LinkSpace: A Toolkit for Linked Spatial Views in Virtual Reality

Kevin Linne
Technische Hochschule Mittelhessen
Giessen, Germany
TU Darmstadt
Darmstadt, Germany
kevin.linne@mni.thm.de

Kilian Reiner Stock
Technische Hochschule Mittelhessen
Gießen, Hessen, Germany
kilian.stock@mni.thm.de

Christin Katharina Kreutz
Technische Hochschule Mittelhessen
Gießen, Germany
ckreutz@acm.org

Jan Gugenheimer
TU Darmstadt
Darmstadt, Germany
jan.gugenheimer@tu-darmstadt.de

Martin Weigel
Technische Hochschule Mittelhessen
Giessen, Germany
martin.weigel@mni.thm.de

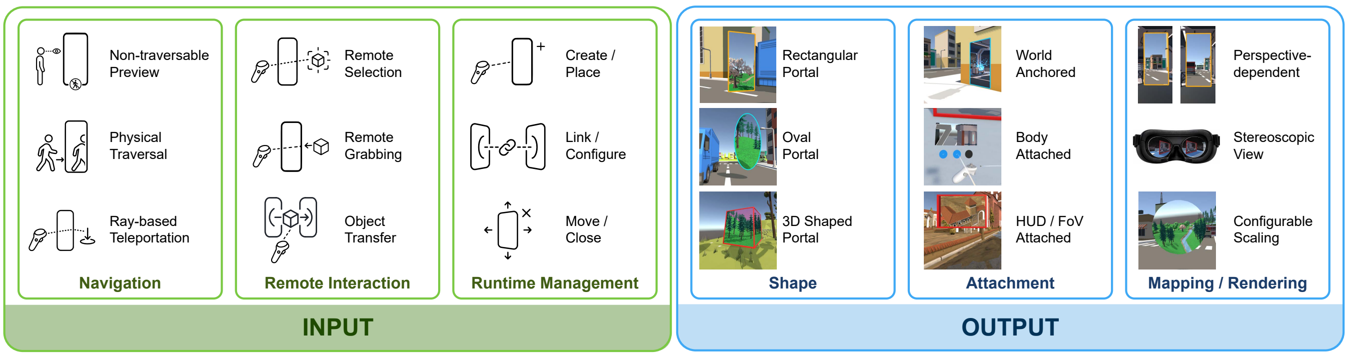


Figure 1: The design-space overview of LinkSpace summarizes how the toolkit structures configurable linked spaces that connect virtual environments, locations, and perspectives in VR. The input side covers navigation, remote interaction, and runtime management, while the output side covers shape, attachment, and mapping/rendering options.

Abstract

Virtual reality (VR) applications often isolate users in a single immersive context, although many tasks require access to distant environments, locations, or perspectives. Prior work addresses this need through portals, transitions, overlays, split views, and spatial previews, but these approaches are usually implemented from ground up for their specific use-case, making design alternatives difficult to compare and prototype. We use the term *linked spaces* to refer to such interactive connections between two spaces or perspectives. Using this terminology to summarize recurring requirements in prior work, we derive a design space for linked spaces and present LinkSpace, a Unity toolkit for creating configurable linked spaces with stereoscopic rendering, traversal, teleportation, remote selection and grabbing, object transfer, and runtime management. We demonstrate the toolkit's versatility and usability in an exploratory workshop with 18 computer science students enrolled in a master's program who used it to design and implement their own linked space prototypes. Participants created diverse concepts, rated the toolkit's usability as good (SUS: 75.1, $SD = 8.1$), and reported high perceived usefulness, adoption intention, and learnability, with qualitative feedback describing the toolkit as useful, flexible, simple, interesting, and intuitive.

CCS Concepts

• **Human-centered computing → Virtual reality**.

Keywords

Virtual reality, multi-environment interaction, linked spaces, VR toolkits, remote interaction, portals, Unity

ACM Reference Format:
Kevin Linne, Kilian Reiner Stock, Christin Katharina Kreutz, Jan Gugenheimer, and Martin Weigel. 2026. LinkSpace: A Toolkit for Linked Spatial Views in Virtual Reality. In *32nd ACM Symposium on Virtual Reality Software and Technology (VRST '26), November 16–18, 2026, Sendai, Japan.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3822517.3848674

ACM ISBN 979-8-4007-2811-2/2026/11
https://doi.org/10.1145/3822517.3848674

1 Introduction

Virtual reality (VR) allows users to interact with spatial information in immersive three-dimensional environments, yet many tasks are not confined to the user's current location. Users may need to inspect a distant area, monitor a remote process, compare alternatives, transfer objects between contexts, or keep one spatial reference visible while acting in another. Transition techniques can move users between contexts [14, 32, 36, 41, 42], but repeated switching can interrupt task flow and increase context loss, search effort, interaction overhead, memory recall problems [24], or cognitive residue [19]. Supporting interaction between spatially separated places therefore remains an important challenge for immersive systems.

Prior work has explored many ways to keep another spatial context accessible while users remain in their current one. Some techniques present other places as portals or transition objects [4, 17, 22], while others use photos, lenses, overlays, blended views, masks, or preview hubs [23, 27, 28, 44, 46, 48]. Related systems also represent remote contexts through proxies, distributed viewpoints, blended spaces, sliced spaces, and world-switching interfaces [13, 20, 33, 40, 49]. These examples support different kinds of spatial links: some mainly provide awareness of another place [44, 49], some let users move or switch between contexts [16, 17, 20, 22, 27], and others enable interaction with remote objects or dynamically placed portals [5, 40]. Additional work focuses on creating, adapting, or automatically arranging such links at the system level [4, 6, 30]. Together, these systems show the breadth of spatial linking in VR. However, they are usually described and implemented for specific visual forms, interaction techniques, and task settings, which makes broader alternatives difficult to compare or prototype.

We refer to these techniques as *linked spaces*: interactive connections between two virtual environments (VEs), distant locations, or perspectives on the same scene. Classical portals are one prominent instance, but the term also covers transitions, overlays, previews, proxies, and multi-environment arrangements. This shared vocabulary lets us describe recurring design questions across otherwise different systems: how a connection is represented, where it is placed, how users interact with it, how coordinate systems are mapped, and how VR-specific rendering constraints such as stereoscopy, clipping, and performance are handled [4]. Based on this framing, we derive a design space for linked spaces. It organizes requirements from prior work around input dimensions of navigation, remote interaction, and system control, followed by output dimensions of visual representation, placement, and rendering.

HCI toolkits turn recurring implementation work into reusable abstractions and thereby make interaction techniques easier to create, adapt, and study. Prior toolkit work demonstrates this role for physical prototyping [21], proximity-aware interaction [31], and toolkit evaluation [25]. VR and XR toolkits follow a similar pattern. Some provide general infrastructure for spatial interaction and VR research [47, 50]. Others support study infrastructure such as experiment control, remote studies, networked multi-user studies, or social VR experiences [11, 18, 26, 39]. Further toolkits address more specialized prototyping needs, including mixed-reality analytics, in-situ spatio-temporal analysis, accessibility, hand redirection, and haptic proxies [12, 15, 35, 51, 52]. Existing portal toolkits and implementations provide important foundations through reusable portal packages [3] and portal rendering and traversal techniques [4], and asset-store packages provide ready-made portal effects such as Pocket Portal VR [1] and Portals for VR [2]. Compared with these portal-specific implementations, LinkSpace does not primarily target optimized portal rendering. Instead, it provides a broader abstraction for configuring connections between environments, locations, and perspectives, including their representation, placement, interaction capabilities, and runtime management. We therefore present LinkSpace[1], an open-source Unity toolkit for implementing configurable linked spaces in VR. The toolkit translates the design-space dimensions into reusable mechanisms for stereoscopic rendering, controller-based interaction, traversal and teleportation, remote object interaction, object transfer, and runtime management of linked spaces.

To explore the toolkit's use in practice, we conducted an exploratory 3 hour workshop with 18 participants from a computer science course in a master's program. The workshop consisted of an introduction to the toolkit and general VR development, a design brainstorming phase in which participants developed and sketched linked-space application ideas, a prototype implementation phase in Unity, and a final evaluation phase using questionnaires and qualitative feedback. The toolkit achieved a good usability rating, with a SUS [10] score of 75.1 ($SD = 8.1$). Custom questionnaire results indicated high perceived usefulness and adoption intention, while debugging support received lower ratings. Qualitative feedback emphasized that creating and connecting linked spaces was simple and flexible, but also identified documentation, debugging, and perspective-dependent interaction through links as areas for improvement.

This paper makes three contributions:

(1) We derive a literature-grounded design space for linked spaces, a term we use to subsume techniques that connect VEs, locations, or perspectives in VR. The design space organizes recurring requirements from prior work around input dimensions such as navigation, remote interaction, and system control, and output dimensions such as shape, placement, and rendering.
(2) We present LinkSpace, a Unity toolkit for VR that implements the design space for configurable linked spaces.
(3) We report an exploratory workshop with 18 participants prototyping in LinkSpace, providing initial evidence on usability, perceived usefulness, flexibility, and remaining implementation challenges.

2 Analysis & Concept

We analyze prior work as a design space for linked spaces rather than as separate families of portals, transitions, previews, or overlays. The analysis first considers the *input* side, describing how users navigate, interact with remote content, and manage links, and then the *output* side, describing how links are represented, placed, and rendered, as summarized in Figure 1. Together, these dimensions form the design space we target with LinkSpace and are summarized as six concept goals below.

2.1 Input

2.1.1 Navigation and Transition. Navigation describes whether and how users can move between linked contexts: some links do not let users move to the linked context, while others support traversal, teleportation, or complete scene transitions. Portals have been used for reorientation during real walking [17], VR-specific transition

[1]The toolkit is available at http://go.int-lab.de/linkspace.

metaphors such as portals and orbs can be preferred over generic transitions [22], and task-driven cross-reality transitions benefit from short and efficient transitions [16]. More generally, portals and previews often serve as entry points for traversal, switching, or teleportation, including technical portal implementations [4], pointing-based portals [5], portable masks [28], and 360° preview hubs [27].

Goal 1: Make navigation behavior configurable per linked space, including physical traversal, ray-based teleportation, scene switching, and non-traversable previews.

2.1.2 Remote Interaction.

Remote interaction concerns whether users can target, select, grab, or transfer objects in a linked context without first moving there. Selection and manipulation are central 3D user-interface categories alongside navigation and system control [9], and 3D selection depends strongly on pointing, feedback, depth, occlusion, and rendering conditions [7]. In linked contexts, these issues become cross-spatial: Poros enables interaction with distant objects through nearby proxies [40], Point & Portal supports action at a distance through dynamically placed portals [5], SliVR enables direct selection from VE previews [27], OVRlap keeps perception across locations while tying interaction to one active viewpoint [44], Blending Spaces studies object transitions between realities [13], and Photoportals support navigation, manipulation, collaboration, and review through spatial references [23]. Adaptive Portals further show that portal-based interaction can support reaching and remote task management [6].

Goal 2: Map interaction rays, object transforms, velocities, and interaction state across linked spaces so users can target, select, grab, and transfer remote objects.

2.1.3 System Control and Runtime Management.

System control covers how linked spaces are created, configured, adjusted, and removed by users or applications. Prior work studies portal creation interactions and technical portal integration in VR [4], portals that adjust in real time to users' needs [6], and automatic portal layouts computed from visibility and walkability criteria [30]. Multi-environment systems such as Off-The-Shelf's see-through masks [28] and SliVR's preview hub [27] additionally show that users may need to manage several links at once by selecting, focusing, or switching between multiple VE previews.

Goal 3: Expose reusable operations for creating, linking, unlinking, configuring, moving, and closing linked spaces at runtime.

2.2 Output

2.2.1 Shape and Visual Representation.

Prior work shows that linked spaces can take many visual forms. Classical portals often use planar openings that behave like doorways or windows into another location, including reorientation portals [17], portal rendering and creation systems [4], and automatically placed navigation portals [30]. Other systems use virtual photos and videos as spatial references [23], configurable proxies of distant spaces [40], oval see-through masks as portable previews of other VEs [28], or life-sized 2D previews in a 360° hub [27]. Related visualization concepts further broaden this space through through-the-lens techniques [46], worlds-in-miniature [45], and Worlds-in-Wedges [34], where alternate views are shown through a lens, miniature representation, or wedge-shaped view.

Goal 4: Support arbitrary visual forms for linked spaces, including flat shapes, portals, previews, masks, and spatial 3D representations.

2.2.2 Attachment, Placement, and Layout.

Linked spaces also differ in where they are attached and how they are arranged. Some systems place portals in the world and let users approach or walk through them, as in reorientation portals [17], portal rendering and creation systems [4], or automatic portal layouts [30], while others bring remote content closer to the body or interaction space, as in Poros' reachable proxies [40] or Adaptive Portals' self-adjusting portals for extending reach [6]. Views can also be attached to field-of-view regions or perceptual layouts rather than fixed world positions, as in OVRlap's overlapping viewpoints [44], MultiSoma's distributed perception across bodies [33], or SliVR's arrangement of multiple VE previews around the user [27]. More general layout work in immersive spaces connects overview to 3D multiview layouts [43], spatial memory to display layout [29], physical-monitor replacement to AR virtual monitors [38], and window management to virtual display layouts [37].

Goal 5: Support linked spaces that can be world-anchored, body-attached, controller-attached, object-attached, field-of-view aligned, or arranged as part of a larger spatial layout.

2.2.3 View Mapping and Rendering Constraints.

Visual representation and placement only define where a linked space appears; making it behave as a spatial connection also requires a mapping between local and target contexts. Through-the-lens techniques describe interaction between two viewpoints, where one view represents the surrounding environment and another manipulated viewpoint is displayed in a dedicated window [46]. Portal rendering makes this mapping more demanding in VR because the rendered view should remain consistent with the user's head pose and stereoscopic perception, while also addressing clipping, physics interactions, and performance [4]. View mapping also includes scale: some linked spaces should preserve life-sized spatial scale, while others intentionally compress a larger context into a smaller representation, as in worlds-in-miniature, miniature previews, or overview displays [45].

Goal 6: Keep linked spaces visually and interactively coherent across viewing, navigation, teleportation, and object transfer, while allowing them to either preserve spatial scale or compress larger contexts into smaller representations.

3 Implementation

We implemented LinkSpace in Unity using the Universal Render Pipeline (URP), OpenXR, the Unity Input System, and the XR Interaction Toolkit. The implementation translates the design space from the six concept goals into reusable Unity components for linked spaces. Rather than treating portals as the only primitive, LinkSpace separates the linked-space endpoint, its visual representation, and the interaction capabilities enabled for it. This section follows the structure of the concept: core toolkit components, input-side navigation, remote interaction, and system control, and output-side representation, placement, and view mapping.

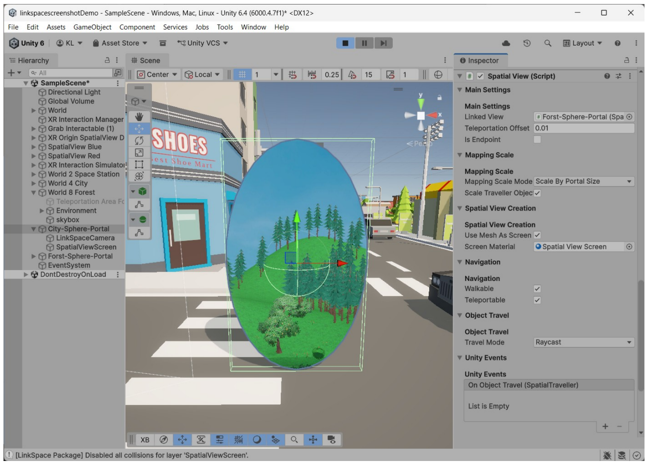


Figure 2: Unity Inspector view of the `SpatialView` component. Developers can attach the script to a Unity object and configure the linked-space instance directly, including its `linkedView`, connection direction, supported interaction modes, navigation behavior, view-scaling behavior, and runtime management options.

3.1 Core Components

The central component is the `SpatialView` script. It represents one endpoint of a linked space and stores a reference to its `linkedView`, the local coordinate frame of the link, the rendered view surface, and the trigger volume used for traveller tracking. Developers can attach the script to a Unity object and configure linked-space properties in the Unity Inspector. As shown in Figure 2, these properties include the linked space, connection direction, enabled navigation and interaction modes, view-scaling behavior, and runtime management options. This keeps configuration in the regular Unity workflow while allowing different instances to act as windows, previews, traversal surfaces, remote-interaction views, or combinations of these roles.

Objects that can move through a linked space use a traveller abstraction. The base class is `SpatialTraveller`; specialized classes such as `PlayerSpatialTraveller` and `GrabSpatialTraveller` handle the XR player and grabbable objects. This separates generic crossing logic from object-specific behavior such as player locomotion, controller ownership, or object transfer.

3.2 Input: Navigation, Remote Interaction, and System Control

The input side implements navigation, remote interaction, and runtime management as optional capabilities of a `SpatialView`, corresponding to Goal 1, Goal 2, and Goal 3. A linked space hence combines different capabilities, ranging from non-traversable previews to physical traversal, routed teleportation, remote selection and grabbing, and runtime system control, reflecting the design-space requirement.

For physical traversal, `SpatialTraveller` instances are tracked with trigger volumes, addressing Goal 1. When a traveller crosses the view plane, the toolkit applies the view-to-view point and rotation mapping to move it to the linked side. Travelers can create a visual clone while intersecting a linked space; slicing shaders clip the original and clone against opposite sides of the view plane so that users or objects appear to pass through continuously while only one physical object remains authoritative.

For ray-based teleportation, a custom ray interactor detects whether the teleport ray intersects a linked-space screen, addressing Goal 1. If so, the ray origin and direction are transformed through the linked space and the teleport arc is evaluated in the target context. Remote selection and grabbing use the same principle: a custom near–far interactor continues the user's interaction ray through the linked space and can trigger XR selection behavior in the target context, addressing Goal 2. For object transfer, the toolkit transforms position, rotation, linear velocity, and angular velocity when the controller-object connection or the held object crosses the view plane, also addressing Goal 2 and Goal 6.

System control is implemented through a lightweight runtime management layer, addressing Goal 3. `LinkSpaceManager` exposes operations for creating, linking, moving, and destroying `SpatialView` instances, while helper components such as `SimpleSpatialViewPlacer` provide ray-based placement previews and interaction shortcuts. These controls were used during development and in the workshop to instantiate linked spaces, link pairs, move existing instances, close instances, and spawn test objects. The layer demonstrates the system-control requirement in Goal 3, but task-specific applications can replace it with custom menus, editor tools, automatic layout strategies, or higher-level authoring interfaces.

3.3 Output: Representation, Placement, and View Mapping

The output side renders the target context onto a screen surface. The default representation is a planar Unity object that can be placed, moved, attached to other objects, or instantiated at runtime. This supports world-anchored windows, body-attached previews, inventory-like views, and traversal surfaces, addressing Goal 5, while the separation between `SpatialView` and its screen object keeps the visual representation replaceable as required by Goal 4.

All output rendering is based on a shared mapping between two `SpatialView` instances, addressing Goal 6. Each view is represented by its root transform: the local x and y axes span the view plane, the local $+z$ axis defines the front-facing normal, and local z is the signed depth used for crossing tests. To map content from view A to view B, the toolkit first transforms it into the local frame of A and then re-expresses it in the root frame of B:

$$T_{A \to B} = T_B \cdot T_A^{-1}$$

Points such as camera positions use the full view-to-view transform, while directions, velocities, and rotations use the corresponding vector or rotational mapping.

View scale is configurable per linked space, also addressing Goal 6. In scale-preserving mode, the linked environment is rendered at consistent world scale independently of the visible surface size, supporting life-sized windows and traversal-oriented views. In surface-size-dependent mode, the represented environment scales

Table 1: Rendering performance for simultaneously active `SpatialViews` on a standalone Meta Quest 3. Statistics were calculated over stable 30 s intervals using a fixed viewpoint from which all active views were visible simultaneously.

Views	GPU median (ms)	GPU P95 (ms)	FPS median	Stale frames/s median
2	4.0	4.5	72	0
4	6.2	6.7	72	0
6	7.4	8.5	69	3
8	9.6	10.0	54	18

with the dimensions of the view surface, allowing small views to act as miniature overviews or worlds-in-miniature.

For each active `SpatialView`, the toolkit creates left- and right-eye render textures and positions a linked-space camera by transforming the user's eye pose through the linked space. The resulting images are assigned to the `SpatialViewScreen` material, and the `LinkSpaceView` shader selects the correct texture for the current eye. The toolkit also applies an oblique near clipping plane at the linked-space plane, removing geometry between the virtual camera and the target-side surface. Together, stereoscopic rendering, view-dependent camera placement, clipping, and configurable scaling address the VR rendering requirements identified in the analysis and in prior work on portal rendering [4].

Overall, the implementation covers all six concept goals: configurable navigation, remote interaction, system control, replaceable representations, flexible placement, shared view mapping, stereoscopic rendering, and object transfer.

3.4 Rendering Performance

Rendering performance depends substantially on the content and complexity of the linked scene. We therefore characterized the toolkit's demo scene on a standalone Meta Quest 3 using a fixed viewpoint and constant rendering settings, varying the number of simultaneously active `SpatialViews` as summarized in Table 1. All active views were simultaneously visible within the frame throughout each measurement. For each condition, we analyzed a stable 30 s interval using the Meta OVR Metrics Tool. The headset operated at 72 Hz with dynamic resolution, Application SpaceWarp, and foveated rendering disabled. App GPU time increased gradually but remained below the nominal GPU frame budget in all tested conditions. LinkSpace maintained 72 FPS without stale frames for up to four active views and still achieved a median of 69 FPS with six views. Substantial frame-delivery degradation appeared only in the dense eight-view condition. In this demo scene, the results therefore demonstrate that LinkSpace can render several simultaneous views at high frame rates on standalone hardware, while unusually dense layouts may require application-specific optimization.

4 Exploratory Workshop

We conducted an exploratory hands-on workshop with LinkSpace. The goal was to understand whether participants could use the toolkit to design and implement diverse linked-space concepts within a short prototyping session. We also wanted to collect feedback on usability, perceived usefulness, flexibility, and implementation challenges.

4.1 Workshop Design

The workshop lasted three hours and consisted of four phases: (1) introduction, (2) ideation, (3) prototyping, and (4) evaluation. First, participants received a 30-minute introduction to LinkSpace. The introduction covered the motivation behind the toolkit, its main concepts, and its use in Unity. We demonstrated how to create, link, configure, and interact with portals.

After the introduction, participants received a list of concepts and application areas for inspiration. The list included *body-attached portals*, *dynamic or moving portals*, *unusual portal shapes*, *remote interaction*, *portal information displays*, *system control*, *portal-based navigation*, *multiple portals*, and *bidirectional object transfer*. These terms were intended to stimulate ideas without prescribing concrete applications.

Participants then formed groups of three. Each group developed an application idea for linked spaces and sketched the intended concept on paper (see I1–I6 in Figure 3). In the following implementation phase, groups had 100 minutes to implement their concept. They created a new Unity project, initialized it for VR development, imported the provided LinkSpace Unity package, and used it to build their prototype. Figure 3 illustrates the overall workshop process and links each group's ideation sketch to the resulting prototype.

4.2 Participants and Apparatus

Eighteen computer science students enrolled in a master's program participated in the workshop. We refer to individual participants as S1–S18 when reporting quotes. Participants were between 22 and 31 years old ($M = 24.8$, $SD = 2.2$). Twelve participants identified as male and six as female. Participants had general programming experience, but limited prior experience with Unity and VR development. On average, they reported 5.7 years of programming experience ($SD = 1.8$), 1.0 years of 3D or game development experience ($SD = 1.4$), 0.6 years of Unity experience ($SD = 1.0$), and 0.2 years of VR development experience ($SD = 0.7$). Fifteen participants reported no prior VR development experience. This participant group was therefore suitable for exploring whether the toolkit can be used by developers with general programming knowledge but limited to no VR-specific experience. Four of the six groups worked with a Windows computer and a Meta Quest 3 headset, while two groups chose to work on macOS and use the Unity XR simulator.

4.3 Data Collection and Analysis

After the implementation phase, participants completed several questionnaires (see supplementary materials). These included the System Usability Scale (SUS), a custom Likert questionnaire, open-ended feedback questions, and a demographic questionnaire. The custom Likert questionnaire used a five-point scale and covered learnability, flexibility, debugging, usefulness, adoption intention, and expected efficiency with more experience. The workshop was conducted in German, and all questionnaire materials, open-ended

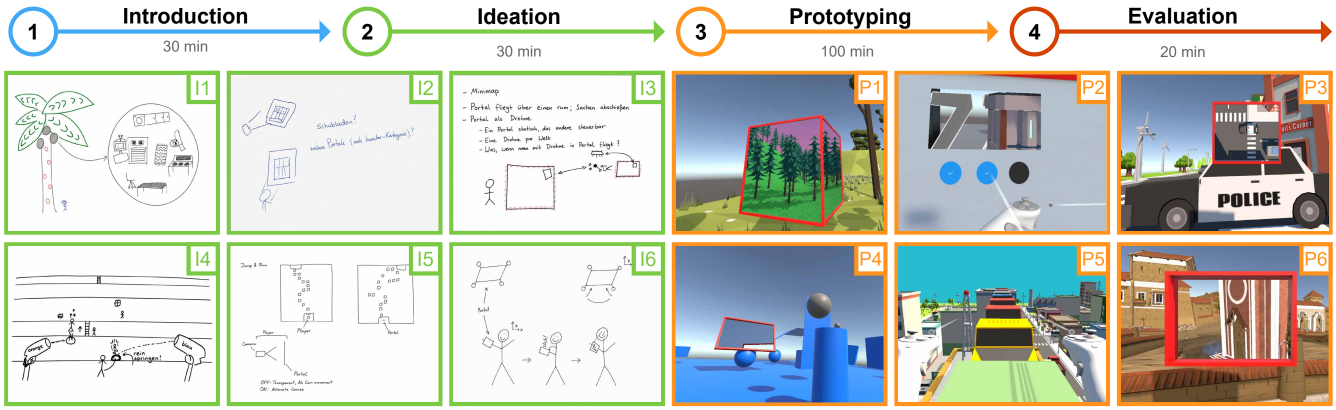


Figure 3: Workshop process and resulting prototypes. The timeline summarizes the four workshop phases: introduction, ideation, prototyping, and evaluation. The green panels show the six ideation sketches (I1–I6), and the orange panels show the corresponding prototype outcomes (P1–P6): Pocket Universe (I1/P1), Inventory System (I2/P2), Minimap (I3/P3), Portal Gun (I4/P4), Parkour (I5/P5), and Virtual Drone (I6/P6).

responses, and participant quotes were translated into English for reporting in this paper.

The open-ended questions were designed to capture both general impressions and more specific feedback. The first two questions were intentionally phrased in a neutral way. Participants were asked to name three adjectives they associated with the toolkit and to describe their general first impression. These questions were intended to capture an initial overall perception without directing attention to particular strengths or weaknesses. The remaining three questions asked more explicitly about perceived strengths, perceived weaknesses, and suggestions for improvement. Together, these questions were intended to provide a structured view on how the toolkit was experienced after the workshop.

We analyzed the quantitative data descriptively (see Figure 4) and report means and standard deviations for SUS and Likert scores. The open-ended responses were reviewed qualitatively and grouped according to three perspectives: general first impressions, perceived strengths and weaknesses, and suggested improvements. The adjective question was additionally summarized visually in a word cloud to provide an overview of the terms most frequently associated with the toolkit.

5 Results

The workshop produced several linked-space concepts within the available prototyping time. The results provide initial evidence that LinkSpace can support rapid implementation of portal-based ideas, even when participants have limited prior VR development experience.

5.1 Prototype Outcomes

The six groups implemented a range of applications that used different aspects of the toolkit. The resulting prototypes are linked to the outcome panels P1–P6 in Figure 3:

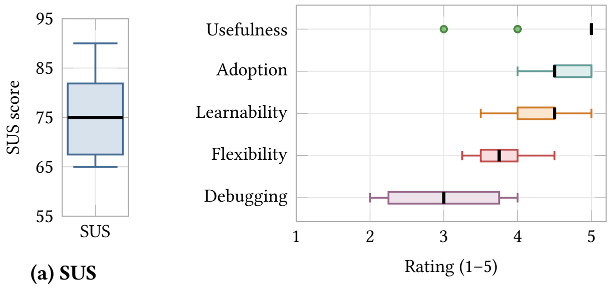
- **P1: Pocket Universe** connected distant places through two portals with different sizes and explored the idea of a distorted pocket universe;
- **P2: Inventory System** used a wrist-mounted body-attached portal through which users could store objects at a remote location;
- **P3: Minimap** showed a top-down perspective in the user's HUD;
- **P4: Portal Gun** created portals on distant objects and automatically linked them;
- **P5: Parkour** let users move across a chain of vehicles while parts of the view were obstructed;
- **P6: Virtual Drone** showed the perspective of a remote-controlled drone as an extension of the user's field of view.

These outcomes covered several of the inspiration prompts, including body-attached portals, dynamic portals, unusual shapes, remote interaction, portal-based navigation, multiple portals, and bidirectional object transfer. Although the prototypes were not complete applications, they show that participants were able to move beyond the provided example scene and implement their own concepts within less than two hours.

5.2 Questionnaire Results

The toolkit achieved a SUS score of 75.1 ($SD = 8.1$; range: 65–90). According to common interpretations of SUS scores, this corresponds to a good level of perceived usability [8].

The custom Likert questionnaire also showed generally positive ratings. On a scale from 1 to 5, the highest-rated dimension was usefulness ($M = 4.78$, $SD = 0.55$), followed by adoption intention ($M = 4.61$, $SD = 0.37$) and learnability ($M = 4.31$, $SD = 0.39$). Flexibility received a positive but somewhat lower rating ($M = 3.75$, $SD = 0.39$). Debugging received the lowest rating ($M = 3.00$, $SD = 0.77$). Overall, these results suggest that participants perceived the toolkit as useful and learnable, while understanding and resolving errors remained more difficult.



(a) SUS

(b) Custom Likert dimensions

Figure 4: Distribution of quantitative evaluation results across participants (N = 18). (a) System Usability Scale (SUS) scores. (b) Ratings for the custom questionnaire dimensions.

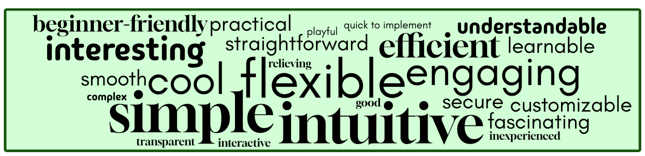


Figure 5: Word cloud based on the adjectives participants used to describe LinkSpace. Larger words indicate more frequent mentions.

5.3 General First Impressions

The open-ended responses indicate a largely positive initial impression. Participants frequently described the toolkit with terms such as *useful*, *flexible*, *simple*, *interesting*, and *intuitive*. Figure 5 summarizes the adjectives named by participants.

The more general first-impression question showed a similar pattern. Several participants stated that the toolkit was easy to approach after the introduction and that it enabled quick progress toward visible results. For example, S4 stated that “using the basic functions was easy to understand and quick to implement independently”. S7 similarly noted that “after the joint introduction, it was very easy to implement custom portals”. Even participants with limited Unity experience reported being able to use the toolkit productively: “Despite having little experience with Unity, I was able to use the [toolkit] relatively quickly and implement basic functions” (S9). Others emphasized the toolkit's potential as a basis for further development. S18 wrote that “creating portals was very easy and it was possible to build well on top of it”.

5.4 Perceived Strengths and Weaknesses

Participants most often described the ease of creating and linking portals as a strength. S9 wrote that “linking the portals is very intuitive”, while S8 stated that “coupling portals was very simple”. Other participants emphasized that the toolkit reduced implementation effort for core portal functionality. For example, S17 highlighted “how easy it is to create portals”, and S18 described the portal as “only one script”, which “makes the use very easy”. S16 similarly reported that “the basic functionalities were easy to use”, although “a few more specific use cases required some debugging and testing”. The same participant also noted that “it was very easy to integrate and connect portals into the worlds” and that “teleportation worked as expected”. Taken together, these responses suggest that the toolkit was particularly successful in providing quick access to core functionality such as portal creation, linking, viewing, and traversal.

The main weaknesses concerned setup effort, debugging, and more complex interaction through portals. Some of these issues were related to the development environment rather than to the toolkit itself. For example, S8 described “the initial integration of [the toolkit]” as difficult, but added that this was “mainly due to a lack of Unity experience”. S18 referred to challenges in understanding “how to use the Quest controls”. S16 reported that “grabbing objects through portals was difficult to implement, because from some perspectives one did not grab at the expected position”. As a workaround, the participant used larger objects. This aligns with the lower debugging rating in the Likert questionnaire and indicates that advanced interaction cases were harder to understand and control than basic portal operations.

5.5 Suggested Improvements

The improvement suggestions were consistent with the reported weaknesses. Participants most frequently called for clearer documentation and better support for debugging. S8 suggested that the “README could be more detailed”, while S16 proposed to “adapt grabbing through portals to the perspective”. Other suggestions concerned additional functionality or robustness. For example, S5 suggested “jumping” and “snap to HUD”, while S15 pointed to remaining rendering issues and wrote that “objects still flicker”.

Overall, the qualitative feedback complements the questionnaire results. The first impressions indicate that the toolkit was generally perceived as useful, flexible, and approachable. The more explicit feedback shows that its main strengths lie in fast portal prototyping and simple linking, while the main challenges concern setup, debugging, and more advanced interaction behavior.

5.6 Post-Workshop Toolkit Improvements

The workshop feedback directly informed subsequent improvements to LinkSpace. Because several comments pointed to setup effort, documentation gaps, and uncertainty about available functionality, we extended the README with a more detailed getting-started guide, feature descriptions, and explanations of the main scripts. We also added a dedicated description of controller operation to make basic interaction behavior more explicit for first-time users, especially when working with Quest controllers.

We further investigated the reported issue with grabbing objects through portals. This issue was observed by one participant only and could not be reproduced in subsequent tests. We therefore did not treat it as a confirmed implementation bug, but kept it as an indicator that perspective-dependent interaction through linked spaces requires clearer debugging support and more explicit documentation.

6 Discussion and Limitations

The workshop suggests that LinkSpace can support rapid prototyping of linked spaces. Participants with limited Unity and VR development experience were able to create diverse prototypes within a short implementation phase.

6.1 Design Challenges

Linked spaces introduce several application-dependent design challenges:

- *Visual presentation:* Persistent or body-attached views may occlude task-relevant content, compete for attention, or overlap with one another.
- *Rendering:* Our performance characterization indicates that several simultaneous views are feasible on standalone hardware: the target frame rate was maintained for up to four views and remained close with six, while substantial degradation appeared only with eight. Dense linked-space layouts may therefore require trade-offs in view number, resolution, visibility, or update behavior.
- *Spatial interaction:* Mappings between different scales or perspectives can complicate spatial understanding and remote interaction.

Their relevance depends on the application, surrounding scene, and user task. LinkSpace therefore exposes options for placement, attachment, scaling, interaction, and runtime management, while leaving application-level policies to developers. Automatic layout, adaptive visibility, and application-aware resource allocation remain higher-level strategies that could be implemented on top of the toolkit.

6.2 Evaluation Limitations

At the same time, the workshop has several limitations. It was exploratory and did not include a baseline condition. We therefore cannot quantify how much faster or easier LinkSpace is compared to implementing similar functionality from scratch or with existing portal assets. The participants were computer science students enrolled in a master's program, which means that the results may not generalize to non-programmers or professional VR developers. Questionnaires were completed individually, but participants prototyped in six teams of three; responses within teams may therefore partly reflect shared implementation experiences. The groups also used different development setups: most groups used Quest 3 headsets, while two groups used a simulator on macOS. This may have influenced the implementation experience.

The results should therefore be read as initial evidence from an exploratory workshop rather than a controlled validation. They show that the toolkit enabled a range of linked-space prototypes and was perceived as useful and usable in a short workshop.

6.3 Technical Limitations and Future Work

The toolkit itself also has technical limitations. LinkSpace is implemented as a Unity toolkit and is therefore tied to one specific game engine, its XR stack, and its rendering pipeline. It also does not currently provide explicit multi-scene support. Instead, applications that use multiple VEs represent them as separate levels with their own skyboxes while keeping them technically within a single Unity scene. The performance characterization represents a single scene and device configuration; rendering costs additionally depend on scene complexity, view size, and render resolution.

Future work should compare LinkSpace against alternative implementation approaches, include longer development sessions, and study how developers use the toolkit in more realistic project settings.

7 Conclusion

This paper introduced *linked spaces* as a term for techniques that connect virtual environments, locations, or perspectives in VR. Under this term, we discussed classical portals as well as related forms such as spatial previews, lenses, HUD elements, and other portal-like links. Based on a design space derived from recurring requirements across these systems, we presented LinkSpace, a Unity toolkit for prototyping configurable linked spaces in VR. The toolkit supports runtime creation, linking, movement, and closing of linked spaces, as well as stereoscopic rendering, traversal, ray-based teleportation, remote selection, grabbing, and object transfer.

The workshop showed that LinkSpace enables developers to create varied linked-space prototypes within a short time, including body-attached portals, inventory systems, remote views, minimaps, portal guns, and bidirectional object transfer. Participants rated the toolkit as usable (SUS: 75.1, $SD = 8.1$) and described it as useful, flexible, and easy to apply for rapid prototyping, while also pointing to needs for clearer documentation, debugging support, setup guidance, and perspective-dependent interaction.

Overall, LinkSpace shows how reusable and configurable VR components can make linked-space techniques easier to prototype and study. We hope this contribution sparks further creative research and supports new systems for working across linked spaces.

References

[1] 2018. Pocket Portal VR | Particles/Effects | Unity Asset Store. https://assetstore.unity.com/packages/tools/particles-effects/pocket-portal-vr-85657

[2] 2026. Portals for VR | VFX | Unity Asset Store. https://assetstore.unity.com/packages/vfx/portals-for-vr-228871

[3] 2026. WearableComputerLab/VRPortalToolkit. original-date: 2022-08-12T01:59:18Z. https://github.com/WearableComputerLab/VRPortalToolkit

[4] Daniel Ablett, Andrew Cunningham, Gun A. Lee, and Bruce H. Thomas. 2022. Portal Rendering and Creation Interactions in Virtual Reality. In *2022 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. 160–168. ISSN: 1554-7868. doi:10.1109/ISMAR55827.2022.00030

[5] Daniel Ablett, Andrew Cunningham, Gun A. Lee, and Bruce H. Thomas. 2023. Point & Portal: A New Action at a Distance Technique For Virtual Reality. In *2023 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. 119–128. ISSN: 2473-0726. doi:10.1109/ISMAR59233.2023.00026

[6] Daniel Ablett, Andrew Cunningham, Gun A. Lee, and Bruce H. Thomas. 2024. Adaptive Portals: Enhancing Virtual Reality Interaction Spaces With Real-Time Self-Adjusting Portals. In *2024 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. 1010–1018. ISSN: 2473-0726. doi:10.1109/ISMAR62088.2024.00117

[7] Ferran Argelaguet and Carlos Andujar. 2013. A survey of 3D object selection techniques for virtual environments. *Computers & Graphics* 37, 3 (May 2013), 121–136. doi:10.1016/j.cag.2012.12.003

[8] Aaron Bangor, Philip Kortum, and James Miller. 2009. Determining what individual SUS scores mean: adding an adjective rating scale. *J. Usability Studies* 4, 3 (May 2009), 114–123. https://dl.acm.org/doi/10.5555/2835587.2835589

[9] Doug A. Bowman, Ernst Kruijff, Joseph J. LaViola, and Ivan Poupyrev. 2001. An Introduction to 3-D User Interface Design. *Presence: Teleoperators and Virtual Environments* 10, 1 (Feb. 2001), 96–108. doi:10.1162/105474601750182342

[10] John Brooke. 1996. SUS: a quick and dirty usability scale. *Usability evaluation in industry* 189, 3 (1996), 189–194.

[11] Jack Brookes, Matthew Warburton, Mshari Alghadier, Mark Mon-Williams, and Faisal Mushtaq. 2020. Studying human behavior with virtual reality: The Unity Experiment Framework. *Behavior Research Methods* 52, 2 (April 2020), 455–463. doi:10.3758/s13428-019-01242-0

[12] Wolfgang Büschel, Anke Lehmann, and Raimund Dachselt. 2021. MIRIA: A Mixed Reality Toolkit for the In-Situ Visualization and Analysis of Spatio-Temporal

Interaction Data. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems (CHI '21)*. Association for Computing Machinery, New York, NY, USA, 1–15. doi:10.1145/3411764.3445651

[13] Robbe Cools, Augusto Esteves, and Adalberto L. Simeone. 2022. Blending Spaces: Cross-Reality Interaction Techniques for Object Transitions Between Distinct Virtual and Augmented Realities. In *2022 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. 528–537. ISSN: 1554-7868. doi:10.1109/ISMAR55827.2022.00069

[14] Satabdi Das, Arshad Nasser, and Khalad Hasan. 2024. Exploring Finger-Worn Solutions for Transitioning between the Reality-Virtuality Continuum. In *2024 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. 1167–1176. ISSN: 2473-0726. doi:10.1109/ISMAR62088.2024.00133

[15] Martin Feick, Cihan Biyikli, Kiran Gani, Anton Wittig, Anthony Tang, and Antonio Krüger. 2023. VoxelHap: A Toolkit for Constructing Proxies Providing Tactile and Kinesthetic Haptic Feedback in Virtual Reality. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology (UIST '23)*. Association for Computing Machinery, New York, NY, USA, 1–13. doi:10.1145/3586183.3606722

[16] Nico Feld, Pauline Bimberg, Benjamin Weyers, and Daniel Zielasko. 2024. Simple and Efficient? Evaluation of Transitions for Task-Driven Cross-Reality Experiences. *IEEE Transactions on Visualization and Computer Graphics* 30, 12 (Dec. 2024), 7601–7618. doi:10.1109/TVCG.2024.3356949

[17] Sebastian Freitag, Dominik Rausch, and Torsten Kuhlen. 2014. Reorientation in virtual environments using interactive portals. In *2014 IEEE Symposium on 3D User Interfaces (3DUI)*. 119–122. doi:10.1109/3DUI.2014.6798852

[18] Sebastian J Friston, Ben J Congdon, David Swapp, Lisa Izzouzi, Klara Brandstätter, Daniel Archer, Otto Olkkonen, Felix Johannes Thiel, and Anthony Steed. 2021. Ubiq: A System to Build Flexible Social Virtual Reality Experiences. In *Proceedings of the 27th ACM Symposium on Virtual Reality Software and Technology (VRST '21)*. Association for Computing Machinery, New York, NY, USA, 1–11. doi:10.1145/3489849.3489871

[19] Matt Gottsacker, Hiroshi Furuya, Laura Battistel, Carlos Pinto Jimenez, Nicholas LaMontagna, Gerd Bruder, and Gregory F. Welch. 2024. Exploring Spatial Cognitive Residue and Methods to Clear Users' Minds When Transitioning Between Virtual Environments. In *2024 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. 1000–1009. ISSN: 2473-0726. doi:10.1109/ISMAR62088.2024.00116

[20] Matt Gottsacker, Yahya Hmaiti, Mykola Maslych, Hiroshi Furuya, Jasmine Joyce DeGuzman, Gerd Bruder, Gregory F. Welch, and Joseph J. LaViola. 2026. From One World to Another: Interfaces for Efficiently Transitioning Between Virtual Environments. In *Proceedings of the 2026 CHI Conference on Human Factors in Computing Systems (CHI '26)*. Association for Computing Machinery, New York, NY, USA, 1–17. doi:10.1145/3772318.3791912

[21] Saul Greenberg and Chester Fitchett. 2001. Phidgets: easy development of physical interfaces through physical widgets. In *Proceedings of the 14th annual ACM symposium on User interface software and technology*. ACM, Orlando Florida, 209–218. doi:10.1145/502348.502388

[22] Malte Husung and Eike Langbehn. 2019. Of Portals and Orbs: An Evaluation of Scene Transition Techniques for Virtual Reality. In *Proceedings of Mensch und Computer 2019 (MuC '19)*. Association for Computing Machinery, New York, NY, USA, 245–254. doi:10.1145/3340764.3340779

[23] André Kunert, Alexander Kulik, Stephan Beck, and Bernd Froehlich. 2014. Photoportals: shared references in space and time. In *Proceedings of the 17th ACM conference on Computer supported cooperative work & social computing (CSCW '14)*. Association for Computing Machinery, New York, NY, USA, 1388–1399. doi:10.1145/2531602.2531727

[24] Maarten H. Lamers and Maik Lanen. 2021. Changing Between Virtual Reality and Real-World Adversely Affects Memory Recall Accuracy. *Frontiers in Virtual Reality* 2 (March 2021). doi:10.3389/frvir.2021.602087

[25] David Ledo, Steven Houben, Jo Vermeulen, Nicolai Marquardt, Lora Oehlberg, and Saul Greenberg. 2018. Evaluation Strategies for HCI Toolkit Research. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. ACM, Montreal QC Canada, 1–17. doi:10.1145/3173574.3173610

[26] Jaewook Lee, Raahul Natarrajan, Sebastian S. Rodriguez, Payod Panda, and Eyal Ofek. 2022. RemoteLab: A VR Remote Study Toolkit. In *Proceedings of the 35th Annual ACM Symposium on User Interface Software and Technology (UIST '22)*. Association for Computing Machinery, New York, NY, USA, 1–9. doi:10.1145/3526113.3545679

[27] Kevin Linne, Sven Thomas, Jennifer Roth, and Martin Weigel. 2025. SliVR: A 360° VR-Hub for Fast Selections in Multiple Virtual Environments. In *2025 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. 551–559. ISSN: 2473-0726. doi:10.1109/ISMAR67309.2025.00065

[28] Kevin Linne, Sven Thomas, and Martin Weigel. 2024. Off-The-Shelf: Exploring 3D Arrangements of See-Through Masks to Switch between Virtual Environments. In *Proceedings of the 30th ACM Symposium on Virtual Reality Software and Technology (VRST '24)*. Association for Computing Machinery, New York, NY, USA, 1–2. doi:10.1145/3641825.3689501

[29] Jiazhou Liu, Arnaud Prouzeau, Barrett Ens, and Tim Dwyer. 2022. Effects of Display Layout on Spatial Memory for Immersive Environments. *Proc. ACM Hum.-Comput. Interact.* 6, ISS (Nov. 2022), 576:468–576:488. doi:10.1145/3567729

[30] Xiaolong Liu, Lili Wang, Yi Liu, and Jian Wu. 2024. Automatic portals layout for VR navigation. *Virtual Reality* 28, 1 (Jan. 2024), 9. doi:10.1007/s10055-023-00897-7

[31] Nicolai Marquardt, Robert Diaz-Marino, Sebastian Boring, and Saul Greenberg. 2011. The proximity toolkit: prototyping proxemic interactions in ubiquitous computing ecologies. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*. ACM, Santa Barbara California USA, 315–326. doi:10.1145/2047196.2047238

[32] Liang Men, Nick Bryan-Kinns, Amelia Shivani Hassard, and Zixiang Ma. 2017. The impact of transitions on user experience in virtual reality. In *2017 IEEE Virtual Reality (VR)*. 285–286. ISSN: 2375-5334. doi:10.1109/VR.2017.7892288

[33] Reiji Miura, Shunichi Kasahara, Michiteru Kitazaki, Adrien Verhulst, Masahiko Inami, and Maki Sugimoto. 2021. MultiSoma: Distributed Embodiment with Synchronized Behavior and Perception. In *Proceedings of the Augmented Humans International Conference 2021 (AHs '21)*. Association for Computing Machinery, New York, NY, USA, 1–9. doi:10.1145/3458709.3458878

[34] Jung Who Nam, Krista McCullough, Joshua Tveite, Maria Molina Espinosa, Charles H. Perry, Barry T. Wilson, and Daniel F. Keefe. 2019. Worlds-in-Wedges: Combining Worlds-in-Miniature and Portals to Support Comparative Immersive Visualization of Forestry Data. In *2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*. 747–755. doi:10.1109/VR.2019.8797871

[35] Michael Nebeling, Maximilian Speicher, Xizi Wang, Shwetha Rajaram, Brian D. Hall, Zijian Xie, Alexander R. E. Raistrick, Michelle Aebersold, Edward G. Happ, Jiayin Wang, Yanan Sun, Lotus Zhang, Leah E. Ramsier, and Rhea Kulkarni. 2020. MRAT: The Mixed Reality Analytics Toolkit. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems (CHI '20)*. Association for Computing Machinery, New York, NY, USA, 1–12. doi:10.1145/3313831.3376330

[36] Sebastian Oberdörfer, Martin Fischbach, and Marc Erich Latoschik. 2018. Effects of VE Transition Techniques on Presence, Illusion of Virtual Body Ownership, Efficiency, and Naturalness. In *Proceedings of the 2018 ACM Symposium on Spatial User Interaction (SUI '18)*. Association for Computing Machinery, New York, NY, USA, 89–99. doi:10.1145/3267782.3267787

[37] Leonardo Pavanatto, Feiyu Lu, Chris North, and Doug A. Bowman. 2025. Multiple Monitors or Single Canvas? Evaluating Window Management and Layout Strategies on Virtual Displays. *IEEE Transactions on Visualization and Computer Graphics* 31, 3 (March 2025), 1713–1730. doi:10.1109/TVCG.2024.3368930

[38] Leonardo Pavanatto, Chris North, Doug A. Bowman, Carmen Badea, and Richard Stoakley. 2021. Do we still need physical monitors? An evaluation of the usability of AR virtual monitors for productivity work. In *2021 IEEE Virtual Reality and 3D User Interfaces (VR)*. 759–767. ISSN: 2642-5254. doi:10.1109/VR50410.2021.00103

[39] Stéven Picard, Ningyuan Sun, and Jean Botev. 2024. XR MUSE: An Open-Source Unity Framework for Extended Reality-Based Networked Multi-User Studies. *Virtual Worlds* 3, 4 (Dec. 2024), 404–417. doi:10.3390/virtualworlds3040022

[40] Henning Pohl, Klemen Lilija, Jess McIntosh, and Kasper Hornbæk. 2021. Poros: Configurable Proxies for Distant Interactions in VR. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems (CHI '21)*. Association for Computing Machinery, New York, NY, USA, 1–12. doi:10.1145/3411764.3445685

[41] Fabian Pointecker, Judith Friedl, Daniel Schwajda, Hans-Christian Jetter, and Christoph Anthes. 2022. Bridging the Gap Across Realities: Visual Transitions Between Virtual and Augmented Reality. In *2022 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. 827–836. ISSN: 1554-7868. doi:10.1109/ISMAR55827.2022.00101

[42] Fabian Pointecker, Judith Friedl-Knirsch, Hans-Christian Jetter, and Christoph Anthes. 2024. From Real to Virtual: Exploring Replica-Enhanced Environment Transitions along the Reality-Virtuality Continuum. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems (CHI '24)*. Association for Computing Machinery, New York, NY, USA, 1–13. doi:10.1145/3613904.3642844

[43] Kadek Ananta Satriadi, Barrett Ens, Maxime Cordeil, Tobias Czauderna, and Bernhard Jenny. 2020. Maps Around Me: 3D Multiview Layouts in Immersive Spaces. *Proc. ACM Hum.-Comput. Interact.* 4, ISS (Nov. 2020), 201:1–201:20. doi:10.1145/3427329

[44] Jonas Schjerlund, Kasper Hornbæk, and Joanna Bergström. 2022. OVRlap: Perceiving Multiple Locations Simultaneously to Improve Interaction in VR. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems (CHI '22)*. Association for Computing Machinery, New York, NY, USA, 1–13. doi:10.1145/3491102.3501873

[45] Richard Stoakley, Matthew J. Conway, and Randy Pausch. 1995. Virtual reality on a WIM: interactive worlds in miniature. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '95)*. ACM Press/Addison-Wesley Publishing Co., USA, 265–272. doi:10.1145/223904.223938

[46] Stanislav L. Stoev and Dieter Schmalstieg. 2002. Application and taxonomy of through-the-lens techniques. In *Proceedings of the ACM symposium on Virtual reality software and technology (VRST '02)*. Association for Computing Machinery, New York, NY, USA, 57–64. doi:10.1145/585740.585751

[47] Tuukka M. Takala. 2014. RUIS: a toolkit for developing virtual reality applications with spatial interaction. In *Proceedings of the 2nd ACM symposium on Spatial user*

interaction (SUI '14). Association for Computing Machinery, New York, NY, USA, 94–103. doi:10.1145/2659766.2659774

[48] Chiu-Hsuan Wang, Bing-Yu Chen, and Liwei Chan. 2022. RealityLens: A User Interface for Blending Customized Physical World View into Virtual Reality. In *Proceedings of the 35th Annual ACM Symposium on User Interface Software and Technology (UIST '22)*. Association for Computing Machinery, New York, NY, USA, 1–11. doi:10.1145/3526113.3545686

[49] 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 (UIST '20)*. Association for Computing Machinery, New York, NY, USA, 805–817. doi:10.1145/3379337.3415868

[50] Matthias Wölfel, Daniel Hepperle, Christian Felix Purps, Jonas Deuchler, and Wladimir Hettmann. 2021. Entering a new Dimension in Virtual Reality Research: An Overview of Existing Toolkits, their Features and Challenges. In *2021 International Conference on Cyberworlds (CW)*. 180–187. ISSN: 2642-3596. doi:10.1109/CW52790.2021.00038

[51] André Zenner, Hannah Maria Kriegler, and Antonio Krüger. 2021. HaRT - The Virtual Reality Hand Redirection Toolkit. In *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems (CHI EA '21)*. Association for Computing Machinery, New York, NY, USA, 1–7. doi:10.1145/3411763.3451814

[52] Yuhang Zhao, Edward Cutrell, Christian Holz, Meredith Ringel Morris, Eyal Ofek, and Andrew D. Wilson. 2019. SeeingVR: A Set of Tools to Make Virtual Reality More Accessible to People with Low Vision. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI '19)*. Association for Computing Machinery, New York, NY, USA, 1–14. doi:10.1145/3290605.3300341