

Article

A Framework Designed with Perceptual Symmetry and Interactive Asymmetry for XR Collaboration

Gustavo Adolfo Murillo Gutierrez ¹, Rong Jin ², Juan-Pablo I. Ramirez-Paredes ¹
and Uriel Haile Hernandez Belmonte ^{3,*}

¹ Department of Electronics Engineering, University of Guanajuato, Carretera Salamanca—Valle de Santiago km 3.5 + 1.8, Comunidad de Palo Blanco, Salamanca 36787, Mexico; ga.murillogutierrez@ugto.mx (G.A.M.G.); jpi.ramirez@ugto.mx (J.-P.I.R.-P.)

² Department of Computer Science, California State University, Fullerton, CA 92831, USA; rong.jin@fullerton.edu

³ Department of Art and Bussines, University of Guanajuato, Carretera Salamanca—Valle de Santiago km 3.5 + 1.8, Comunidad de Palo Blanco, Salamanca 36787, Mexico

* Correspondence: uh.hernandez@ugto.mx; Tel.: +52-464-647-9940

Abstract

Collaborative experiences are enriched through cross-platform interactions in the context of eXtended Reality (XR) systems. In this paper, we introduce SRVS-C (Spatially Referenced Virtual Synchronization for Collaboration), a centralized framework designed to support co-located, real-time AR (on smartphone) and VR (in headset) interactions over local networks. The framework adopts an architecture of interactive asymmetry, where the interaction roles, input modalities, and rendering responsibilities are adapted to the unique capabilities and constraints of each device. Concurrently, the framework maintains perceptual symmetry, guaranteeing a coherent spatial and semantic experience for all users. This is achieved through anchor-based spatial registration and unified data representations. Compared to prior work that relies on cloud services or symmetric platforms (e.g., VR–VR, AR–AR, and PC–PC pairings), SRVS-C supports seamless communication between AR and VR endpoints, operating entirely over TCP sockets using serialization-agnostic message formats. We evaluated SRVS-C in a dual-user scenario involving a mobile AR and a VR headset, using shared freehand drawing tasks. These tasks include simple linear strokes and geometry-rich drawing content to assess how varying interaction complexity—ranging from low-density sketches to intricate, high-vertex structures—impacted the end-to-end latency, state replication timing, and collaborative fluency. The results show that the system sustains latency between 35 ms and 175 ms, even during rapid, continuous drawing actions that generate a high number of stroke updates per second, and when handling drawings composed of numerous vertices and complex shapes. Throughout these conditions, the system maintains perceptual continuity and spatial alignment across users by applying platform-specific interactive asymmetry.

Keywords: collaborative XR system; interactive asymmetry; perceptual symmetry; spatial anchoring; centralized architecture



Academic Editor: Jie Yang

Received: 1 September 2025

Revised: 24 September 2025

Accepted: 24 September 2025

Published: 3 November 2025

Citation: Murillo Gutierrez, G.A.; Jin, R.; Ramirez-Paredes, J.-P.I.; Hernandez Belmonte, U.H. A Framework Designed with Perceptual Symmetry and Interactive Asymmetry for XR Collaboration. *Symmetry* **2025**, *17*, 1842. <https://doi.org/10.3390/sym17111842>

Copyright: © 2025 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and

conditions of the Creative Commons Attribution (CC BY) license

(<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Collaborative eXtended Reality (XR) systems have evolved from isolated platforms to mixed AR/VR modes across diverse platforms [1]. This blend has enabled XR environments to now support persistent virtual content shared by co-located or remote users,

anchored to either real-world locations or fully immersive simulations [2,3]. However, these cross-platform XR systems in real-world settings remain constrained by three key challenges: (i) ensuring synchronization fidelity across diverse platforms with different tracking capabilities; (ii) maintaining interoperability among devices with heterogeneous hardware capabilities and input modalities; and (iii) designing interaction models that accommodate both symmetric (role-equivalent) and asymmetric (role-specialized) collaboration paradigms [4,5]. These challenges are not only technical but also impact collaborative fluency, spatial awareness, and mutual understanding between users operating on different platforms or devices. For example, a user wearing a VR headset with 6 Degrees of Freedom (6DoF) controllers may benefit from high-fidelity spatial input and feedback, while a mobile AR user may be constrained by a limited field of view, less accurate anchoring, and restricted input capabilities. When such disparities are not explicitly handled, user interaction becomes fragmented, interpretations of the shared scene diverge, and collaboration devolves into unilateral, parallel actions rather than a shared, collaborative experience.

Previous work has explored various architectural strategies to address those challenges. Client-server models [6–8] and hierarchical delegation schemes [9] focus on maintaining global state consistency across platforms, while cloud-based approaches [10,11] and multi-platform development toolkits [12] address cross-device integration. However, many of these studies assume homogeneity in platform capabilities or rely on proprietary cloud infrastructure that introduces latency, scalability limitations, and compliance issues in local area or bandwidth-constrained environments. Even those widely adopted commercial frameworks such as the Microsoft Mixed Reality Toolkit [13], Vuforia Engine [14], or Meta’s Mixed Reality Utility Kit [15] impose cloud tethering, licensing restrictions, or hardware lock-ins that restrict their applicability to open-source for developing collaborative contexts. These limitations highlight the need for system architectures that explicitly accommodate heterogeneous platforms, operate independently of cloud infrastructure, and support real-time communication in both perceptually symmetric and interactively asymmetric configurations. This demand is particularly necessary for on-site collaboration in classrooms, labs, or meeting rooms, where people work in shared physical spaces using their diverse devices.

In this work, we present SRVS-C (Spatially Referenced Virtual Synchronization for Collaboration), a centralized architecture supporting co-located, real-time collaborative XR experiences across heterogeneous AR (smartphone-based) and VR headset devices over local access networks. This framework is designed to preserve environmental coherence across spatial (geometric alignment), semantic (shared meaning of objects/actions), and temporal (synchronized state updates) dimensions under conditions of hardware disparity. It supports devices communicating without reliance on cloud services, solely using TCP-based communication and open serialization standards. The full implementation of SRVS-C is released as free and open-source software and is available through its public repository [16]. In summary, the SRVS-C framework introduces three key features as follows:

- **An interactive asymmetry design** in which interaction roles and input modalities are deliberately differentiated according to the unique affordances and constraints of each platform. For example, AR users may contribute through touch-based object manipulation, while VR users may operate within fully immersive, controller-based environments.
- **A perceptual symmetry model** redefined in this work to describe the condition in which all users, regardless of the technology they use, perceive the shared experience in a comparable, coherent, and consistent manner. Unlike interactive asymmetry, which accepts and leverages differences in how users act within the system, perceptual symmetry focuses on ensuring that these differences do not lead to mismatches in

what users see, understand, or experience within the collaborative environments. This reinterpretation aims to evaluate whether the system truly aligns perceptual outcomes across heterogeneous devices.

- **A data synchronization mechanism** that integrates local spatial anchoring with platform-independent data serialization to guarantee perceptual symmetry, ensuring that all devices maintain a consistent and coherent shared spatial state regardless of hardware differences.

To validate its feasibility, we implemented a dual-user test scenario with a mobile AR and a VR headset, using shared freehand drawing tasks. These tasks include simple linear strokes and geometry-rich drawing content to assess how varying interaction complexity—ranging from low-density sketches to intricate, high-vertex structures—impacted the end-to-end latency, state replication timing, and collaborative fluency.

The remaining part of this paper is structured as follows: Section 2 reviews related work about hybrid interactions over AR and VR platforms, introducing architectures and limitations in heterogeneous devices. Section 3 introduces the proposed SRVS-C system, outlining its architecture and key technologies such as synchronization mechanisms, data abstraction layers, and a cross-platform interface. Section 4 describes the experimental setup and evaluation protocol used to benchmark system performance. Section 5 presents the experimental results through various test scenarios and we discuss some key findings in this work. Lastly, Section 6 summarizes the work and outlines a few potential future directions in scalability of the system and its real-world applications.

2. Related Work

The foundation for understanding asymmetry in XR systems has been approached from multiple perspectives in recent literature [17,18]. The conceptual studies of Numan and Steed [5] as well as Sereno et al. [19] articulate how technological asymmetries, that is, inherent disparities in device output capabilities, are exhibited as two distinct asymmetries: perceptual asymmetry, where variations in sensory representation (e.g., field of view, haptic feedback) lead to diverging user experiences, and interactive asymmetry, which arises from disparities in control paradigms and task allocation strategies. This framework was later operationalized by Brehault et al. [20], who decomposed these asymmetries into eight measurable dimensions, including input fidelity and spatial awareness gradients, and thus provided an evaluation matrix that connects core technological mismatches to their collaborative implications.

Recent studies challenge the assumption that asymmetric configurations are inherently detrimental to collaboration in virtual environments. Grandi et al. [21] provide empirical evidence showing that asymmetric VR–AR interaction can be as effective as two symmetric setups (AR–AR pairs and VR–VR pairs), facilitating the development of hybrid collaborative systems in XR across diverse devices. Zhang et al. [22] demonstrated that a PC–VR asymmetric setup outperforms symmetric configurations such as AR–AR pairs, PC–PC pairs, and VR–VR pairs, particularly in user experience (e.g., closeness of relationships, social presence) and task performance (e.g., manipulation efficiency and overall score). Also, Agnes et al. [23] found that while symmetric VR–VR configurations tend to generate a greater number of ideas during collaboration, asymmetric setups such as VR–PC pairings lead to more efficient task delegation and communication. In addition, Schäfer et al. [24] demonstrated that calibrated interactive asymmetry can enhance collaborative performance by promoting task specialization without undermining spatial or social awareness. These findings validate our architectural approach, which embraces interactive asymmetry as a constructive design parameter rather than a constraint.

Several frameworks have been developed to support collaborative XR experiences; however, many face notable limitations in deployment versatility. For example, Herskovitz et al. [25] introduced valuable capabilities for shared spatial recognition in AR applications, but its design is tailored for homogeneous device ecosystems, lacking native mechanisms to interoperate with heterogeneous configurations that combine diverse devices and both AR and VR platforms. Likewise, Meta’s MR Utility Kit offers a robust foundation for scene management and interaction in mixed reality, but imposes platform-specific constraints that restrict its applicability in infrastructure-limited contexts. In addition, much prior work has focused on cloud–edge hybrids [26] or fully cloud-based systems [11], deliver scalable processing and storage capabilities, yet their performance hinges on the availability of high-bandwidth, low-latency internet connections and centralized cloud services, requirements that are often impractical in local, infrastructure-constrained collaborative scenarios.

XR collaboration applications have been explored across various domains—including education [27], cultural heritage [28], and manufacturing [29]. However, these implementations often prioritize perceptual fidelity over device interoperability or operate exclusively within environments where bandwidth, latency, and connectivity stability are guaranteed, such as laboratory testbeds or institution-managed local networks.

3. Proposed System: Spatially Referenced Virtual Synchronization for Collaboration (SRVS-C)

The SRVS-C operates over local networks, processing data locally to minimize reliance on external servers. The design targets low-latency communication performance, enabling cross-platform interoperability without requiring specialized hardware. Inspired by the centralized consistency model of Chang et al. [30] and building on the prior work [31], SRVS-C introduces three key enhancements: (1) support for asymmetric AR/VR platforms, enabling real-time collaboration between devices with distinct capabilities and interaction modes; (2) heartbeat-based connection monitoring, in which lightweight periodic signals verify link availability and promptly detect disconnections; and (3) selective event propagation to reduce network congestion.

As shown in Figure 1, SRVS-C is composed of four core components: a local collaborative framework, a centralized synchronization mechanism, a serialization subsystem, and dual platform-agnostic Unity APIs. The system implements an interactive asymmetric model where mobile AR users interact via touch-based input and visual-inertial anchoring, while VR users utilize room-scale 6DoF controllers. Environmental coherence is maintained through a unified spatial referencing system, in which device localization is achieved via anchor-based referential mapping—anchors are fixed spatial reference points in the physical world, tracked by the device’s sensors, to which virtual content is consistently aligned. Each device defines its own local origin O_i at coordinates $(0, 0, 0)$, synchronizing spatial relationships through propagated anchor transformations.

The following section describes each component in detail, showing how they collectively ensure seamless collaborative experiences across AR and VR devices.

3.1. Local Collaborative Framework

Implemented in Node.js, the collaboration layer follows a client–server model over TCP sockets, supporting real-time messaging and session isolation. Upon connection, users are assigned system-generated unique identifiers to ensure secure and unambiguous user tracking within the system. They are grouped into isolated *collaborative room sessions* to prevent interference between separate sessions and maintain data integrity.

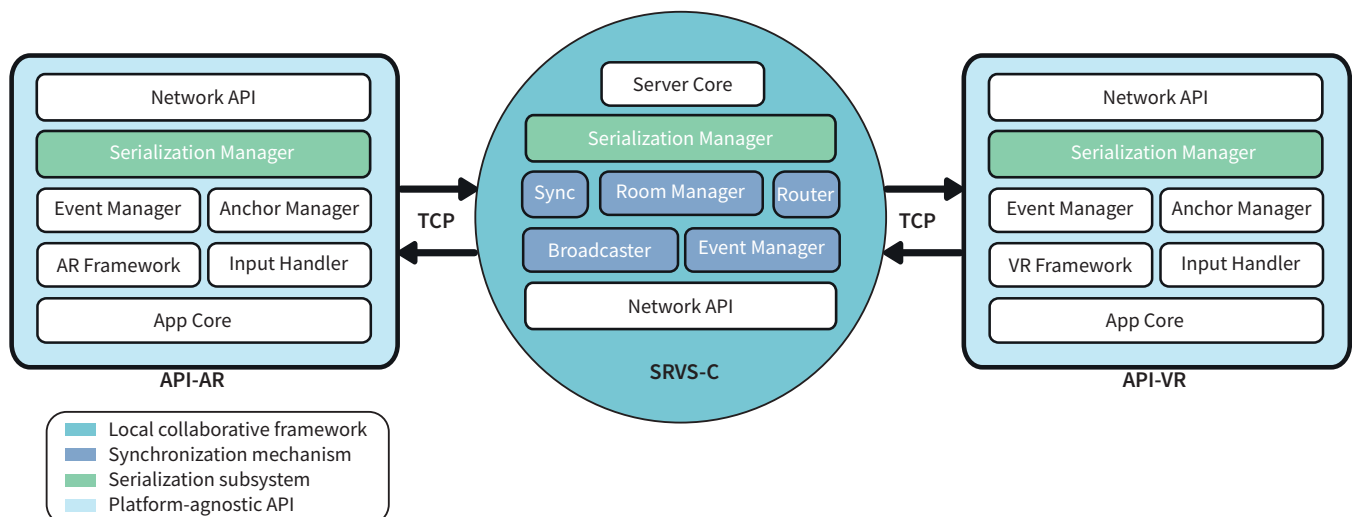


Figure 1. Architecture overview of the proposed SRVS-C system. The system is structured into three layers: API-AR, SRVS-C, and API-VR. AR and VR clients rely on platform-agnostic APIs for input, anchor management, and event handling, while a *Serialization Manager* ensures data compatibility. At the core, SRVS-C provides synchronization and session management, supported by a shared *Network API* that abstracts TCP communication across heterogeneous platforms.

To monitor connectivity, a heartbeat mechanism tracks user responsiveness. The server expects a heartbeat message from each client every 60 s (*heartbeat_interval*). A client is flagged as unresponsive and its connection is gracefully terminated if three consecutive heartbeats are missed (*missedHeartbeats*), indicating a likely disconnection. The server temporarily retains the session state of a disconnected user for a grace period of 320 s (*heartbeat_timeout*), allowing for seamless reconnection. Beyond this period, the session state is discarded from the server. The server also manages anchor ownership, user presence, and interaction logs for session recovery and state reconstruction.

3.2. Centralized Synchronization Mechanism

To ensure spatial consistency, SRVS-C uses logical anchors defined as relative transformations from each device's local origin. These anchors, identified by UUIDs and affine transformation matrices, are created by users and propagated through the server, which validates and redistributes them.

Users reconstruct received anchors into their local space, maintaining perceptual symmetry despite device-specific mapping variations. This referential model avoids complex global SLAM or map-merging, making the system robust to sensor drift or intermittent tracking—particularly critical in mobile AR scenarios. Anchor updates and user events are logged for efficient state recovery and late joiner integration.

3.3. Serialization Subsystem

SRVS-C abstracts inter-user communication using a serialization layer with interchangeable encoding formats (JSON, Protobuf, and MessagePack). This enables hybrid sessions across heterogeneous devices without enforcing strict data homogeneity.

Messages encapsulate collaborative entities—such as spatial strokes, anchor events, or user interactions—within a common schema (Table 1). This schema defines not only the content of the interaction, but also the runtime metadata necessary for reliable synchronization across heterogeneous devices. Each message carries a player identifier, a socket reference, and a room association to ensure that interactions are correctly attributed and routed within the system. Temporal markers, such as the latest heartbeat and last

interaction timestamp, enable the server to monitor presence and session continuity. The fields `isActive` and `missedHeartbeats` provide resilience mechanisms for connection recovery and state reconnection. A buffer is also maintained per session, ensuring that partially received payloads—common in streaming or unstable network conditions—are reconstructed before deserialization.

Table 1. Player session attributes and runtime connection metadata used for server-side state management.

Attribute	Description
<code>id : String</code>	Unique player identifier generated by the server using a cryptographic hash.
<code>connection : Socket</code>	Instance of the user’s TCP socket, enabling direct data transmission without intermediary layers.
<code>roomId : String</code>	Associates a player with the currently active collaborative room.
<code>lastSeen : Date</code>	Latest timestamp for both the most recent heartbeat and significant user interaction.
<code>isActive : Boolean</code>	Functional availability flag indicating active participation, regardless of physical connection status.
<code>missedHeartbeats : Number</code>	Cumulative count of missed heartbeat signals, used to trigger connection loss handling and recovery routines progressively.
<code>buffer : String</code>	Temporary storage area for partially received data over the socket stream.

3.4. User Interfaces of the SRVS-C System

The SRVS-C system comprises two distinct user interfaces—*API-AR* for augmented reality and *API-VR* for virtual reality, as shown in Figure 2. They are both implemented as script plug-ins in Unity 2022.3.55f1.

API-AR:

Implemented for Android smartphones with ARCore [32], supports touch-based interaction and surface anchoring. User gestures are interpreted via Unity’s `EnhancedTouch` and raycast into 3D strokes, anchored to detected planes. These strokes are serialized and transmitted to the server, with anchors encoded as relative transforms to support persistent spatial alignment. Cached anchors allow realignment after tracking loss.

API-VR:

API-VR enables immersive 3D interaction via 6DoF controllers. Users draw in space by triggering raycast-based strokes. As in AR, strokes are serialized, transmitted, and reconstructed incrementally, ensuring fluid rendering and alignment with remote content. The extended input capabilities of VR—such as anchor manipulation and in-scene UI interaction—are supported without compromising semantic consistency across platforms.

The APIs have been instantiated in a collaborative spatial sketching application, as it is a fundamental modality in immersive co-design workflows [33], to validate the interoperability of SRVS-C in practical settings.

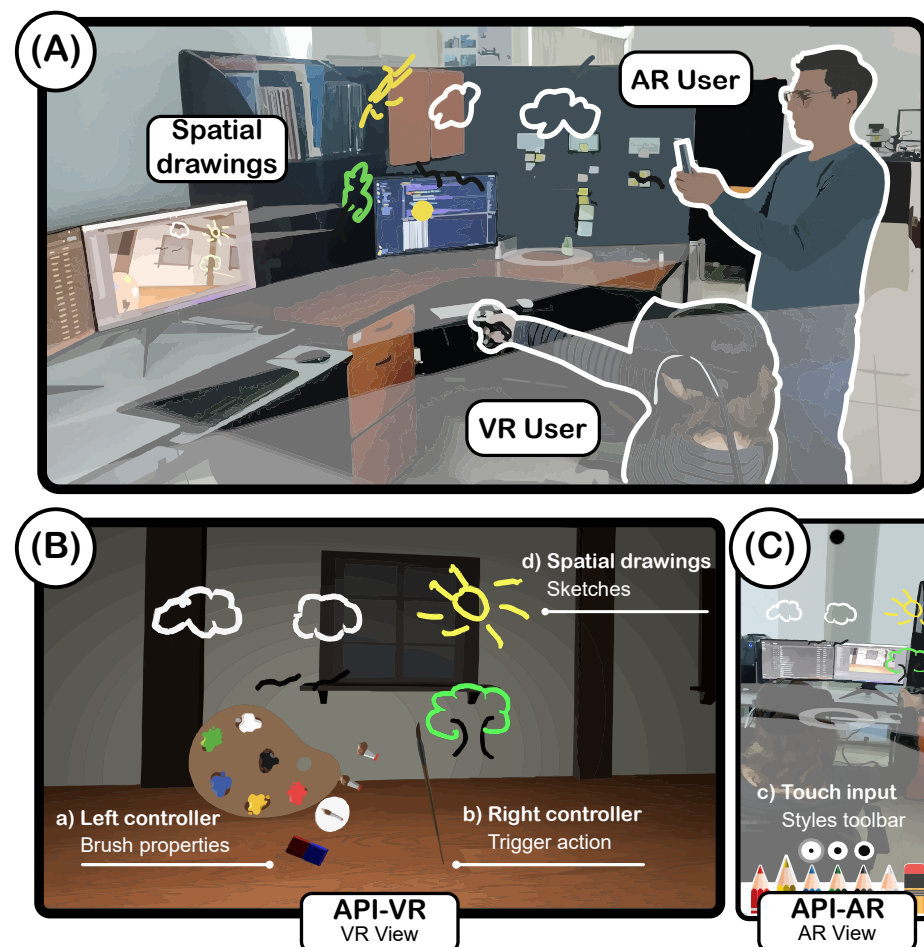


Figure 2. Overview of a co-located collaborative sketching scenario using SRVS-C. (A) illustrates the spatial arrangement of both participants, one using mobile AR and the other using VR, along with their respective virtual strokes rendered in a shared virtual scene. (B) shows the API-VR interface, where users draw in 3D space using handheld 6DoF controllers and adjust brush properties through an in-world palette. (C) displays the API-AR interface, where users sketch via touch input on an ARCore-enabled smartphone and customize strokes using a 2D overlay toolbar.

4. Experimental Methodology

The evaluation of the SRVS-C system was structured to assess two dimensions: system performance, focusing on communication and rendering latency; and collaborative consistency, encompassing synchronization fidelity and semantic coherence across heterogeneous devices. All tests were conducted in a controlled indoor environment with stable lighting and minimal physical occlusion to ensure consistent spatial tracking and experimental reproducibility.

4.1. Experimental Setup

The testbed consisted of two users: an ARCore-compatible Android smartphone running the API-AR client, and a PC-based VR system equipped with a 6DoF controller, running the API-VR client. Both participants operated in a shared physical space measuring approximately $3 \times 3 \times 3$ m, allowing for direct spatial colocalization throughout the experiment. At the beginning of each session, both devices were placed side by side at a common spatial origin, with the AR device immediately adjacent to the VR headset (see Figure 2). This ensured that both users experienced the collaborative environment from an equivalent starting point, facilitating consistent spatial anchoring and alignment of annotations across platforms.

Network connectivity was established via a LAN using an ASUS RT-AC5300 router (ASUSTek Computer Inc., Taipei, Taiwan) operating on the 5 GHz band. The router broadcast multiple SSIDs; however, all test devices were connected exclusively to a dedicated SSID configured for performance evaluation. No additional devices were connected to the test-specific network during measurements. According to the manufacturer's specification for the selected router, the data rate for the 802.11ac interface is up to 1734 Mbps. Communication was handled exclusively via persistent TCP sockets to a central SRVS-C server instance hosted on a PC machine (see Table 2), with no reliance on external or cloud-based infrastructure.

Table 2. Device specifications used in the collaborative testbed.

Device Type	Platform	Input Modalities	Description
Smartphone (Device A)	Android 11	Touchscreen, camera tracking	API-AR user running on Motorola One Hyper
VR Headset (Device B)	Windows 11	6DoF hand-held controller	API-VR user running on Oculus Rift and Alienware Aurora R7
PC	Mac OS 15.5	N/A	SRVS-C server instance on Mac Studio M2 Max

Within this setup, AR and VR users collaboratively created 3D annotations that remained persistently anchored in space, exercising the full communication and synchronization pipeline of SRVS-C while serving as a rigorous testbed for evaluating responsiveness, anchor consistency, and cross-device visual alignment under realistic network and tracking conditions.

4.2. Stroke Settings

To explore a range of collaborative scenarios, six geometric figures were defined as structured tracing templates, as shown in Figure 3. These templates were visually overlaid at the center of the participant's application view, serving as a guide within a drawing area of approximately 25×25 cm. Each participant was instructed to replicate the figures in 3D space by sequentially tracing the template's vertices in the indicated direction, following a protocol adapted from [34].

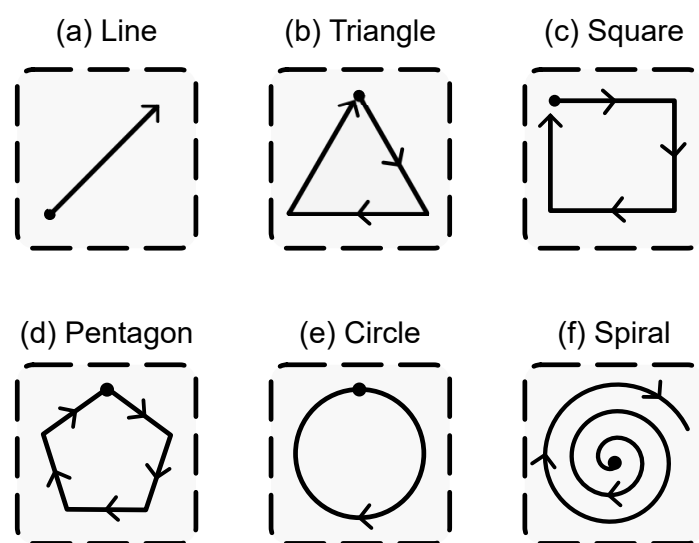


Figure 3. Six geometric figures used as tracing templates for testing the SRVS-C system. The templates include: (a) line, (b) triangle, (c) square, (d) pentagon, (e) circle, and (f) spiral. Participants traced these figures in 3D space, following the indicated direction of the arrows, to accurately replicate them. The black dots mark the starting point of each trace.

The evaluation was conducted bidirectionally: first, the VR device performed the drawing tasks while the AR device measured replication time, then the roles were reversed. This alternating protocol prevented platform-specific bias. Each figure was replicated 50 times on both devices, yielding 300 trials per transmission direction and a total of 600 annotated events.

The raw spatial data captured from user input was processed using Unity's `Vector3.Lerp` function for linear interpolation between the sampled points. This step generated a continuous and uniformly sampled stroke, thereby enhancing the smoothness and consistency of the visual representation. Strokes were transmitted as a continuous stream to emulate the dynamism of live user input, avoiding buffering artifacts from fixed-size batching.

While the visual template standardized the stroke path and size, the drawing speed was subject to natural variation between participants. Although efforts were made to encourage a consistent, moderate pace, it was not enforced as a controlled variable. However, since the primary latency metrics measure system processing time per stroke (after completion) rather than real-time streaming performance, the impact of drawing speed on the results is considered minimal. Throughout all trials, both devices recorded detailed timestamps for each stroke, including: message transmission time to server, server echo response reception time, and stroke reconstruction completion time.

4.3. Evaluation Metrics

To address clock skew across devices, a custom message–echo protocol was implemented that avoided the need for inter-device clock synchronization. Latency measurements were conducted in alternating transmission rounds, where only one device acted as the sender per trial. Upon receiving a message, the SRVS-C server immediately rebroadcasts it to all participants, including the original sender. The sender recorded both the transmission and echo reception timestamps using its own local clock, while the receiver independently logged the reception and reconstruction timestamps within its own clock domain. Each transmitted stroke was tagged with a unique message identifier generated at the sender side and preserved throughout the broadcast. During post-processing, these identifiers were used to match corresponding events across sender and receiver logs, enabling coherent alignment of timing data without relying on global time coordination. This design ensured that all latency metrics were derived from internally consistent clock domains, eliminating inaccuracies due to drift or offset between devices and enabling reliable point-to-point latency analysis.

For each transmitted stroke, the following timestamps were recorded independently on both sender and receiver devices:

Sender-side timestamps:

t_1 : Timestamp immediately before dispatching the stroke message to the SRVS-C server.

t_2 : Timestamp upon receiving the echoed stroke message from the server.

t_3 : Timestamp after rendering and anchor reconstruction of the echoed stroke.

Receiver-side timestamps:

t_4 : Timestamp upon receiving the rebroadcasted stroke message from the SRVS-C server.

t_5 : Timestamp after completing visual rendering and anchor reconstruction of the received stroke.

Based on these measurements, the following metrics were calculated for each geometric figure:

- **Number of spatial points** generated to complete each stroke P_n .

- **Server response time:** $\Delta t_s = t_2 - t_1$, capturing the full cycle of transmission.
- **Replication latency:** $\Delta t_r = t_5 - t_4$, isolating the user-side delay from message reception to full visual replication.
- **Figure recreation speed** in points/second $\Delta t_r / P_n$.

5. Experimental Results and Performance Analysis

This section presents the results of the latency and replication experiments performed under varying levels of stroke complexity. The resulting dataset was analyzed using descriptive statistics (mean, min, max, standard deviation). The analysis focuses on identifying trends in system responsiveness and processing overhead, evaluating the SRVS-C architecture's ability to support real-time collaboration.

5.1. System Latency

Table 3 reports the statistics of the system latency across all test conditions, categorized by two devices and the specific test figures. Device A corresponds to a smartphone running Android 11 (Motorola One Hyper), equipped with a Qualcomm Snapdragon 675 processor, 4 GB RAM, and a 6.5 inch FHD+ display, used for API-AR interactions. Device B designates a VR headset setup consisting of an Oculus Rift connected to an Alienware Aurora R7 desktop featuring an Intel Core i7-8700 CPU, 16 GB RAM, and NVIDIA GTX 1080 GPU, used for API-VR tasks.

Table 3. Average system latency across conditions.

Device	Condition	Mean (ms)	Min (ms)	Max (ms)	Std (ms)	Size (Kb)
A	P_{Line}	137.33	28	266	89.25	1.23
	$P_{Triangle}$	139.90	25	352	87.30	2.89
	P_{Square}	160.96	31	354	81.53	3.19
	$P_{Pentagon}$	163.30	30	431	88.65	3.24
	P_{Circle}	169.50	31	333	80.63	3.29
	P_{Spiral}	127.11	29	472	91.38	5.09
B	P_{Line}	57.71	10	124	37.85	1.84
	$P_{Triangle}$	73.20	11	189	40.95	4.61
	P_{Square}	63.51	10	178	40.44	3.95
	$P_{Pentagon}$	69.57	11	165	41.04	4.49
	P_{Circle}	80.45	11	321	53.33	6.72
	P_{Spiral}	90.16	23	234	57.87	11.07

The results reveal a fundamental performance disparity between the two hardware platforms. Device A exhibited not only higher average latencies but also significantly greater variability, as indicated by its large standard deviations (ranging from 80.63 ms to 91.38 ms) and wide min-max ranges for all figures. For instance, for the simple Line figure, the latency on Device A varied from 28 ms to 266 ms. This pattern suggests inherent volatility in the mobile platform's performance, likely due to background processes, thermal throttling, or variable resource allocation.

In contrast, Device B demonstrated consistently lower and more stable latencies. Its standard deviations were substantially smaller (ranging from 37.85 ms to 57.87 ms), and the min-max ranges were tighter across all figures, confirming the stability of the more powerful hardware setup. This performance gap is visually synthesized in Figure 4, which clearly illustrates the wider dispersion and higher median latencies for Device A compared to the tight clustering of results for Device B.

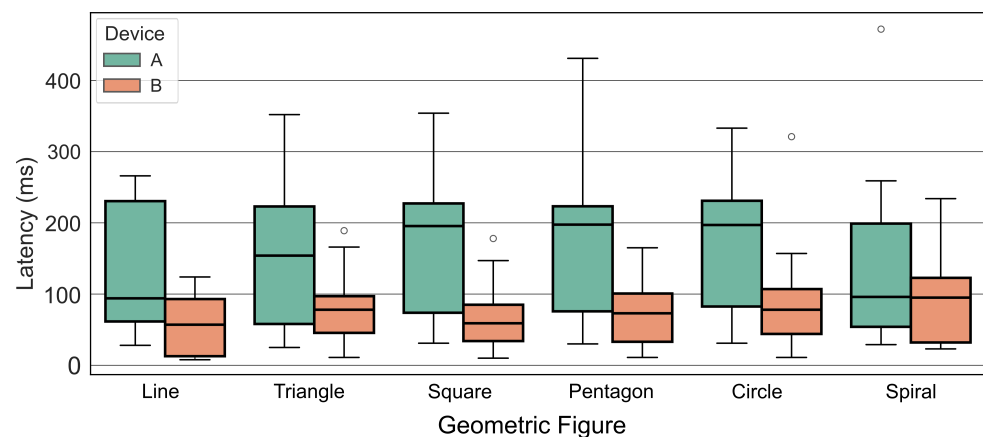


Figure 4. Boxplot of total system latency values for the six test conditions, depicting median latency, dispersion range, and outliers. Device A corresponds to a Motorola One Hyper smartphone (API-AR client), while Device B designates an Oculus Rift headset connected to an Alienware Aurora R7 (API-VR client).

5.2. Replication Latency

Table 4 presents the replication latency Δt_r , which captures the user-side delay from message receipt to the completion of visual rendering and anchor instantiation. This metric reflects the local processing overhead after data transmission is completed, using the same two devices described in Section 5.1.

Table 4. Average replication latency across test conditions.

Device	Condition	Mean (ms)	Min (ms)	Max (ms)	Std (ms)	Point Count
A	P_{Line}	1.94	0	76	9.85	14.01
	$P_{Triangle}$	3.50	0	49	10.29	36.64
	P_{Square}	2.21	0	76	10.18	41.60
	$P_{Pentagon}$	1.09	0	15	2.50	41.51
	P_{Circle}	3.07	0	62	11.56	42.49
	P_{Spiral}	1.68	0	35	5.10	67.40
B	P_{Line}	11.23	5	21	3.01	22.33
	$P_{Triangle}$	10.94	5	16	2.58	60.97
	P_{Square}	10.89	3	19	2.90	51.82
	$P_{Pentagon}$	10.73	6	18	2.18	59.38
	P_{Circle}	9.77	2	21	3.64	90.55
	P_{Spiral}	10.44	3	19	3.07	152.05

The results reveal a contrasting performance profile compared to the system latency analysis. While Device B demonstrated a superior performance in overall system latency, it consistently exhibited higher replication latencies across all test figures, with averages ranging from 9.77 ms to 11.23 ms. However, these values showed remarkable stability, as evidenced by low standard deviations (2.18–3.64 ms) and tight min–max ranges, indicating predictable rendering performance regardless of figure complexity.

Conversely, Device A achieved significantly lower average replication latencies (1.09–3.50 ms) but displayed substantial volatility in its performance. The standard deviations were notably higher (2.50–11.56 ms), and extreme outliers were observed across multiple figures, with maximum latencies reaching 76 ms for both the Line and Square figures. This pattern suggests that while the mobile device can often render strokes very

quickly, it is susceptible to unpredictable processing delays, likely due to resource contention in the android system.

Notably, the relationship between figure complexity (message size and point count) and replication latency is weak for both devices, as shown in Figure 5. Device B maintains stable latencies despite increasing point counts, while Device A shows no consistent trend between complexity and mean latency. This suggests that the rendering pipeline itself is efficient, but platform-specific scheduling and resource management practices significantly impact performance consistency.

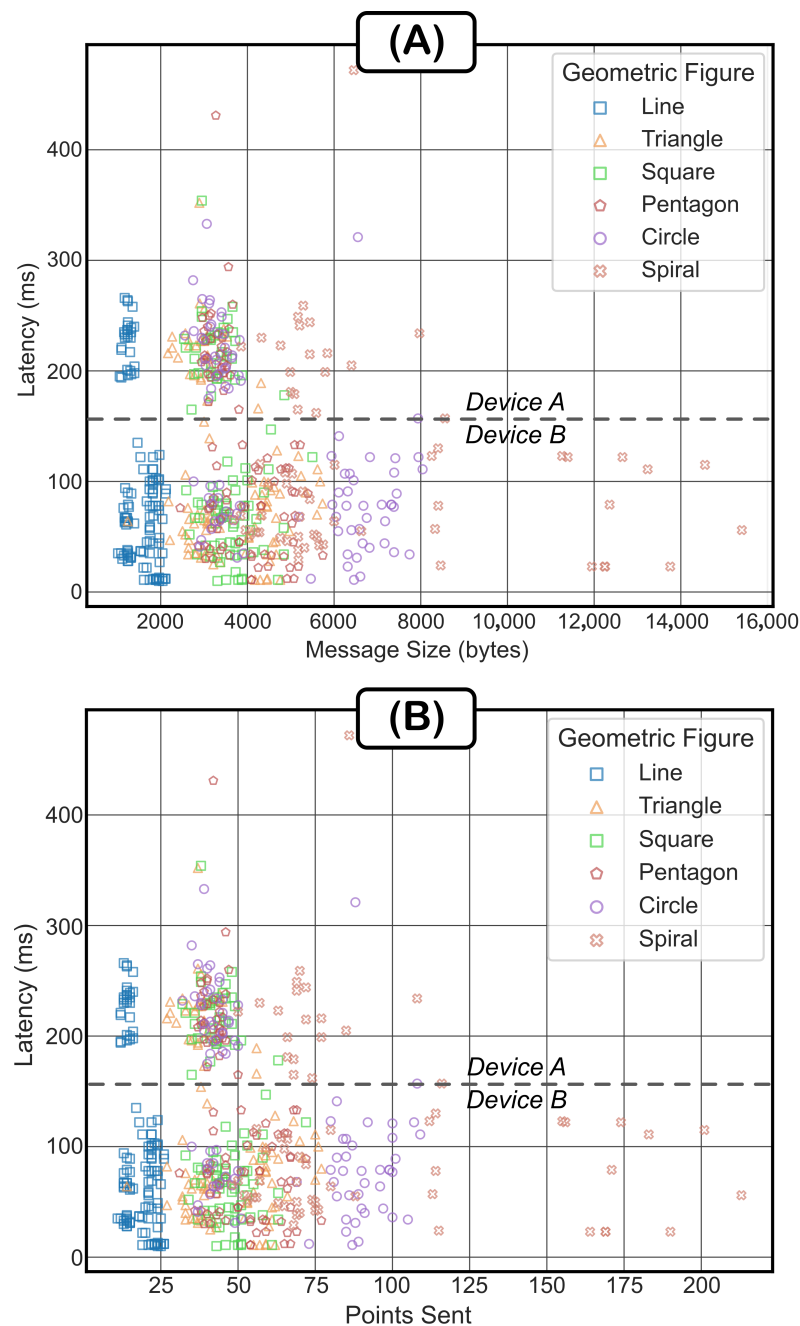


Figure 5. Latency distribution across geometric figures under varying message size and point densities. (A) illustrates the relationship between total system latency and message size for each of the six tracing templates. An approximate dividing line was added to visually separate the results obtained from two different devices: values above the line correspond to Device A, while values below the line correspond to Device B. (B) presents latency as a function of the number of spatial points transmitted per stroke.

All measured replication latencies remained well below 15 ms on average, confirming the system's ability to provide real-time visual feedback once data is received. The minimal values observed (often 0 ms for Device A) further support the efficiency of the local rendering processes.

5.3. Performance Analysis

The experimental results support the core design principles of the SRVS-C architecture, confirming its viability for real-time, co-located collaboration across heterogeneous AR/VR devices. The system successfully maintained average end-to-end latencies below 180 ms even for the most complex strokes, which is acceptable for continuous interactions like spatial sketching [35]. More importantly, the replication latency, critical for user-perceived responsiveness, remained exceptionally low (under 12 ms on average), ensuring immediate visual feedback upon data receipt.

The analysis reveals a fundamental performance asymmetry between the two platforms, which aligns with their respective hardware capabilities. Device B consistently delivered lower and more predictable system latency, demonstrating its ability to handle networking and computational loads efficiently. While the replication latency of Device B was marginally higher than that of Device A, it remained remarkably stable and was unaffected by the increasing stroke complexity. This characterizes it as a reliable, high-performance node ideal for data-intensive tasks. In contrast, Device A exhibited higher volatility in both system and replication latency. Despite often achieving lower average replication times, it was prone to significant outliers, reflecting the resource constraints and non-deterministic nature of a mobile platform. This performance profile underscores the importance of the SRVS-C's modular architecture, which allows each device to handle rendering and local processing according to its capabilities without compromising the shared state consistency.

This asymmetry reinforces the benefit of designing for interactive asymmetry as a first-class principle. The SRVS-C architecture does not enforce parity but leverages the strengths of each platform: enabling precise, immersive manipulation on powerful VR setups while allowing agile, gesture-based annotation from more accessible, albeit variable-performance, AR smartphones.

However, the results also highlight limitations for real-world scalability. The centralized server, while enabling consistent state propagation and spatial coherence, may become a bottleneck during high-frequency, multi-user interactions. Furthermore, the local anchor-based alignment, though avoiding cloud dependency, proved sensitive to tracking loss and occasionally led to perceptual drift. Finally, the performance volatility of mobile devices under load indicates a need for more sophisticated rendering pipelines with adaptive throttling or level-of-detail mechanisms to maintain fluency during content-heavy sessions.

6. Conclusions and Future Work

This work introduced the SRVS-C system, a centralized architecture for real-time, co-located collaboration across AR and VR platforms. The SRVS-C system is designed to operate entirely over local networks without reliance on proprietary cloud infrastructure. The system integrates a modular synchronization mechanism, a serialization-agnostic communication protocol, and platform-specific APIs for mobile AR and room-scale VR. Experimental latency evaluation under controlled conditions demonstrated that SRVS-C maintains low-latency performance and spatial consistency across devices, even under minimal and moderate workloads, and in the presence of interactive asymmetries.

A critical finding from the performance analysis is the significant impact of hardware resources on system behaviour. The results revealed a distinct performance asymmetry:

the VR setup (Device B), with its superior processing power, delivered consistently lower and more stable system latency, while the mobile AR device (Device A) exhibited greater volatility, particularly under load, despite achieving lower average replication times. This underscores the necessity of an architecture that accommodates, rather than attempts to equalize, heterogeneous devices capabilities.

The overall results reinforce SRVS-C's capability for XR collaborations, particularly in contexts where cloud connectivity is constrained or intentionally avoided. Furthermore, they validate the design principle of embracing interactive asymmetry, allowing devices to contribute according to their strengths.

In future work, we aim to address some technical limitations identified through our analysis and discussion. First, we aim to implement adaptive synchronization strategies (e.g., prioritized event queues, delta encoding, and dynamic rate control) to support larger user groups without compromising responsiveness. Specifically, we will explore client-side optimizations, such as level-of-detail rendering for resource-constrained devices, to mitigate the performance volatility observed on mobile hardware. Second, we will explore hybrid synchronization models that combine centralized coordination with decentralized fallback logic (e.g., gossip protocols) to enhance fault tolerance and scalability. These efforts will be guided by the goal of transforming the SRVS-C system from a controlled research prototype into a scalable framework for deployment in real-world hybrid XR collaboration.

Author Contributions: Conceptualization, G.A.M.G. and R.J.; methodology, G.A.M.G.; software, G.A.M.G. and R.J.; validation, G.A.M.G. and R.J.; data curation, G.A.M.G.; data analysis, G.A.M.G. and R.J.; resources, R.J., J.-P.I.R.-P. and U.H.H.B.; writing—original draft preparation, G.A.M.G.; writing—review and editing, G.A.M.G. and R.J.; review, J.-P.I.R.-P. and U.H.H.B.; visualization, G.A.M.G. and R.J.; supervision, J.-P.I.R.-P., R.J. and U.H.H.B.; project administration, U.H.H.B.; funding acquisition, J.-P.I.R.-P. and U.H.H.B. All authors have read and agreed to the published version of the manuscript.

Funding: This research was partially supported by the New Faculty Start-up Fund, Junior Faculty Research Grant from California State University, Fullerton, and Mexico's SECIHTI grant number 351478.

Data Availability Statement: The original contributions presented in this study are included in the article. Further inquiries can be directed to the corresponding author.

Conflicts of Interest: The authors declare no conflicts of interest.

References

- Kim, K.; Billinghurst, M.; Bruder, G.; Duh, H.B.L.; Welch, G.F. Revisiting Trends in Augmented Reality Research: A Review of the 2nd Decade of ISMAR (2008–2017). *IEEE Trans. Vis. Comput. Graph.* **2018**, *24*, 2947–2962. [\[CrossRef\]](#) [\[PubMed\]](#)
- Zhang, S.; Li, Y.; Man, K.L.; Yue, Y.; Smith, J. Towards Cross-Reality Interaction and Collaboration: A Comparative Study of Object Selection and Manipulation in Reality and Virtuality. In Proceedings of the 2023 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW), Shanghai, China, 25–29 March 2023; pp. 330–337. [\[CrossRef\]](#)
- Papadopoulos, T.; Evangelidis, K.; Kaskalis, T.H.; Evangelidis, G.; Sylaiou, S. Interactions in Augmented and Mixed Reality: An Overview. *Appl. Sci.* **2021**, *11*, 8752. [\[CrossRef\]](#)
- Çöltekin, A.; Lochhead, I.; Madden, M.; Christophe, S.; Devaux, A.; Pettit, C.; Lock, O.; Shukla, S.; Herman, L.; Stachoň, Z.; et al. Extended Reality in Spatial Sciences: A Review of Research Challenges and Future Directions. *ISPRS Int. J. Geo-Inf.* **2020**, *9*, 439. [\[CrossRef\]](#)
- Numan, N.; Steed, A. Exploring User Behaviour in Asymmetric Collaborative Mixed Reality. In Proceedings of the 28th ACM Symposium on Virtual Reality Software and Technology, New York, NY, USA, 29 November–1 December 2022; VRST '22. [\[CrossRef\]](#)
- Lee, Y.; Yoo, B. XR collaboration beyond virtual reality: work in the real world. *J. Comput. Des. Eng.* **2021**, *8*, 756–772. [\[CrossRef\]](#)
- Tümler, J.; Toprak, A.; Yan, B. Multi-user Multi-platform xR Collaboration: System and Evaluation. In Proceedings of the Virtual, Augmented and Mixed Reality: Design and Development, Virtual Event, 26 June–1 July 2022; Chen, J.Y.C., Fragomeni, G., Eds.; Springer International Publishing: Cham, Switzerland, 2022; pp. 74–93. [\[CrossRef\]](#)

8. Lin, C.; Sun, X.; Yue, C.; Yang, C.; Gai, W.; Qin, P.; Liu, J.; Meng, X. A Novel Workbench for Collaboratively Constructing 3D Virtual Environment. *Procedia Comput. Sci.* **2018**, *129*, 270–276. [CrossRef]
9. Villanueva, A.; Zhu, Z.; Liu, Z.; Peppler, K.; Redick, T.; Ramani, K. Meta-AR-App: An Authoring Platform for Collaborative Augmented Reality in STEM Classrooms. In Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems, New York, NY, USA, 25–30 April 2020; CHI '20; pp. 1–14. [CrossRef]
10. Huang, Y.; Wang, H.; Qiao, X.; Su, X.; Li, Y.; Dustdar, S.; Zhang, P. SCAXR: Empowering Scalable Multi-User Interaction for Heterogeneous XR Devices. *IEEE Netw.* **2024**, *38*, 250–258. [CrossRef]
11. Mourtzis, D.; Siatras, V.; Angelopoulos, J.; Panopoulos, N. An Augmented Reality Collaborative Product Design Cloud-Based Platform in the Context of Learning Factory. *Procedia Manuf.* **2020**, *45*, 546–551. [CrossRef]
12. An, J.; Lee, J.H.; Park, S.; Ihm, I. Integrating Heterogeneous VR Systems Into Physical Space for Collaborative Extended Reality. *IEEE Access* **2024**, *12*, 9848–9859. [CrossRef]
13. Mixed Reality Toolkit Organization. Mixed Reality Toolkit 3 for Unity. 2023. Available online: <https://github.com/MixedRealityToolkit/MixedRealityToolkit-Unity> (accessed on 29 July 2025).
14. PTC Inc. Vuforia Engine SDK. 2025. Available online: <https://developer.vuforia.com/> (accessed on 29 July 2025).
15. Meta Platforms, Inc. MR Utility Kit: Space Sharing. 2025. Available online: <https://developers.meta.com/horizon/documentation/unity/unity-mr-utility-kit-space-sharing> (accessed on 23 July 2025).
16. Murillo Gutierrez, G.A.; Jin, R.; Ramirez-Paredes, J.P.I.; Hernandez Belmonte, U.H. SRVS-C: Source Code for the Spatially Referenced Virtual Synchronization for Collaboration Algorithm. 2025. Available online: <https://github.com/MurilloLog/SRVS-C> (accessed on 25 september 2025).
17. Grønbaek, J.E.S.; Pfeuffer, K.; Velloso, E.; Astrup, M.; Pedersen, M.I.S.; Kjær, M.; Leiva, G.; Gellersen, H. Partially Blended Realities: Aligning Dissimilar Spaces for Distributed Mixed Reality Meetings. In Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems, New York, NY, USA, 23–28 April 2023; CHI '23. [CrossRef]
18. Fink, D.I.; Zagermann, J.; Reiterer, H.; Jetter, H.C. Re-locations: Augmenting Personal and Shared Workspaces to Support Remote Collaboration in Incongruent Spaces. *Proc. ACM Hum.-Comput. Interact.* **2022**, *6*, 1–30. [CrossRef]
19. Sereno, M.; Wang, X.; Besançon, L.; McGuffin, M.J.; Isenberg, T. Collaborative Work in Augmented Reality: A Survey. *IEEE Trans. Vis. Comput. Graph.* **2022**, *28*, 2530–2549. [CrossRef] [PubMed]
20. Bréhault, V.; Dubois, E.; Prouzeau, A.; Serrano, M. A Systematic Literature Review to Characterize Asymmetric Interaction in Collaborative Systems. In Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems, New York, NY, USA, 26 April–1 May 2025; CHI '25. [CrossRef]
21. Grandi, J.G.; Debarba, H.G.; Maciel, A. Characterizing Asymmetric Collaborative Interactions in Virtual and Augmented Realities. In Proceedings of the 2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR), Osaka, Japan, 23–27 March 2019; pp. 127–135. [CrossRef]
22. Zhang, S.; Li, Y.; Man, K.L.; Smith, J.S.; Yue, Y. Understanding User Experience, Task Performance, and Task Interdependence in Symmetric and Asymmetric VR Collaborations. *Virtual Real.* **2024**, *29*, 6. [CrossRef]
23. Agnès, A.; Sylvain, F.; Vanukuru, R.; Richir, S. Studying the Effect of Symmetry in Team Structures on Collaborative Tasks in Virtual Reality. *Behav. Inf. Technol.* **2023**, *42*, 2467–2475. [CrossRef]
24. Schäfer, A.; Reis, G.; Stricker, D. A Survey on Synchronous Augmented, Virtual, and Mixed Reality Remote Collaboration Systems. *ACM Comput. Surv.* **2022**, *55*, 1–27. [CrossRef]
25. Herskovitz, J.; Cheng, Y.F.; Guo, A.; Sample, A.P.; Nebeling, M. XSpace: An Augmented Reality Toolkit for Enabling Spatially-Aware Distributed Collaboration. *Proc. ACM Hum.-Comput. Interact.* **2022**, *6*, 277–302. [CrossRef]
26. Di Martino, B.; Pezzullo, G.J.; Bombace, V.; Li, L.H.; Li, K.C. On Exploiting and Implementing Collaborative Virtual and Augmented Reality in a Cloud Continuum Scenario. *Future Internet* **2024**, *16*, 393. [CrossRef]
27. Drey, T.; Albus, P.; der Kinderen, S.; Milo, M.; Segschneider, T.; Chanzab, L.; Rietzler, M.; Seufert, T.; Rukzio, E. Towards Collaborative Learning in Virtual Reality: A Comparison of Co-Located Symmetric and Asymmetric Pair-Learning. In Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems, New York, NY, USA, 29 April–5 May 2022; CHI '22. [CrossRef]
28. Li, Y.; Ch'ng, E.; Cai, S.; See, S. Multiuser Interaction with Hybrid VR and AR for Cultural Heritage Objects. In Proceedings of the 2018 3rd Digital Heritage International Congress (DigitalHERITAGE) held jointly with the 24th International Conference on Virtual Systems & Multimedia (VSMM 2018), Francisco, CA, USA, 26–30 October 2018; pp. 1–8. [CrossRef]
29. Kim, J.; Jeong, J. Design and Implementation of OPC UA-Based VR/AR Collaboration Model Using CPS Server for VR Engineering Process. *Appl. Sci.* **2022**, *12*, 7534. [CrossRef]
30. Chang, E.; Lee, Y.; Billingham, M.; Yoo, B. Efficient VR-AR communication method using virtual replicas in XR remote collaboration. *Int. J.-Hum.-Comput. Stud.* **2024**, *190*, 103304. [CrossRef]

31. Murillo Gutierrez, G.A.; Jin, R.; Ramirez Paredes, J.P.I.; Hernandez Belmonte, U.H. A Framework for Collaborative Augmented Reality Applications. In Proceedings of the Companion Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games, New York, NY, USA, 7–9 May 2025; I3D Companion '25. [CrossRef]
32. Google. ARCore Supported Devices. 2025. Available online: <https://developers.google.com/ar/devices> (accessed on 7 July 2025).
33. Close, A.; Field, S.; Teather, R. Visual thinking in virtual environments: Evaluating multidisciplinary interaction through drawing ideation in real-time remote co-design. *Front. Virtual Real.* **2023**, *4*, 1304795. [CrossRef]
34. Dudley, J.J.; Schuff, H.; Kristensson, P.O. Bare-Handed 3D Drawing in Augmented Reality. In Proceedings of the 2018 Designing Interactive Systems Conference, New York, NY, USA, 9–13 June 2018; DIS '18; pp. 241–252. [CrossRef]
35. Nielsen, J. Response Times: The 3 Important Limits. 1993. Available online: <https://www.nngroup.com/articles/response-times-3-important-limits/> (accessed on 23 July 2025).

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.