connected components in the Causality graph if no dependencies exist across sets of events. This happens when events occur in separate parts of the room or, more generally, when separate objects are never manipulated within one event (e.g., the right wing in the scenario).

All Triggers have associated trigger zones in the physical space. Approaching them requests playback of the event that the Trigger is assigned to. It makes no difference, which Trigger of an event (if it has more than one) is used to initiate its playback. Whenever a playback is triggered, we recursively check whether the event depends on Causality nodes of events that are not played back yet. We play back those dependencies first (e.g., triggering ⑤ in Figure 8.C would play ①, ② and ③ first, but *not* ④). If an event was played back, then all its Causality nodes are 'reached', meaning it does not need to be played back again if future events that depend on those are played back. Furthermore, its Triggers become 'unmasked' and we remove associated trigger zones, i.e., the user then sees the object live and can interact with it.

It is worth noting that users do not need to understand the causal relationships between the events to experience a coherent reality. Causality must be fulfilled for subsequent events to make sense, but, as with other forms of communication, peers do not need to keep track of the explicit causal relationships of their actions or actions of others. Therefore, instead of revealing the causal relationships, we utilize the Physical intrusion to trigger the playback of events while the Causality determines the sequence of events, i.e., an Asynchronous Reality based system handles this automatically.

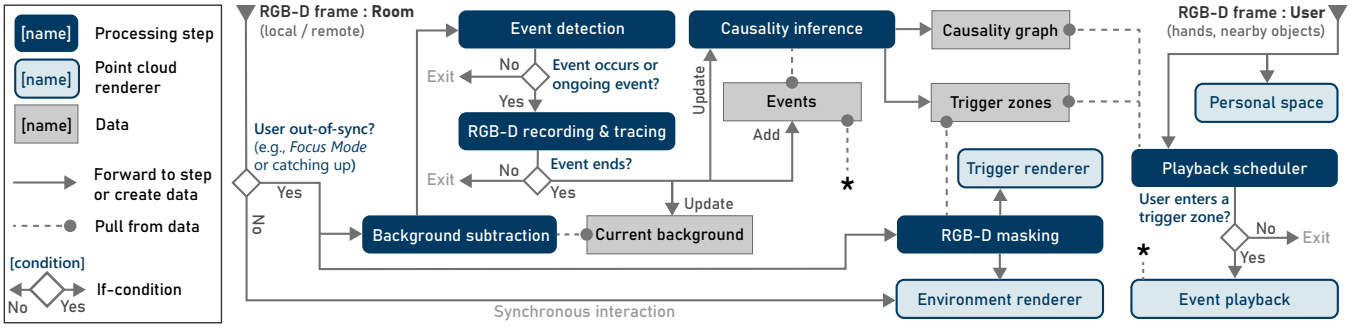

Figure 9: Overview of the general system architecture and data flow of AsyncReality. The system receives RGB-D data from the local or remote room (top-left) as well as from the space around the user (top-right). If the user unavailable (e.g., *Focus Mode*) or currently catching up with reality, then the asynchronous processing and rendering pipeline is used (see left-most condition). Otherwise, the system simply renders the point clouds live (local space and/or from remote space during calls).

5 ARCHITECTURE AND PROTOTYPE

In this section, we explain how to turn an Asynchronous Reality into practice by describing our *AsyncReality* system. This includes a general software architecture as well as our specific prototype implementation, both of which are not tied to our scenario apparatus presented earlier. We conclude with use-case agnostic system tests and a brief reflection on our approach.

5.1 Components and algorithm

In this subsection, we follow Figure 9 to describe AsyncReality's software components and dataflow. The primary input of AsyncReality consists of multiple RGB-D streams (including from remote). Note that for all point cloud renderers (bright blue nodes), we remove points that are close to the static surfaces of the environment because those surfaces are already rendered by our virtual replication (Figure 2). In the case of surfaces of interest (such as the whiteboard in the scenario), we additionally project time-accumulated color values from the RGB-D camera to the texture of the static mesh (always updating portions that are currently not occluded). The point cloud around the user is always rendered (note how RGB-D FRAME : USER is unconditionally pointing to PERSONAL SPACE renderer). The remaining parts depend on the user's availability. If the user is available and his or her reality is in sync (see left-most condition), then we simply render the live point cloud of the environment (note the direct arrow to ENVIRONMENT RENDERER). Otherwise, we initiate the asynchronous pipeline, which includes *event detection* and *causality detection*.

5.1.1 Event detection. We start with a BACKGROUND SUBTRACTION, whereas the initial background is captured when the user freezes time (e.g., enters *Focus Mode*). By averaging over multiple frames and masking out contours of the captured background, we also reduce noise. We detect events (see EVENT DETECTION node) by checking whether enough depth values differ from the background. Whenever an event is detected, we start recording the RGB-D and audio streams. By using circular buffers, we can start the recording a few frames back, i.e., from the moment just before the event starts. In addition to recording, we also 'trace' the regions at which the depth data is close, but not inside the static surfaces (between 2 cm and 5 cm above background) similar to *Lightspace* [65]. We accumulate traces over multiple frames. In particular, for every pixel we count the amount of frames the depth value was within the threshold (this can also be seen in motion as part of the 'SystemTest' videos in the supplementary materials).

Whenever enough depth values stop changing between frames, we stop the event recording. In addition to the recording, the event stores the trace so as to use it later for causality inference as well as another smoothed snapshot right after the event ends. We then add the new event to the set of EVENTS, which is simply a list of all events (with no information about causality at this point). Furthermore, we update the CURRENT BACKGROUND with the smoothed snapshot that we took at the end of the most recent event. This means that subsequent event detection steps are based on the new background and the process repeats.

5.1.2 Causality inference. Whenever a new event was recorded, our system requests the CAUSALITY INFERENCE to update the CAUSALITY GRAPH. By comparing the smoothed snapshots of the end of the event with the start of the event, we find all depth pixels that have changed (additions and subtractions). The connected components then represent the MANIPULATIONS of the event (one Δ per connected component). As soon as a Δ overlaps with a Δ of a previous event, we add a dependency. In addition to checking those states, we check the aforementioned 'traces'. If those traced regions overlap with previous TRIGGERS, then they also generate a causality (e.g., the touched left wing in ④ of Figure 5). It also allows to detect where the sketch at the whiteboard was altered or pointed at (e.g., ⑤ & ⑥ in Figure 5). Finally, our CAUSALITY INFERENCE stores MANIPULATIONS that currently do not have overlap with any other Δ and keeps the TRIGGER ZONES up-to-date accordingly.

5.1.3 Triggers and playback. The RGB-D MASKING step masks regions inside and around the TRIGGER ZONES from the depth stream, so that the user only sees objects that are outside the trigger zones (this is similar to the idea of Diminished Reality [45]). Only this masked stream is then forwarded to the ENVIRONMENT RENDERER. With this, the user can interact with objects, even if not all events in the room have been played back. The TRIGGER RENDERER renders all trigger zones that have not been unmasked yet, i.e., the portions

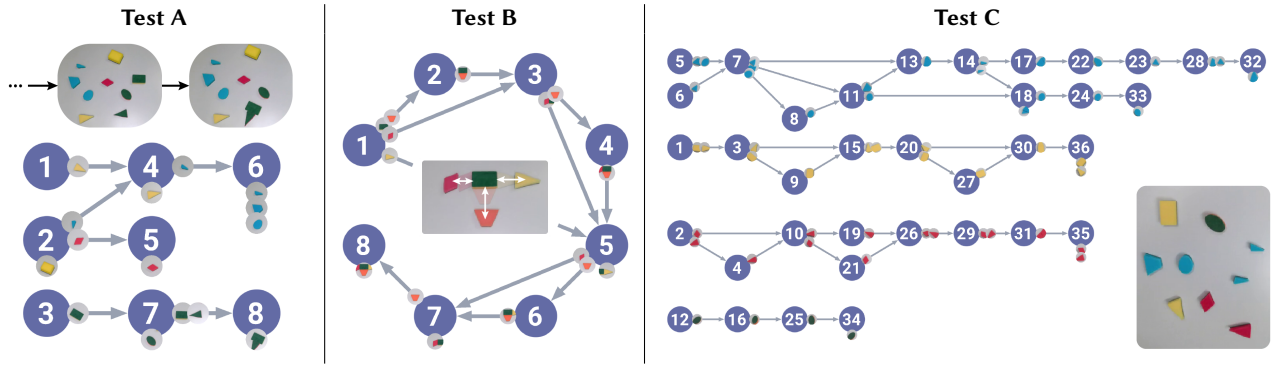

Figure 10: The resulting Causality graphs of our three system tests. In (A), we moved the sponges in a sequence so as to generate various combinations of Causality nodes and Triggers. In (B), we iterated through all combinations of three different sponges touching / not touching a big sponge in the middle (green). (C) is a stress-test with 36 events, whereas we only moved sponges of the same color within each event.

that where removed from the live stream are replaced with the static coarse glossy shapes (we use a bilateral filter with a large spatial kernel). The other input for the pipeline is the point cloud around the user (e.g., in the scenario, primarily the HMD-attached RealSense), which we use to trigger event playback by checking whether a sufficient amount of points overlaps with a trigger zone. Based on the Causality graph, the Playback manager then initiates the event playback (while meeting the Causality constraint by checking which dependent events have been played already) whereas the Event playback renders the playback as point cloud.

5.1.4 Implementation. Our implementation of the architecture runs inside *Unity2020.3.14 LTS*. For handling the dataflow, network streaming and rendering of real-time point clouds, we use the *Velt* framework [15]. On top of using existing dataflow processing steps (e.g., depth image compression [64]), we implemented the specific functionalities like event detection in the form of custom *Velt* nodes.

5.2 System tests

In order to test our implementation in different situations, we moved simple geometric shapes on a table in controlled sequences while the AsyncReality event detection and causality inference were running. We used sponges of different colors and shapes (also see Figure 8.A). In the following, we will use Figure 10, which shows the graphs that AsyncReality generated based on the tests, as a visual reference. We encourage the reader to also watch the 'SystemTest' videos in the supplementary materials, which include the intermediate steps of the Causality graph generation. We used a single Azure Kinect for the system test.

Before conducting **Test A**, we sketched a Causality graph with various combinations of Causality nodes and Triggers. Afterwards, we moved the sponges in a sequence that generates the Causality graph that we sketched beforehand. With this, we wanted to test whether we can reconstruct a given graph and whether the system behaves as expected. In **Test B**, we tested a more structured sequence. We took four sponges of different shapes and colors with the largest one (green rectangle) in the middle. Across the events, we alternately moved different outer sponges towards to or away from to the rectangular sponge. The eight events are then all combinations of the three sponges touching or not touching the big sponge. **Test C** is a stress test with 36 events (we envision such cases to happen when a user is not present for a day for instance and many events occur in the office throughout). We used various sponges with different shapes and colors (1 × green, 2 × red, 2 × yellow, 4 × blue) and first distributed them more or less randomly on the table. Then, within each event, we only moved sponges of the same color (not necessarily all per color) while also occasionally assembling and disassembling two or more same-colored sponges. The expected behavior is then that there is no causality across sponges of different colors. As can be seen in the Causality nodes and Triggers of the resulting Causality graph, AsyncReality generated four independent sub graphs—one for each color.

5.3 Object-based versus region-based detection

It is noteworthy that the connected components in our approach are often, but not always separate objects and we do not keep track of the shape of those regions. With this, it is easy to generate event sequences in which a single object incorrectly causes changes in multiple separate regions. Figure 11 is a minimal example for this: In ①, we put the triangle on the table. Then, in ② we rotate the sponge around its center, meaning that the regions in the middle of the object remain unchanged between events. This generates

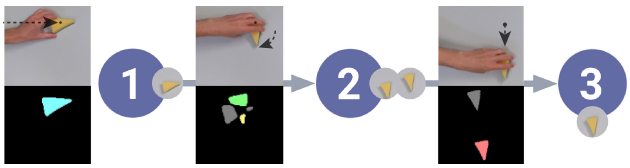

Figure 11: Minimal case for incorrectly identified connected components. The bottom row shows connected components of changed regions within the event. Gray pixels indicate regions where depth values increased from one event to another. When rotating the object in-place, ② identifies two objects, leading to two Causality nodes in ②.

two connected components of changes (see green and yellow parts of incomplete triangle). Accordingly, two Causality nodes are generated in ② pointing to different parts of the same triangle (see Causality graph). Importantly, ③ depends on both of them and generates one final Trigger for the object when we move it to a different location. The described inaccuracies could be addressed with more advanced vision techniques. However, we argue that a tracking approach does not need to be object-centric for causality-based playback. In some cases, if an object is mistakenly detected as relevant for an event, then either the playback behavior (i.e., what the immersed user sees) is unchanged or an additional unnecessary causality is added. However, while this might lead to a decreased Spatial relevance (soft constraint), it does not violate Causality (hard constraint). More problematic are non-detected causalities, e.g., if the tracing does not detect that an object was touched. Therefore, in ambiguous cases, a system should add causalities.

6 DISCUSSION AND FUTURE WORK

Asynchronous Reality is designed for future immersive workplaces. We developed the concept and showed the technical fidelity of our approach, but there are still many open questions for the larger context around future immersive workplaces.

6.1 Prototype versus vision

In our scenario and prototype implementation, we used an existing office to showcase the principles. We added the RGB-D camera infrastructure and moved furniture to best represent the use cases. We use *VR* technology in our prototype because such technology is currently the most suitable for demonstrating the concept. However, our vision of an Asynchronous Reality entails that the infrastructure, interior design, and immersive technology are fully intertwined in future offices. More specifically, the physical architecture will be based around the fact that a large portion of the working hours are spent with immersive technology. Depth cameras will be seamlessly integrated into the environment without standing out as in our apparatus (Figure 3.B) and the camera arrangement will enable optimal coverage of the volume. All surfaces will be optimized for depth sensing (i.e., bright and diffuse). The headset will be natively linked to the depth cameras to optimize rendering performance and reduce power consumption. Potentially, future headsets will be comfortable and safe enough for wear while walking larger distances, e.g., from office to office. All of this points to more detailed questions for such systems.

6.1.1 Other situations and interactions. In this work, we aimed to demonstrate the most crucial interactions for our envisioned system, but there are many situations not yet covered in our design. For instance, how would events be handled that do not make any changes to the environment (e.g., person entering, saying something and leaving again)? According to our current assumption, such events would neither be causal dependent on anything nor create any causal dependency and could therefore just be played back immediately after exiting *Focus Mode*. Furthermore, our work has explored the perspective of a *single* immersed user, whereas the interaction of the other collaboration peers will be equally meaningful in the future. There are still various interaction and system design parameters, e.g., for responding to messages from the past. One solution could be that the immersed user would just verbally respond while standing next to the played-back communication partner, who would then receive the spoken response afterwards (remotely or the next time the person is at the same location). Another challenge will be to deal with clashes of realities, e.g., when a user plays back events while another person manipulates objects in the same space. This currently counts as out-of-sync, so new events would be recorded while currently catching up with previous events. Overall, the co-existence and interaction of multiple asynchronous realities yield multiple interesting research challenges.

6.1.2 Visual fidelity. An important aspect for the usability of a system such as AsyncReality is its visual fidelity. Currently, we use non-overlapping cameras so as to capture as much space as possible with few cameras. Even though Regenbrecht et al. [53] showed that users can adapt and interact within low resolution point clouds, increasing the number of cameras and utilizing more advanced reconstruction approaches [9] would make the system more appealing for long-term usage.

6.1.3 Event and causality detection approach. Our current system has no notion of an object and no semantics or datasets are needed for recognition. This allowed us to keep complexity to a minimum and focus on Physical causality. Future approaches could benefit from 3D object segmentation or data-driven approaches to distinguish and recognize individual objects, which might lead to even higher Spatial relevance when playing back events. Another interesting extension could be increased automation by utilizing more digital or real information. For example, playback could be scheduled automatically, such as by linking events to digital information in addition to associating them with physical objects. Similarly, the *Focus Mode* to manually switch between synchronous and asynchronous communication could be replaced with sensing approaches to automatically decide whether or not the user can be interrupted [6]. Closely related, triggering playback could be based on more factors than location (e.g., gaze and movement direction).

6.2 Human factors

The previous issues are comparatively unproblematic from a technology perspective, however, there are some discussion points from a human-factor perspective. This includes formal user evaluations. While the system as a whole is challenging to evaluate with current devices and work practices, it would be feasible to investigate the understandability of specific interactions and workflows. In the future, we could investigate how well an Asynchronous Reality supports a collaborative task, e.g., by comparing recording and playback of Causality graphs with the baseline of synchronous communication including interruptions. In addition, there are factors that go beyond the effectiveness of the system.

6.2.1 Well-being. There has been little investigation on the implications of using VR devices over a longer period, e.g., on an every-day basis. With devices from today, a system such as AsyncReality could only be occasionally used for a few hours, but probably not for the whole working day. The extent to which a system based on Asynchronous Reality (or more generally immersive workspaces) can be used depends on future developments in terms of ergonomics with immersive devices.

6.2.2 Privacy. Today, much information is already shared digitally (personal and in workplaces) and video calls with personal webcams are common, particularly in recent years and with emerging home office settings. However, installing static cameras in the office raises additional privacy concerns. In future Asynchronous Reality based systems, camera streams could be kept entirely local, possibly with a separate unit to receive, process, and render all streams without a connection to a wide area network. In our scenario, cameras were on at all times for the purpose of concept demonstration. However, in a more minimal system, the event detection could use less data streams (e.g., dedicated devices to track local changes similar to existing *event cameras*) and only turn on RGB-D streaming whenever an asynchronous message needs to be recorded.

6.2.3 Acceptance of asynchronous communication. Dzardanova et al. [11] recently investigated the effectiveness of verbal and non-verbal communication in VR indicating that it can be comparable with face-to-face communication. Those and other studies investigated different aspects of *synchronous* communication, however, there is limited research about an *asynchronous* communication style like demonstrated in our work. A system such as AsyncReality may not be immediately acceptable to office workers if it were deployed from one day to another and would likely require accustomization before it becomes as normal as talking into a smartphone to leave a voice message. While potentially less problematic for *asynchronous—remote* communication, the *asynchronous—co-located* case might cause something akin to an *uncanny valley* effect (primarily known from robotics and virtual characters [61]). When entering the office to leave a message in AsyncReality, our communication partner is physically present, but the response arrives at a later point. Most likely, this communication style will be reserved for very short interactions like delivering a package paired with a short instruction. There are many ways to circumvent this from a system design perspective, e.g., if the person entering the office is wearing VR goggles as well, then the communication partner could be replaced by a more abstract avatar [10] or even by a simple recording icon (also to visually indicate that the person is not available for synchronous communication).

7 CONCLUSION

We presented the Asynchronous Reality concept and our AsyncReality prototype system. Expecting a future in which immersive devices are commonplace for productivity work, we have investigated co-located, remote, synchronous, and asynchronous collaboration styles. Whenever a user is not in the office or unavailable in other ways (e.g., in focused work or in a call), our approach records physical events and detects their causal relationships. We showed that even with a relatively simple low-cost algorithm, it is possible to detect events and causalities reliably enough to allow for immersive playback that meets causality constraints. We presented a concrete example of a AsyncReality-based working day and we conducted use-case agnostic system tests to verify the system behavior for controlled event sequences. While many open technological and human-factor challenges remain before an Asynchronous Reality can become fully feasible, we envision our concept to be applicable to future immersive office environments. With this work, we also aim to foster discussion about how to shape such future offices.

ACKNOWLEDGMENTS

This research was partially supported by the Zurich Information Security and Privacy Center (ZISC). We would like to thank Naja Morell Hjortshøj for acting in the scenario and for general support.

REFERENCES

[1] Mark Billinghurst, Suzanne Weghorst, and T Furness. 1998. Shared space: An augmented reality approach for computer supported collaborative work. *Virtual Reality* 3, 1 (1998), 25–36.

[2] Matthew Brand. 1997. The" inverse hollywood problem": From video to scripts and storyboards via causal analysis. In *AAAI/IAAI*. Citeseer, 132–137.

[3] Marc Cavazza, Jean-Luc Lugrin, Sean Crooks, Alok Nandi, Mark Palmer, and Marc Le Renard. 2005. Causality and Virtual Reality Art. In *Proceedings of the 5th Conference on Creativity and Cognition* (London, United Kingdom) *(C&C '05)*. Association for Computing Machinery, New York, NY, USA, 4âĂŞ12. https://doi.org/10.1145/1056224.1056228

[4] Lung-Pan Cheng, Eyal Ofek, Christian Holz, and Andrew D Wilson. 2019. Vroamer: generating on-the-fly VR experiences while walking inside large, unknown real-world building environments. In *2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*. IEEE, 359–366.

[5] Kevin Chow, Caitlin Coyiuto, Cuong Nguyen, and Dongwook Yoon. 2019. Challenges and Design Considerations for Multimodal Asynchronous Collaboration in VR. *Proc. ACM Hum.-Comput. Interact.* 3, CSCW, Article 40 (Nov. 2019), 24 pages. https://doi.org/10.1145/3359142

[6] Ed Cutrell, Mary Czerwinski, and Eric Horvitz. 2001. Notification, Disruption, and Memory: Effects of Messaging Interruptions on Memory and Performance. In *INTERACT 2001*. IOS Press, 263–269.

[7] Maria Danninger, Roel Vertegaal, Daniel P Siewiorek, and Aadil Mamuji. 2005. Using social geometry to manage interruptions and co-worker attention in office environments. In *Proceedings of Graphics Interface 2005*. 211–218.

[8] Mingsong Dou, Sameh Khamis, Yury Degtyarev, Philip Davidson, Sean Ryan Fanello, Adarsh Kowdle, Sergio Orts Escolano, Christoph Rhemann, David Kim, Jonathan Taylor, Pushmeet Kohli, Vladimir Tankovich, and Shahram Izadi. 2016. Fusion4D: Real-Time Performance Capture of Challenging Scenes. *ACM Trans. Graph.* 35, 4, Article 114 (July 2016), 13 pages.

[9] Ruofei Du, Ming Chuang, Wayne Chang, Hugues Hoppe, and Amitabh Varshney. 2018. Montage4D: Interactive seamless fusion of multiview video textures. In *Proceedings of ACM Interactive 3D Graphics (I3D) 2018*.

[10] Charlotte Dubosc, Geoffrey Gorisse, Olivier Christmann, Sylvain Fleury, Killian Poinsot, and Simon Richir. 2021. Impact of avatar facial anthropomorphism on body ownership, attractiveness and social presence in collaborative tasks in immersive virtual environments. *Computers & Graphics* (2021).

[11] Elena Dzardanova, Vlasios Kasapakis, Damianos Gavalas, and Stella Sylaiou. 2021. Virtual reality as a communication medium: a comparative study of forced compliance in virtual reality versus physical world. *Virtual Reality* (2021), 1–21.

[12] Barrett Ens, Joel Lanir, Anthony Tang, Scott Bateman, Gun Lee, Thammathip Piumsomboon, and Mark Billinghurst. 2019. Revisiting collaboration through mixed reality: The evolution of groupware. *International Journal of Human-Computer Studies* 131 (2019), 81–98.

[13] Facebook. 2019. *Infinite Office*. https://www.youtube.com/watch?v=5_bVkbG1ZCo (Accessed: 23rd of Dec 2021).

[14] Facebook. 2021. *Horizon*. https://about.fb.com/news/2021/08/introducing-horizon-workrooms-remote-collaboration-reimagined/ (Accessed: 23rd of Dec 2021).

[15] Andreas Fender and Jörg Müller. 2018. Velt: A Framework for Multi RGB-D Camera Systems. In *Proceedings of the 2018 ACM International Conference on Interactive Surfaces and Spaces* (Tokyo, Japan) *(ISS âĂŹ18)*. Association for Computing Machinery, New York, NY, USA, 73âĂŞ83. https://doi.org/10.1145/3279778.3279794

[16] Andreas Fender and Jörg Müller. 2019. SpaceState: Ad-Hoc Definition and Recognition of Hierarchical Room States for Smart Environments. In *Proceedings of the 2019 ACM International Conference on Interactive Surfaces and Spaces* (Daejeon, Republic of Korea) *(ISS '19)*. Association for Computing Machinery, New York, NY, USA, 303âĂŞ314. https://doi.org/10.1145/3343055.3359715

[17] Amy Fire and Song-Chun Zhu. 2015. Learning Perceptual Causality from Video. *ACM Trans. Intell. Syst. Technol.* 7, 2, Article 23 (Nov. 2015), 22 pages. https://doi.org/10.1145/2809782

[18] Amy Fire and Song-Chun Zhu. 2017. Inferring Hidden Statuses and Actions in Video by Causal Reasoning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*.

[19] Henry Fuchs, Andrei State, and Jean-Charles Bazin. 2014. Immersive 3D Telepresence. *Computer* 47, 7 (2014), 46–52. https://doi.org/10.1109/MC.2014.185

[20] Ceenu George, Philipp Janssen, David Heuss, and Florian Alt. 2019. Should I Interrupt or Not? Understanding Interruptions in Head-Mounted Display Settings. In *Proceedings of the 2019 on Designing Interactive Systems Conference*. 497–510.

[21] Ceenu George, Julia Schwuchow, and Heinrich Hussmann. 2019. Fearing Disengagement from the Real World. In *25th ACM Symposium on Virtual Reality Software and Technology* (Parramatta, NSW, Australia) *(VRST '19)*. Association for Computing Machinery, New York, NY, USA, Article 8, 5 pages. https://doi.org/10.1145/3359996.3364273

[22] Jan Gugenheimer, Evgeny Stemasov, Julian Frommel, and Enrico Rukzio. 2017. ShareVR: Enabling Co-Located Experiences for Virtual Reality between HMD and Non-HMD Users. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems* (Denver, Colorado, USA) *(CHI '17)*. Association for Computing Machinery, New York, NY, USA, 4021âĂŞ4033. https://doi.org/10.1145/3025453.3025683

[23] Jeremy Hartmann, Christian Holz, Eyal Ofek, and Andrew D. Wilson. 2019. *RealityCheck: Blending Virtual Environments with Situated Physical Reality.* Association for Computing Machinery, New York, NY, USA, 1âĂŞ12. https://doi.org/10.1145/3290605.3300577

[24] Anuruddha Hettiarachchi and Daniel Wigdor. 2016. Annexing Reality: Enabling Opportunistic Use of Everyday Objects as Tangible Proxies in Augmented Reality. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems* (San Jose, California, USA) *(CHI '16)*. Association for Computing Machinery, New York, NY, USA, 1957âĂŞ1967. https://doi.org/10.1145/2858036.2858134

[25] Gaoping Huang, Xun Qian, Tianyi Wang, Fagun Patel, Maitreya Sreeram, Yuanzhi Cao, Karthik Ramani, and Alexander J. Quinn. 2021. *AdapTutAR: An Adaptive Tutoring System for Machine Tasks in Augmented Reality.* Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/3411764.3445283

[26] Andrew Irlitti, Ross T. Smith, Stewart Von Itzstein, Mark Billinghurst, and Bruce H. Thomas. 2016. Challenges for Asynchronous Collaboration in Augmented Reality. In *2016 IEEE International Symposium on Mixed and Augmented Reality (ISMAR-Adjunct)*. 31–35. https://doi.org/10.1109/ISMAR-Adjunct.2016.0032

[27] Hiroshi Ishii, Craig Wisneski, Scott Brave, Andrew Dahley, Matt Gorbet, Brygg Ullmer, and Paul Yarin. 1998. ambientROOM: integrating ambient media with architectural space. In *CHI 98 conference summary on Human factors in computing systems*. 173–174.

[28] Shahram Izadi, David Kim, Otmar Hilliges, David Molyneaux, Richard Newcombe, Pushmeet Kohli, Jamie Shotton, Steve Hodges, Dustin Freeman, Andrew Davison, and Andrew Fitzgibbon. 2011. KinectFusion: Real-Time 3D Reconstruction and Interaction Using a Moving Depth Camera. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology* (Santa Barbara, California, USA) *(UIST '11)*. Association for Computing Machinery, New York, NY, USA, 559âĂŞ568. https://doi.org/10.1145/2047196.2047270

[29] Robert Johansen. 1988. *GroupWare: Computer Support for Business Teams.* The Free Press, USA.

[30] Takeo Kanade, Peter Rander, and P.J. Narayanan. 1997. Virtualized Reality: Constructing Virtual Worlds from Real Scenes. *Multimedia, IEEE* 4 (02 1997), 34 – 47. https://doi.org/10.1109/93.580394

[31] Mohamed Kari, Tobias Grosse-Puppendahl, Luis Falconeri Coelho, Andreas Rene Fender, David Bethge, Reinhard Schütte, and Christian Holz. 2021. TransforMR: Pose-Aware Object Substitution for Composing Alternate Mixed Realities. In *Proceedings of the 20th IEEE International Symposium on Mixed and Augmented Reality (ISMAR '21)*. Association for Computing Machinery, New York, NY, USA, 11 pages.

[32] Pascal Knierim, Thomas Kosch, Gabrielle LaBorwit, and Albrecht Schmidt. 2020. Altering the Speed of Reality? Exploring Visual Slow-Motion to Amplify Human Perception Using Augmented Reality. In *Proceedings of the Augmented Humans International Conference* (Kaiserslautern, Germany) *(AHs '20)*. Association for Computing Machinery, New York, NY, USA, Article 2, 5 pages. https://doi.org/10.1145/3384657.3384659

[33] Marek Kowalski, Jacek Naruniec, and Michal Daniluk. 2015. Livescan3D: A Fast and Inexpensive 3D Data Acquisition System for Multiple Kinect v2 Sensors. In *2015 International Conference on 3D Vision*. 318–325. https://doi.org/10.1109/3DV.2015.43

[34] Gun A. Lee, Seungjun Ahn, William Hoff, and Mark Billinghurst. 2020. Enhancing First-Person View Task Instruction Videos with Augmented Reality Cues. In *2020 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. 498–508. https://doi.org/10.1109/ISMAR50242.2020.00078

[35] Gun A Lee, Theophilus Teo, Seungwon Kim, and Mark Billinghurst. 2018. A user study on mr remote collaboration using live 360 video. In *2018 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. IEEE, 153–164.

[36] Klemen Lilija, Henning Pohl, and Kasper Hornbæk. 2020. *Who Put That There? Temporal Navigation of Spatial Recordings by Direct Manipulation.* Association for Computing Machinery, New York, NY, USA, 1âĂŞ11. https://doi.org/10.1145/3313831.3376604

[37] David Lindlbauer and Andy D. Wilson. 2018. Remixed Reality: Manipulating Space and Time in Augmented Reality *(CHI '18)*. Association for Computing Machinery, New York, NY, USA, 1âĂŞ13. https://doi.org/10.1145/3173574.3173703

[38] Thomas Lopez, Olivier Dumas, Fabien Danieau, Bertrand Leroy, Nicolas Mollet, and Jean-François Vial. 2017. A Playback Tool for Reviewing VR Experiences. In *Proceedings of the 23rd ACM Symposium on Virtual Reality Software and Technology* (Gothenburg, Sweden) *(VRST '17)*. Association for Computing Machinery, New York, NY, USA, Article 83, 2 pages. https://doi.org/10.1145/3139131.3141776

[39] J.-L. Lugrin, P. Libardi, M.J. Barnes, M. Le Bras, and M. Cavazza. 2004. Event-based causality in virtual reality. In *2004 IEEE International Conference on Systems, Man and Cybernetics (IEEE Cat. No.04CH37583)*, Vol. 1. 156–163 vol.1. https://doi.org/10.1109/ICSMC.2004.1398290

[40] Mark McGill, Daniel Boland, Roderick Murray-Smith, and Stephen Brewster. 2015. A Dose of Reality: Overcoming Usability Challenges in VR Head-Mounted Displays. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems* (Seoul, Republic of Korea) *(CHI '15)*. Association for Computing Machinery, New York, NY, USA, 2143âĂŞ2152. https://doi.org/10.1145/2702123.2702382

[41] Manuel Meier, Paul Streli, Andreas Fender, and Christian Holz. 2021. TapID: Rapid Touch Interaction in Virtual Reality using Wearable Sensing. In *2021 IEEE Virtual Reality and 3D User Interfaces (VR)*. 519–528. https://doi.org/10.1109/VR50410.2021.00076

[42] Microsoft. 2021. *Focus assist.* (Function in Windows operating systems).

[43] Paul Milgram and Fumio Kishino. 1994. A taxonomy of mixed reality visual displays. *IEICE TRANSACTIONS on Information and Systems* 77, 12 (1994), 1321–1329.

[44] Roberto A. Montano Murillo, Elia Gatti, Miguel Oliver Segovia, Marianna Obrist, Jose P. Molina Masso, and Diego Martinez Plasencia. 2017. NaviFields: Relevance Fields for Adaptive VR Navigation. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology* (Québec City, QC, Canada) *(UIST '17)*. Association for Computing Machinery, New York, NY, USA, 747âĂŞ758. https://doi.org/10.1145/3126594.3126645

[45] Shohei Mori, Sei Ikeda, and Hideo Saito. 2017. A survey of diminished reality: Techniques for visually concealing, eliminating, and seeing through real objects. *IPSJ Transactions on Computer Vision and Applications* 9 (06 2017). https://doi.org/10.1186/s41074-017-0028-1

[46] Mathieu Nancel and Andy Cockburn. 2014. Causality: A Conceptual Model of Interaction History. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Toronto, Ontario, Canada) *(CHI '14)*. Association for Computing Machinery, New York, NY, USA, 1777âĂŞ1786. https://doi.org/10.1145/2556288.2556990

[47] Bingbing Ni, Shuicheng Yan, and Ashraf Kassim. 2009. Recognizing human group activities with localized causalities. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*. 1470–1477. https://doi.org/10.1109/CVPR.2009.5206853

[48] Niels Christian Nilsson, Tabitha Peck, Gerd Bruder, Eri Hodgson, Stefania Serafin, Mary Whitton, Frank Steinicke, and Evan Suma Rosenberg. 2018. 15 Years of Research on Redirected Walking in Immersive Virtual Environments. *IEEE Computer Graphics and Applications* 38, 2 (2018), 44–56. https://doi.org/10.1109/MCG.2018.111125628

[49] Brid O'Conaill and David Frohlich. 1995. Timespace in the workplace: Dealing with interruptions. In *Conference companion on Human factors in computing systems*. 262–263.

[50] Sergio Orts-Escolano, Christoph Rhemann, Sean Fanello, Wayne Chang, Adarsh Kowdle, Yury Degtyarev, David Kim, Philip L. Davidson, Sameh Khamis, Mingsong Dou, Vladimir Tankovich, Charles Loop, Qin Cai, Philip A. Chou, Sarah Mennicken, Julien Valentin, Vivek Pradeep, Shenlong Wang, Sing Bing Kang, Pushmeet Kohli, Yuliya Lutchyn, Cem Keskin, and Shahram Izadi. 2016. Holoportation: Virtual 3D Teleportation in Real-Time. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology* (Tokyo, Japan) *(UIST '16)*. Association for Computing Machinery, New York, NY, USA, 741âĂŞ754. https://doi.org/10.1145/2984511.2984517

[51] V.M.R. Penichet, I. Marin, J.A. Gallud, M.D. Lozano, and R. Tesoriero. 2007. A Classification Method for CSCW Systems. *Electronic Notes in Theoretical Computer Science* 168 (2007), 237–247. https://doi.org/10.1016/j.entcs.2006.12.007 Proceedings of the Second International Workshop on Views on Designing Complex Architectures (VODCA 2006).

[52] Thammathip Piumsomboon, Youngho Lee, Gun Lee, and Mark Billinghurst. 2017. CoVAR: A Collaborative Virtual and Augmented Reality System for Remote Collaboration. In *SIGGRAPH Asia 2017 Emerging Technologies* (Bangkok, Thailand) *(SA '17)*. Association for Computing Machinery, New York, NY, USA, Article 3, 2 pages. https://doi.org/10.1145/3132818.3132822

[53] Holger Regenbrecht, Katrin Meng, Arne Reepen, Stephan Beck, and Tobias Langlotz. 2017. Mixed Voxel Reality: Presence and Embodiment in Low Fidelity, Visually Coherent, Mixed Reality Environments. In *2017 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. 90–99. https://doi.org/10.1109/ISMAR.2017.26

[54] Tom Rodden and Gordon Blair. 1991. CSCW and distributed systems: The problem of control. In *Proceedings of the Second European Conference on Computer-Supported Cooperative Work ECSCWâĂŹ91*. Springer, 49–64.

[55] Joan Sol Roo and Martin Hachet. 2017. One Reality: Augmenting How the Physical World is Experienced by Combining Multiple Mixed Reality Modalities. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology* (Québec City, QC, Canada) *(UIST '17)*. Association for Computing Machinery, New York, NY, USA, 787âĂŞ795.

[56] Lior Shapira and Daniel Freedman. 2016. Reality skins: Creating immersive and tactile virtual environments. In *2016 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. IEEE, 115–124.

[57] Adalberto L. Simeone, Eduardo Velloso, and Hans Gellersen. 2015. Substitutional Reality: Using the Physical Environment to Design Virtual Reality Experiences. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems* (Seoul, Republic of Korea) *(CHI '15)*. Association for Computing Machinery, New York, NY, USA, 3307âĂŞ3316. https://doi.org/10.1145/2702123.2702389

[58] Misha Sra. 2016. Asymmetric Design Approach and Collision Avoidance Techniques For Room-Scale Multiplayer Virtual Reality. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology* (Tokyo, Japan) *(UIST '16 Adjunct)*. Association for Computing Machinery, New York, NY, USA, 29âĂŞ32. https://doi.org/10.1145/2984751.2984788

[59] Misha Sra, Sergio Garrido-Jurado, and Pattie Maes. 2018. Oasis: Procedurally Generated Social Virtual Spaces from 3D Scanned Real Spaces. *IEEE Transactions on Visualization and Computer Graphics* 24, 12 (2018), 3174–3187. https://doi.org/10.1109/TVCG.2017.2762691

[60] Edward R Sykes. 2011. Interruptions in the workplace: A case study to reduce their effects. *International Journal of Information Management* 31, 4 (2011), 385–394.

[61] Angela Tinwell, Mark Grimshaw, and Debbie Abdel-Nabi. 2014. *The Uncanny Valley and Nonverbal Communication in Virtual Characters*. ETC Press, Pittsburgh, PA, USA, 325âĂŞ341.

[62] Chiu-Hsuan Wang, Chia-En Tsai, Seraphina Yong, and Liwei Chan. 2020. Slice of Light: Transparent and Integrative Transition Among Realities in a Multi-HMD-User Environment. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology* (Virtual Event, USA) *(UIST '20)*. Association for Computing Machinery, New York, NY, USA, 805âĂŞ817. https://doi.org/10.1145/3379337.3415868

[63] Tianyi Wang, Xun Qian, Fengming He, Xiyun Hu, Ke Huo, Yuanzhi Cao, and Karthik Ramani. 2020. CAPturAR: An Augmented Reality Tool for Authoring Human-Involved Context-Aware Applications. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology* (Virtual Event, USA) *(UIST '20)*. Association for Computing Machinery, New York, NY, USA, 328âĂŞ341. https://doi.org/10.1145/3379337.3415815

[64] Andrew D. Wilson. 2017. Fast Lossless Depth Image Compression. In *Proceedings of the 2017 ACM International Conference on Interactive Surfaces and Spaces* (Brighton, United Kingdom) *(ISS '17)*. Association for Computing Machinery, New York, NY, USA, 100âĂŞ105. https://doi.org/10.1145/3132272.3134144

[65] Andrew D. Wilson and Hrvoje Benko. 2010. *Combining Multiple Depth Cameras and Projectors for Interactions on, above and between Surfaces*. Association for Computing Machinery, New York, NY, USA, 273âĂŞ282.

[66] Andrew D. Wilson and Hrvoje Benko. 2016. Projected Augmented Reality with the RoomAlive Toolkit *(ISS '16)*. Association for Computing Machinery, New York, NY, USA, 517âĂŞ520. https://doi.org/10.1145/2992154.2996362

[67] Haijun Xia, Sebastian Herscher, Ken Perlin, and Daniel Wigdor. 2018. Spacetime: Enabling Fluid Individual and Collaborative Editing in Virtual Reality. In *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology* (Berlin, Germany) *(UIST '18)*. Association for Computing Machinery, New York, NY, USA, 853âĂŞ866. https://doi.org/10.1145/3242587.3242597

[68] Jackie Yang, Christian Holz, Eyal Ofek, and Andrew D Wilson. 2019. Dreamwalker: Substituting real-world walking experiences with a virtual reality. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology*. 1093–1107.

[69] Hongming Zhang, Yintong Huo, Xinran Zhao, Yangqiu Song, and Dan Roth. 2020. Learning Contextual Causality from Time-consecutive Images. *arXiv preprint arXiv:2012.07138* (2020).

[70] Qin Zhang, Chunling Dong, Yan Cui, and Zhihui Yang. 2014. Dynamic Uncertain Causality Graph for Knowledge Representation and Probabilistic Reasoning: Statistics Base, Matrix, and Application. *IEEE Transactions on Neural Networks and Learning Systems* 25, 4 (2014), 645–663. https://doi.org/10.1109/TNNLS.2013.2279320

[71] Yiwei Zhao and Sean Follmer. 2018. A Functional Optimization Based Approach for Continuous 3D Retargeted Touch of Arbitrary, Complex Boundaries in Haptic Virtual Reality. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (Montreal QC, Canada) *(CHI '18)*. Association for Computing Machinery, New York, NY, USA, 1âĂŞ12. https://doi.org/10.1145/3173574.3174118

[72] Michael Zollhöfer, Patrick Stotko, Andreas Görlitz, Christian Theobalt, Matthias Nießner, Reinhard Klein, and Andreas Kolb. 2018. State of the Art on 3D Reconstruction with RGB-D Cameras. In *Computer graphics forum*, Vol. 37. Wiley Online Library, 625–652.