Spatialstrates: Cross-Reality Collaboration through Spatial Hypermedia

Marcel Borowski
marcel.borowski@cs.au.dk
Aarhus University
Aarhus, Denmark

Jens Emil Grønbæk
jensemil@cs.au.dk
Aarhus University
Aarhus, Denmark

Peter W. S. Butcher
p.butcher@bangor.ac.uk
Bangor University
Bangor, United Kingdom

Panagiotis D. Ritsos
p.ritsos@bangor.ac.uk
Bangor University
Bangor, United Kingdom

Clemens N. Klokmose
clemens@cs.au.dk
Aarhus University
Aarhus, Denmark

Niklas Elmqvist
elm@cs.au.dk
Aarhus University
Aarhus, Denmark

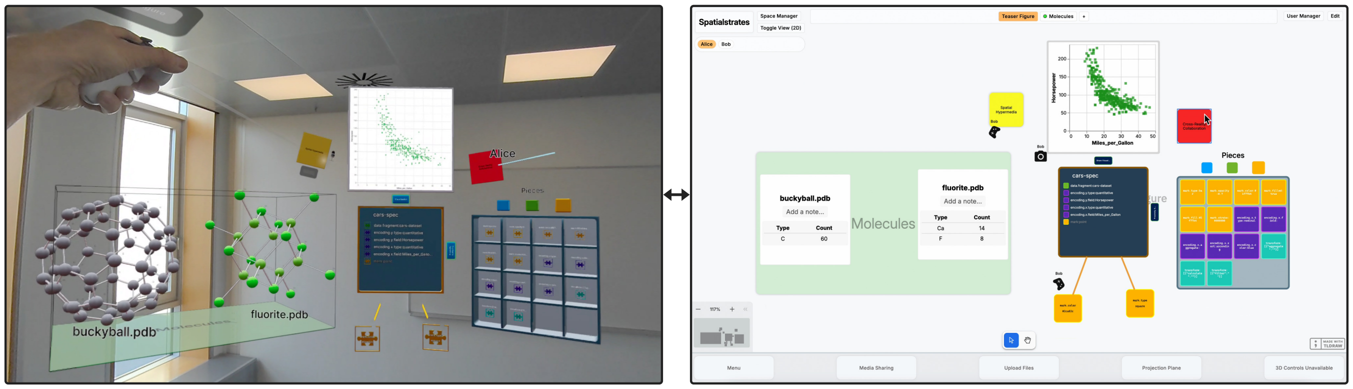

Figure 1: Cross-reality collaboration within Spatialstrates. Both sides show the same space with molecule structures, sticky notes, and DashSpace immersive analytics elements. *Left:* The 3D space in the immersive AR view from the perspective of Bob. A blue cursor avatar line on the red sticky note indicates the cursor position of the Alice in 2D. *Right:* The same space in 2D on a desktop computer from the perspective of Alice. Elements' positions are projected onto a 2D plane and have a distinct representation. Black avatars indicate the projected position of Bob's headset and two controllers.

ABSTRACT

Consumer-level XR hardware now enables immersive spatial computing, yet most knowledge work remains confined to traditional 2D desktop environments. These worlds exist in isolation: writing emails or editing presentations favors desktop interfaces, while viewing 3D simulations or architectural models benefits from immersive environments. We address this fragmentation by combining spatial hypermedia, shareable dynamic media, and cross-reality computing to provide (1) composability of heterogeneous content and of nested information spaces through spatial transclusion, (2) pervasive cooperation across heterogeneous devices and platforms, and (3) congruent spatial representations despite underlying environmental differences. Our implementation, the Spatialstrates platform, embodies these principles using standard web technologies to bridge 2D desktop and 3D immersive environments. Through four scenarios—collaborative brainstorming, architectural design, molecular science visualization, and immersive analytics—we demonstrate how Spatialstrates enables collaboration between desktop 2D and immersive 3D contexts, allowing users to select the most appropriate interface for each task while maintaining collaborative capabilities.

CCS CONCEPTS

• **Human-centered computing → Mixed / augmented reality**; *Hypertext / hypermedia*; *Collaborative interaction*; Web-based interaction.

KEYWORDS

Cross-reality, hypermedia, Augmented Reality, Extended Reality, collaboration, 2D/3D integration.

ACM Reference Format:
Marcel Borowski, Jens Emil Grønbæk, Peter W. S. Butcher, Panagiotis D. Ritsos, Clemens N. Klokmose, and Niklas Elmqvist. 2025. Spatialstrates: Cross-Reality Collaboration through Spatial Hypermedia. In *The 38th Annual ACM Symposium on User Interface Software and Technology (UIST '25), September 28-October 1, 2025, Busan, Republic of Korea.* ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3746059.3747708

1 INTRODUCTION

It is the best of times: consumer-level XR (Extended Reality) hardware is now within easy reach of even casual users, and spatial computing's potential for supporting both games and entertainment as well as office productivity and data analytics is unparalleled in history. However, it is also the worst of times: the 2D desktop environments where we conduct most of our knowledge work are hopelessly fragmented from the 3D game environments—such as Unity and Unreal—that provide the best up-to-date support for such novel XR hardware [8]. There are clear situations in everyday computing that favor one platform or the other: writing an email or editing a presentation is still better suited for the standard desktop, whereas viewing the results from a turbulence simulation or understanding a building's relationship to the sun is easier in immersive 3D. Common between them is also a growing need for synchronous collaboration, fundamental to all its applications. To recall these two platforms to life, we must not only reconcile both platforms into a common medium that can compound their strengths and complement their weaknesses, but also seamlessly connect desktop 2D with immersive 3D in asymmetric and real-time collaboration.

In this paper, we address this challenge by combining three themes: *spatial hypermedia* [17, 36], where information objects and spaces can be **composed** in spatial interlinked arrangements; *shareable dynamic media* [28], where **cooperation** support is pervasive; and *cross-reality computing* [1, 13], where work is spatially **congruent** across conventional and immersive spaces. These yield three core properties of next-generation spatial computing systems:

- **Composability** of heterogeneous interactive content in spaces that can be interlinked and transcluded;
- **Cooperability** of users across heterogeneous devices and platforms, across 2D and 3D interfaces, and across conventional and immersive interfaces; and
- **Congruency** of spatial representations across 2D and 3D environments, including content manipulation, gestures, avatar positioning, and WYSIWIS [51] transclusion.

To demonstrate and validate these properties, we present Spatialstrates, an XR platform built using standard web technologies to address the challenges of cross-reality computing through collaboration between 2D desktop and immersive 3D. It serves both as a rapid prototyping tool and a generative environment for exploring spatial hypermedia. We evaluate Spatialstrates as a modern spatial hypermedia platform through demonstration [31] and discuss it in terms of Olsen's system evaluation criteria [40]. We showcase its capabilities through four concrete scenarios: (1) an architecture and lighting application showcasing immersive 3D collaboration; (2) a molecular biology application supporting collaborative viewing of Protein Databank (PDB) 3D models; (3) an immersive analytics scenario for multidimensional data using the DashSpace system [4] in both 2D and 3D; and (4) a 2D desktop-based brainstorming environment involving UX and usability evaluation data.

The contributions of this paper are as follows:

- A revitalization of *spatial hypermedia* principles as means for bridging immersive 3D and conventional 2D interfaces;
- The Spatialstrates platform for developing collaborative spatial hypermedia in XR; and
- Four practical usage scenarios of spatial hypermedia, including for Miro-style XR brainstorming, architectural collaboration, molecular science, and immersive analytics.

2 RELATED WORK

We review prior work on cross-reality computing, aiming to address challenges of spatial composability ( ) of content, cooperability ( ) in XR, and communication congruency ( ).

2.1 XR Needs a Platform for Flexible Spatial *Composability* ( ) of Content

With prior cross-reality work focusing on individual techniques and transitions, little attention has been given to underlying compositional mechanisms for creating and sharing content spatially across 2D and 3D workspaces. We propose to rethink content composition and organization by drawing upon key principles of hypermedia: *cross-platform support*, *spatiality*, and *transclusion*.

Cross-Platform Support. The advent of smartphones, broadband connectivity, and advances in display technology have brought XR applications out of research labs and into consumer hands [2]. This shift has been supported by efforts to bring standards-based XR into common Web browsers [9, 33, 34]. For XR to become ubiquitous, it must work out-of-the-box. However, prior efforts often required installing dedicated browser prototypes. Apple's visionOS allows placement of 2D and 3D content in AR, however, this content is only accessible within the HMD and cannot be viewed on other devices or in 2D. The Web offers promise as a de facto platform-independent ecosystem for building and sharing persistent, collaborative XR experiences [44, 45]. With increasing support for WebXR across modern HMDs [8] through standards like the WebXR Device API [55] or A-Frame, the time is ripe for building a hypermedia foundation for cross-reality computing, bringing the cross-platform support of Web technologies to spatial computing.

Spatiality. A body of prior work has proposed spatial forms of hypermedia, e.g., to support latent content composition for information triage (Spatial Hypertext [36]), provide meaningful metaphors for organizing mixed physical and digital objects [17, 18], or to compose virtual 3D environments (HyperReal [46]).

The Topos system [18] introduced a foundational approach to integrating hypermedia with spatial information. By combining metaphorical (workspace-based organization) with literal (geospatial representations) spaces, Topos demonstrated how information could remain meaningful across different spatial contexts. However, Topos did not provide means for live synchronous collaboration between 2D and 3D workspaces, a capability that is essential for modern cross-reality computing environments.

Transclusion. Transclusion is a fundamental hypermedia principle that refers to including part or all of a document in another by reference [39]. On the Web, document-based transclusion can be achieved with the `iframe` element. Webstrates [28] has used transclusion as a mechanism for software composition, creating document-application relationships, and realizing non-web native hypermedia principles such as bidirectional links [7]. Transclusion can also be spatial. Topos [18] introduced proxy-based workspace composition, enabling the transclusion of spatial elements between interconnected workspaces. This allowed hypermedia elements to be embedded, referenced, and synchronized without duplication, ensuring consistency across spatial contexts.

Application to XR. Despite being highly relevant for cross-reality computing, the above hypermedia principles have yet to be applied to modern Web-based XR. Spatialstrates applies these concepts to modern cross-reality computing, supporting interoperability across 3D XR and 2D desktop interfaces.

2.2 XR Needs a Platform that Supports *Cooperability* (👥)

Cross-Reality Collaboration. Systems for cross-reality collaboration exemplify the need for true cooperability between 2D and 3D environments. Butscher et al. [10] showed that immersive environments enable fluid analysis during collaborative sensemaking of multidimensional data. Taking this further, Lee et al. [32] explored shared surfaces and spaces for co-located immersive collaboration, finding that spatial awareness mechanisms were key for coordinated work across shared virtual environments. Saffo et al. [47] established principles for effective coordination during VR and desktop collaboration to support mutual understanding across platforms. Fröhler et al. [13] formalize this emerging area as *cross-virtuality analytics* (XVA), defining it as a field that enables visual analytics through transitional and collaborative interfaces that seamlessly integrate different devices across the reality-virtuality continuum. ShareVR [20] enables shared co-located immersive experiences with interactions between HMD and non-HMD users. Loki [52] explores a variety of guidance tasks through bidirectional cross-reality communication cues.

Kumaravel and Hartmann's conceptual framework [53] articulates the experiences with the above systems as *cross-dimensional* collaboration. Although these prior systems provide extensive support for cross-dimensional collaboration, they all require installing dedicated native applications, which limits the possibilities for ad hoc sharing and cooperability.

Shareable Dynamic Media and Webstrates. The Webstrates [28] platform and its Codestrates authoring environment [5] are built to lower the threshold of ad hoc sharing and cooperability. They exemplify the paradigm of *shareable dynamic media*, which dissolves the traditional boundaries between applications and documents, developer and user, and where collaboration support is the norm rather than the exception. This approach is underpinned by three core principles: *shareability*—software is inherently collaborative—, *distributability*—software is accessible across heterogeneous devices and platforms—, and *malleability*—software is amendable, allowing users to tailor it to their needs. Webstrates demonstrates these principles through conventional web technology with software running in an unmodified Web browser. It enables asymmetric collaboration where users can collaborate in real-time on the same content with different tools. By removing the technical boundary between editing content and code, Webstrates enables software to be tailored—and even reprogrammed—while in use (as, e.g., demonstrated in Mirrorverse [19]). Most related to our paper is the work of Borowski et al. [4], who bring the above principles to XR with DashSpace, a web-based XR platform for ubiquitous analytics built on Webstrates using WebXR, supporting 3D collaboration for desktop interfaces, VR, and AR.

Creating a Generalized Platform. With Spatialstrates, we build on the work from Borowski et al. [4], but expand their approach into a more general-purpose Web-based XR platform supporting collaboration between 2D and 3D interfaces, while enabling the combination of multiple spaces through spatial transclusion.

2.3 XR Needs a Platform that Provides *Congruency* (⇄) Despite Heterogeneity

When collaborating in XR, users benefit from seamless interaction with congruent communication cues. The challenge is to overcome different types of heterogeneity–i.e. disparities between the users' respective workspaces. We distinguish between two types identified in prior work; physical space and device heterogeneity.

Heterogeneity Across Distributed Physical Spaces. When distributed users collaborate in XR, heterogeneity across disjoint physical spaces creates challenges for rendering congruent non-verbal communication cues, such as mutual gaze or consistent pointing gestures [58]. Such non-verbal communication support is often referred to as providing cues for workspace awareness [21, 22], with a consistent reference space [11] to enable effective deixis through pointing gestures [14]. Recent work proposes more flexible shared space models that accommodate disjoint collaborative environments (e.g. Partially Blended Realities [15] and Re-locations [12]), while maintaining congruent cues within the vicinity of the active workspace area. Most recently, systems have started to provide mechanisms for users to create more tailored spaces for congruent collaboration, e.g. MRTransformer [56], Blended Whiteboard [16], or SurfShare [24]. For an extensive review of how such XR techniques can establish spatial congruency, refer to [58].

Heterogeneity Across Multiple Devices. Another form of heterogeneity occurs when users collaborate across workspaces with different dimensionalities, such as 2D desktop devices and 3D immersive environments [1, 13]. Such cross-platform collaboration offers unique advantages, as devices can mutually scaffold each other's weaknesses [23]. Systems like ReLive [25] and Wang et al.'s particle visualization [57] demonstrate how 2D analytical (ex-situ) views complement 3D immersive (in-situ) perspectives. In VRxD, Saffo et al. [47] propose "the eyes and shoes" principle for facilitating coordination through increasing degrees of common ground across 2D and 3D viewing conditions. Reski et al. [43] evaluated synchronous collaboration between desktop and VR users with asymmetric visualization design, providing visual cues such as highlighting shared data and communicating VR user location to desktop collaborators. Their findings indicated high levels of group awareness despite platform differences. Similarly, Tong et al. [54] demonstrated how well-designed asymmetric visualization views improve task productivity and reduce mental demand across desktop and VR environments.

A Spatial Composition Model. While some research explores how collaborators may transition between environments [1, 26, 37, 49], the provision of consistent *real-time communication cues* across 2D and 3D remains underexplored. With Spatialstrates, we propose a spatial composition model that flexibly addresses incongruencies across virtual and physical information spaces in both 2D and 3D. Besides composition, we demonstrate initial avatar communication cues across 2D and 3D.

3 SPATIALSTRATES (🧩 👥 ⇄)

We present Spatialstrates, a proof-of-concept XR platform that implements the three properties needed for realizing effective spatial hypermedia; *composability*, *cooperability* and *congruency*.

In Spatialstrates, diverse content—including notes, images, 3D models as well as interactive data visualizations—is composed of spaces (🧩) that can be represented either in 2D as a conventional infinite canvas (similar to, e.g., Miro [38]) or as a 3D scene. Spaces are referenced through hyperlinks, and spaces can be transcluded into one-another allowing for content to be shared between spaces and for users to navigate between spaces. Through Spatialstrates, multiple users can cooperate in real-time (👥) across 2D and 3D as well as between immersive and desktop representations of space. Congruency between 2D and 3D (⇄) is realized through a computed projection plane mediating movement of objects, pointing, and placement of user avatars. Congruency between different physical spaces is supported through a WYSIWIS approach to spatial transclusion, where the bounds of a transcluded space is shared.

Spatialstrates enables a wide range of use cases for spatial applications in desktop 2D, immersive 3D, and across the two. It supports a variety of collaborative scenarios including synchronous, asynchronous, co-located, and remote collaboration on desktop computers or immersive HMDs. It provides a flexible platform that can be extended to accommodate a variety of use cases. Allowing users to choose the best device (HMD or desktop) for the task at hand, while preserving the ability to collaborate between both, providing an environment for exploring spatial hypermedia.

Spatialstrates' builds on the codebase of the point-design XR data visualization application DashSpace [4]. Yet, Spatialstrates is designed and architected as an extensible platform that streamlines development of XR applications. Additionally, Spatialstrates adds support for multiple interconnected spaces, spatial transclusion, and a new 2D canvas view which uses a 2D-3D projection plane for its mapping. It also adds initial cross-dimensional avatar support for positioning and pointing.

3.1 Workspaces, Spaces, and Elements

Spatialstrates is web-based and accessed through the browser. A Spatialstrates *workspace* is a webstrate [28] that contains the Spatialstrates implementation and is accessible through a unique URL.

Within a workspace, users can create *spaces*. A space is an infinite continuous space in which *elements* can be placed. Like switching tabs in a web browser, users can switch between spaces. Spaces—as well as the elements within them—are persisted and synchronized between all users accessing a workspace.

3.1.1 Spaces. A *space* is a continuous 3D space with a name and an optional color tag. A workspace can have an arbitrary number of spaces, which can be managed using the space manager, and a user can be at most in one space at a time and switch spaces using the space manager or space tabs (Figure 2). The active space is stored in the hash property of the URL.[1] A space also stores a *projection plane* to project it to a 2D space (see Section 3.3), and a *slice boundary* for use in containers (see Section 3.4).

Spaces can be viewed either in 3D or 2D and users can toggle between them using a button in the interface. The 3D space is navigated through "WASD" keys and the mouse on desktop computers, through handheld AR on phones and tablets, and through immersive AR/VR with motion controllers or hand tracking in HMDs. In AR, a space can be anchored in a room by moving a triangular calibration marker (same as in DashSpace [4]) to physical location.[2]

3.1.2 Elements. An *element* is an arbitrary (interactive) object in a space. They can range from simple sticky notes or images, to 3D models or light sources, to more complex interactive objects like the bookshelf from DashSpace [4]. What all elements have in common, is that they are located at most in a single space at any time, and that they have 3D position and rotation within that space. Each type of elements, e.g., a sticky note element, has a distinct representation for each the 3D and 2D views. These representations can be similar, or homogeneous, or heterogeneous and offer different visual representations and features depending on the view. It is also possible to have only a representation in one of the views.

Elements are usually created using the menu or can be created by other elements, for instance, the DashSpace bookshelf. Existing elements can be deleted using a menu or the trashcan element. Elements can also be cloned using the menu.

In our current implementation, we support the following element types: sticky notes, images, trashcan, screen-sharing streams, 3D models, flashlights, PDB molecules, and DashSpace's bookshelf, pieces, and groups.

3.2 Managers and Menus

The *space manager* is a dialog box, which lists all available spaces of a workspace. Users can use it to switch between spaces, create new or delete existing spaces, and to rename and color tag spaces. The *user manager*, similarly, is a dialog that allows users to create, select, or delete a user when using Spatialstrates.

An extensible *menu* is located at the bottom of the interface of desktop computers and mobile devices, and a *controller menu* is attached to the left controller or hand in immersive AR/VR. The menus can be extended using an API, e.g., to create new types of file uploaders or buttons to instantiate custom elements.

Spatialstrates also inherits general communication features from DashSpace, like audio and video streams as well as screen sharing.

3.3 Creating Congruent (⇄) Information Spaces Through Projection Planes

Element positions in Spatialstrates are stored as 3D coordinates. To position elements in a 2D environment, one could simply omit one of the coordinates, creating, for instance, a top-down view (omitting the y-axis) or a frontal view (omitting the x- or z-axis). However,

[1]Example: `www.server.com/workspaceId/#spaceId`
[2]This is required due to the current lack of persisted anchors or marker tracking in the WebXR Device API spec.

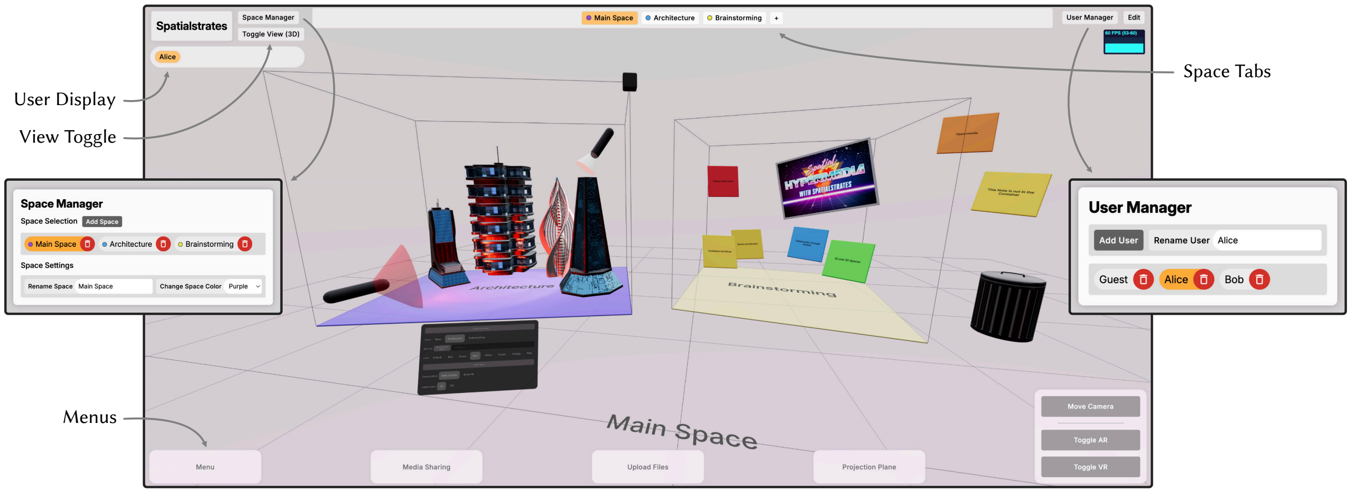

Figure 2: Spatialstrates desktop 3D interface. The view toggle switches between 2D and 3D. The space manager and user manager dialog boxes enable organizing spaces and users. The space tabs provide a quick way of switching and creating new spaces. The bottom menus can be dynamically extended and can be displayed on the controller menu in immersive AR/VR. The 3D space shows 3D models and flashlights in the left container; sticky notes and an image in the right container, and more sticky notes and a trashcan element outside the containers. The left container is selected and its menu is visible underneath it.

depending on the location of elements, such a naive approach disregards elements' positions and can cause significant levels of overlapping elements in 2D; e.g., when elements are arranged vertically in 3D, they are overlapped in a top-down projection.

Instead, we are employing a tailorable *projection plane* (Figure 3) to project from 3D to 2D. The projection plane can be either automatically optimized using PCA (principal component analysis) or manually adjusted to the users preference. Elements are then projected to 2D by computing the 2D coordinates of the intersection point of an vector, which is orthogonal to the projection plane and starts in the 3D position of the element, with the projection plane.

3.4 Enabling Composition ( ) Through Transclusion of Space with Containers

Inspired by the iframe elements on the web, *containers* in Spatialstrates enable the transclusion of *space slices*. A container is a special type of an element, i.e., it is located in and has a position in a space. A container is represented as a box in 3D and a rectangle in 2D. It behaves similar to an `iframe` embedding a website into another: a container embeds a slice of a (source) space into the current (target) space (Figure 5). However, unlike an `iframe` that embeds a whole document, containers only embeds a slice of a space. The position and size of a slice, the *slice boundary*, is stored and shared within the source space and shared across all containers that embed the same source space. This means that the size of all containers transcluding the same source space and its contained elements are the same.

The size of the boundary of a space a user is in can be modified by activating the boundary preview and resizing the box. Similarly, if a space is embedded in a container, the container can also be resized, which forwards the new size to the boundary of the source space and all other containers embedding the same source space.

On the one hand, this allows for the flexible composition of elements and spaces, e.g., by arranging them in hierarchical structures. On the other hand, containers are designed to act as shared spaces and to support ad-hoc collaboration. For instance, consider the case of Alice and Bob (Figure 6), each having their personal spaces ("Alice' Space" and "Bob's Space") with project related or personal documents in a Spatialstrates workspace. To collaborate on one project, they can create a third space ("Project Space"), which they both transclude into a container in their respective personal spaces. While staying in their personal spaces, they can shared elements in the container, creating a shared space between the two. As the size of the containers (and embedded slice) is always the same, it creates a shared WYSIWIS [51] (What You See Is What I See) reference.

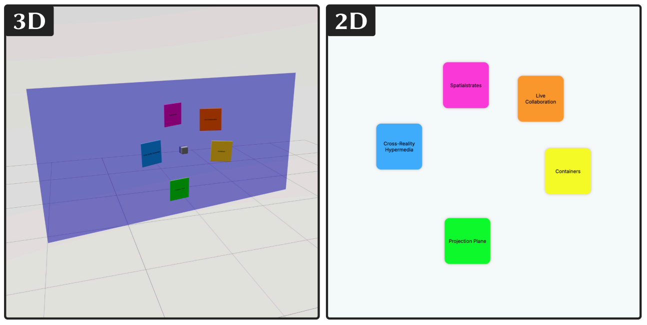

Figure 3: The Projection Plane. The projection plane is hidden by default but can be previewed in the 3D view using the menu. The box in the middle of the plane in 3D allows to drag and move the projection plane manually.

3.5 Facilitating Distributability Through WebXR and Shareability with Cross-Dimensional Avatars ( )

By building on the Webstrates platform and the web, workspaces in Spatialstrates can be shared using a URL, and by appending the hash to the URL even specific spaces can be shared via a URL. Spatialstrates, further, relies on the open WebXR standard (for more details see Section 5), which enables cross-platform support across desktop computers, mobile devices, and immersive HMDs without the need to install an additional application.

Spatialstrates facilitates basic real-time collaboration awareness using a variety of *avatars*—including cross-dimensional avatars from 3D to 2D and vice versa (Figure 1). From 3D to 3D, camera, controller, and hand avatars indicate the location and orientation of other users—these are also projected to 2D in the case of 3D to 2D. Given the tailorable projection plane, we decided against using field-of-view avatars in 2D, as their orientation could be confusing when perpendicular to the plane. From 2D to 2D, cursor avatars indicate the position of mouse cursors of other users. From 2D to 3D, the same cursor avatars are indicated as a line orthogonal to the projection plane, as there is no single cursor position available in this direction. Similar to DashSpace, avatars can also add a video feed when a desktop computer or mobile device is used, and provide audio communication through WebRTC.

Avatars of other users are generally only visible if they are in the same space as oneself. An exception is being in 3D space in close proximity to a container. Containers can be toggled to be collaborative, which forwards avatars to other spaces, in which a container embeds the same source space. Continuing the above example of Alice and Bob, this would enable them to see each others avatar and use ad-hoc audio communication when in proximity of the container with the shared project space. Similar to Re-locations [12] and MRTransformer [56], this enables users to create distinct zones for collaboration in their spaces, and place them according to their own physical space—overcoming heterogeneous physical spaces [58].

4 USAGE SCENARIOS

We demonstrate Spatialstrates' capabilities for bridging immersive 3D and desktop 2D environments through four usage scenarios (Table 1). Each scenario addresses some key challenges of creating a cross-reality platform for immersive 3D and desktop 2D: (1) Elevating legacy 2D applications to cross-reality and supporting ad-hoc sharing and persistence; (2) allowing for flexible ad-hoc collaboration in heterogeneous physical spaces; (3) using heterogeneous element representations in 3D and 2D to use the strengths of each information space; and (4) enabling unified interaction across immersive 3D and desktop 2D, and enabling collaboration across them.

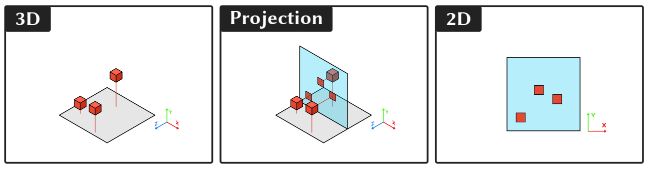

Figure 4: Projection from 3D to 2D. Elements in 3D space are projected into the projection plane using a vector that is orthogonal to the projection plane.

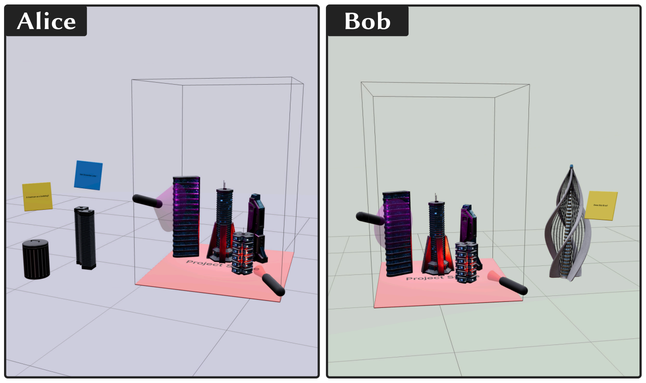

Figure 5: Containers transclude slices of another space. They are represented as wire frame boxes in 3D space.

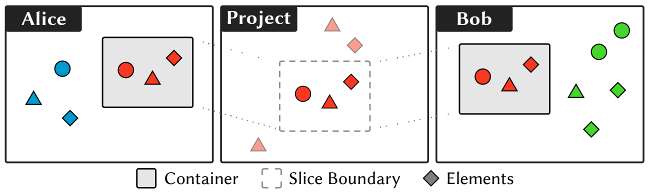

Figure 6: Transclusion of space. On the left, Alice' space with her personal elements (blue). On the right, Bob's space with his personal elements (green). Each of them has a container that transcludes the project space. In the middle, the project space with its elements (red) shows the boundary of the slice of space that is transcluded—it has the same size as the containers that transclude it. By default, elements outside the boundary are not transcluded (faded red elements).

4.1 Elevating 2D Sticky Note Brainstorming to Immersive 3D

Conventional desktop or web apps are usually distinct form immersive counterparts: A Miro board can only be viewed as a 2D website in immersive HMDs. Similarly, immersive apps are rarely accessible from a desktop computer. We demonstrate how Spatialstrates enables to combine both within the same platform, easily transitioning between the two.

4.1.1 Individual Brainstorming with Sticky Notes in 2D. Alex and James work at a healthcare company. They analyze patient feedback from their company's existing systems. They meet in Alex' office to analyze the feedback and brainstorm ideas using a Spatialstrates 2D canvas. Alex creates a new workspace and sends the link to James, who joins from his own laptop. Within the canvas, they can create sticky notes and add images. They start creating clusters of notes, using colors to categorize them (Figure 7A), and then both add some notes on possible solutions.

Scenario	Space		Collaboration Dimension			Collaboration Location		Transitions	
	3D	2D	3D ↔ 3D	3D ↔ 2D	2D ↔ 2D	Co-located	Remote	3D → 2D	2D → 3D
(1) Brainstorming	✓	✓	✓	–	✓	✓	–	–	✓
(2) Architecture Design	✓	–	✓	–	–	–	✓	–	–
(3) Molecular Science	✓	✓	✓	✓	–	✓	–	✓	✓
(4) Immersive Analytics	✓	✓	✓	✓	–	✓	–	✓	✓

Table 1: Scenario overview. We validate Spatialstrates using four disparate scenarios that illustrate different aspects of the tool, including visual dimensionality, collaboration, location, and transitions.

4.1.2 *Joint Discussion in Immersive Space.*

After adding some notes, they decide to discuss their results around the physical whiteboard. They open the space on their AR headsets and start an immersive session, moving the scene to align with the physical whiteboard. In AR, their notes and images are arranged in the same way they left them on their laptops, allowing them to seamlessly continue their brainstorming session. Using AR, they can present their notes to each other and decide on a plan to address the feedback (Figure 7B).

4.2 Ad-hoc Remote Collaboration Around Shared Containers in Architecture Design

Collaborative XR applications, e.g., created in Unity, commonly require being installed on devices or have to be compiled to work across multiple platforms. We demonstrate how Spatialstrates facilitates ad-hoc collaboration through shared containers, and how 3D scenes with 3D models can be shared with others.

4.2.1 *Preparing an Architecture Composition in Immersive 3D.*

Mike, an architecture model designer, creates a new space and imports multiple architecture models. He opens the 3D scene in a browser and adds the architectural elements. To arrange them effectively, he switches to immersive AR, putting on his headset. This allows him to visualize the composition in the room and make adjustments. To test how the structures react to different lighting conditions, Mike adds flashlight elements that cast directional lights on the models.

4.2.2 *Remote Collaboration Through Shared Containers.*

The next day, he prepares to present his composition to his supervisor, Samantha. To facilitate remote collaboration, Mike creates a shared container that both he and Samantha can access in their respective spaces. He moves the container around the architectural models and lights, ensuring that Samantha can also see them. Once either Mike or Samantha approach the shared volume, their avatars appear in the other user's scene and audio communication is shared (Figure 7C–D). Now, they can collaborate remotely around the shared container, arranging models and lights together. For instance, Samantha removes one of Mike's models and adds a model from her own space to the shared container, believing it complements the architectural composition well.

4.3 Heterogeneous Element Representations in Molecular Structure Science

Immersive 3D and desktop 2D environments each have unique strengths and weaknesses. For instance, typing text on a computer with a physical keyboard is straightforward, but using a virtual floating keyboard in immersive space can be challenging. On the other hand, while viewing 3D models on a 2D computer screen is feasible, exploring them in immersive and stereoscopic 3D offers an enhanced level of experience. We demonstrate how elements in Spatialstrates can be represented in multiple ways across 3D and 2D, making use of the strengths of each platform.

4.3.1 *Import and Annotation of Molecule Structures in Desktop 2D.*

Steve, a molecular scientist specializing in molecule structures, begins his work by creating a new workspace and uploading several structures. The structures appear in the 2D canvas, where Steve can access a summary of atoms used in the molecule and add a note. This allows him to add questions he later wants to check on the 3D model.

4.3.2 *Exploring and Clustering Molecules in Immersive 3D.*

After importing all structures, Steve wants to explore the variations in 3D. He uses his AR headset to open the same space in 3D and arranges the molecules in his office. To organize his explorations, Steve strategically places different regions in his office to cluster his findings. His notes from earlier are displayed under molecules.

4.3.3 *Collaboration Across Immersive 3D and Desktop 2D.*

Steve's colleague, Tom, joins him in his office to work on the molecules, bringing his own headset. Steve shares the URL to the workplace with Tom and, after he joins, demonstrates his exploration in immersive AR to him. During their discussion, Tom suggests exploring a variation of one of the structures, prompting Steve to switch back to his computer. Tom updates the projection plane to create a top down view for Steve. Steve creates a new molecule, uploads it to the workspace, and adds it to the space (Figure 8A). Tom can see the new molecule show up in her immersive view (Figure 8B) and he sees how Steve moves it closer to the other variant they wanted to compare it to. Steve resumes wearing his headset and compares the molecule with Tom's variant, facilitating a collaborative analysis.

4.4 Interaction Across Immersive 3D and Desktop 2D in Immersive Analytics

Most software exists either as regular desktop applications with a 2D interface or as immersive apps, that can be used with a HMD. As shown in the previous examples, Spatialstrates bridges this gap. However, another challenge is how to interact with applications that can be used across 3D and 2D. In this final scenario, we demonstrate how, using Spatialstrates, users can author data visualizations both

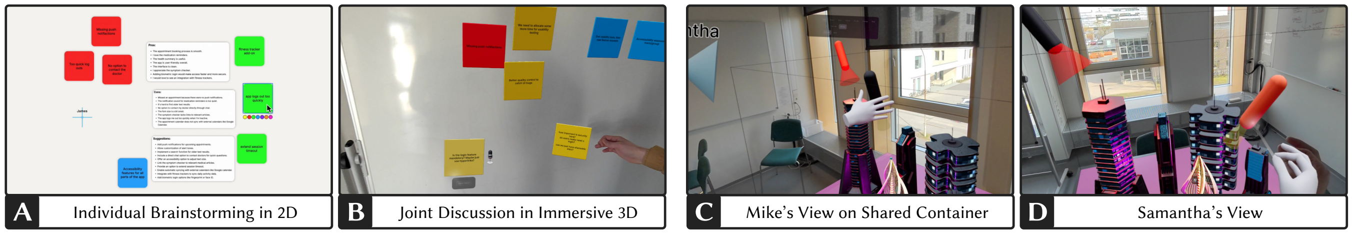
A Individual Brainstorming in 2D
B Joint Discussion in Immersive 3D
C Mike's View on Shared Container
D Samantha's View

Figure 7: Scenarios. Brainstorming (A–B) and Architecture (C–D). (A) Alex and James collaborating remotely using the 2D view. (B) Alex and James collaborating in the same office in immersive AR. (C) Remote collaboration using a shared container, Mike's view. (D) Samantha's view; she is fine-tuning a light in the scene in the container.

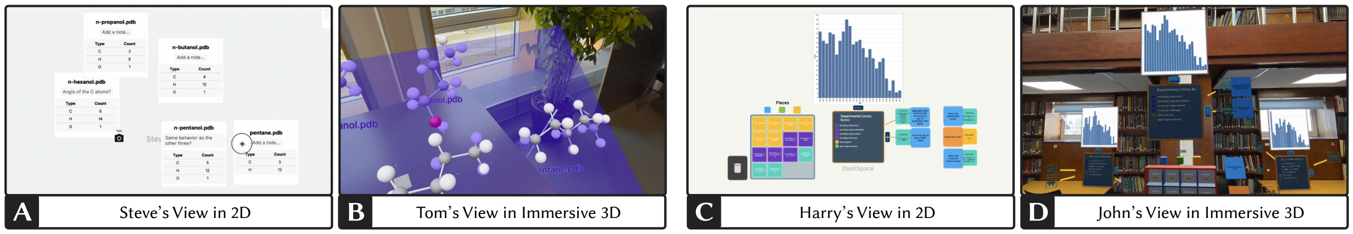

Figure 8: Scenarios. Molecule Structure (A–B) and Immersive Analytics (C–D). (A) Cross-dimensional collaboration, Steve's view in 2D. (B) Tom' view in immersive 3D. The blue plane is the projection plane Tom used to align Steve's view in 2D. (C) Harry's view of a data visualization in 2D during setup. (D) John's view in Immersive 3D after a collaborative analysis session.

in 3D and 2D[3], enabling the same interaction model for visual programming across 3D and 2D—in contrast to the third scenario, which uses heterogeneous element representations.

4.4.1 Import Data and Set Up Analysis in Desktop 2D. Harry, a lecturer in Computer Science, teaches Vega-Lite [48] as part of his visualization course. He decides to use the data visualization elements from Spatialstrates to explain the principles of the language using block-based visual programming. Using his laptop, Harry loads a dataset into Spatialstrates containing details of books in the departmental library. Harry adds a bookshelf, trash can, and some basic example transforms to the scene which he labels with sticky notes ahead of his tutorial session.

4.4.2 Tutorial on a Large Touchscreen Display. Harry uses a large touchscreen display to demonstrate the basic principles of Vega-Lite to students by dragging pieces out of the bookshelf. Harry combines the library dataset piece with a bar mark piece to create a group. Harry then pulls field name, field type, and transform pieces from the bookshelf, setting appropriate properties to each, arranging them around the visualization using DashSpace's proximity authoring and grouping features. The resulting bar chart summarizes the contents of the departmental library (Figure 8C).

4.4.3 Situating Data in Immersive 3D. Harry invites two students, John and Richard, to use Quest 3s to join him in an immersive 3D view of the analysis. Other students are able to follow along on their laptops using the link to the Spatialstrates workspace Harry had shared. John and Richard are instructed to clone a copy of Harry's original visualization and move their copy to different sections of the library. Using the transform piece, John filters his visualization by genre to show only Computer Science books. Richard filters books within the Electronic Engineering genre. John and Richard step back to compare all three charts now present in the library (Figure 8D) before returning to discussions with the rest of the class using the 2D touchscreen.

[3]In 3D, we reuse DashSpace's elements for authoring visualizations. In 2D, we use newly created 2D canvas components that recreate DashSpace's functionality in 2D.

5 TECHNICAL IMPLEMENTATION

Spatialstrates[4] builds on the DashSpace [4] codebase, which itself builds on the Webstrates [28] and Codestrates [5] platforms. Spatialstrates is a client-side application accessible in a web browser, can be run on any Webstrates server, and does not include any specialized server code. It is implemented in JavaScript, HTML, CSS and uses React, React Three Fiber,[5] and its WebXR integration.[6] The new 2D view is realized with the tldraw[7] infinite canvas.

The implementation of the 3D scene, the element architecture, and the immersive analytics components are reused from DashSpace, and we refer to the paper for their implementation [4]. The new element types, such as 3D models, flashlights, and molecules, use the `GLTFLoader` and `PDBLoader` components as well as other standard Three.js components like directional lights.

5.1 Projection Plane Optimization Using PCA

Each space stores its own projection plane consisting of a vector with 12 values—the position, the first and second principal component vectors, and the normal to the component vectors. Together,

[4]https://github.com/Webstrates/Spatialstrates
[5]https://github.com/pmndrs/react-three-fiber
[6]https://github.com/pmndrs/xr
[7]https://github.com/tldraw/tldraw

these are used to create the preview of the projection plane (Figure 3) and to project elements into 2D.

The projection plane can be updated automatically using PCA. We are using an open JavaScript implementation of PCA.[8] It takes the positions of all elements in a space and computes the principal component vectors, as well as the mean of all positions. These are then used to update the projection plane vector of a space. Alternatively, the projection plane can be updated manually using a movable box at the origin of the plane (Figure 3). In the automatic optimization, we prevent rotation around the z-axis to avoid a tilted projection plane, because in our testing a tilted plane caused confusion, as the upward direction in 2D might, e.g., point to the right in 3D.

5.2 Transclusion of Space Using a Slice Boundary

Each space stores its slice boundary in a vector with six values—the position and size (width, height, depth) of the boundary. The boundary of a space defines the size of a container embedding it as the source space. A boundary preview can be activated for the current space, allowing to modify its position or size. Changing the size of a container, likewise, changes the size of the slice boundary of its embedded space.

When a space is embedded, Spatialstrates, by default, renders only the elements that are within the slice boundary in the source space. However, containers have an option to show all elements, even ones that are outside the boundary.

5.3 Reprogrammability for XR

We designed Spatialstrates as an extensible platform for exploring spatial hypermedia. By building on the Webstrates [28] and Codestrates [5] stack, we inherit the ability to use Codestrates' Cauldron editor, a code editor that can be accessed from within the web browser by clicking on the "Edit" button in the corner of the interface (Figure 2).

Spatialstrates supports live code editing directly in the browser. It is designed to detect and selectively reload only modified elements while preserving the overall 3D scene. This allows for interactive, incremental changes without disrupting the spatial context. More fundamental changes, however, require a full refresh of the Three.js scene. All changes are propagated to other users in the shared space, enabling collaborative and remote reprogramming.

For instance, this allows Alice to reprogram the visual appearance of sticky notes on her laptop, while Bob sees the updates live in AR through a head-mounted display. Application logic can also be reprogrammed: Alice might modify an element that counts nearby sticky notes to also display a breakdown by color (see code in Appendix B).

We also make extensive use of Webstrates Package Manager (WPM) and the code of Spatialstrates is structured into portable WPM packages. This makes it possible to pick-and-mix features and element types per workspace, as well as extend the platform with new packages (see Appendix for an example package).

[8]https://github.com/mljs/pca

6 DISCUSSION

We evaluate Spatialstrates using Olsen's [40] heuristics for evaluating systems, discuss implications and possible directions for future work on open platforms and collaboration across 2D and 3D, and discuss limitations of Spatialstrates.

6.1 Systems-Oriented Evaluation

Through its web-based approach, Spatialstrates leverages *power in common infrastructure* [40], a foundational virtue in hypermedia systems. Interactive data visualizations from DashSpace implemented with Vega-Lite [48] can coexist with 3D molecular models, and we can leverage the `tldraw` framework for creating 2D infinite canvases. The **composability** ( ) in Spatialstrates demonstrates the *power in combination* where atomic elements *inductively combine* to form spaces that enable work on complex collaborative tasks.

The *generality* of Spatialstrates is evident through our diverse usage scenarios spanning brainstorming, data analytics, architectural design, and molecular visualization. Each scenario demonstrates how Spatialstrates adapts to different domains without requiring domain-specific customization of the core platform.

The **cooperability** ( ) in Spatialstrates provides a unique platform for cross-reality computing. It *reduces solution viscosity* through three key mechanisms: (1) relying on the web for XR, so deployment of new solutions is frictionless by loading a URL or reloading the browser; (2) leveraging the flexibility of the Webstrates platform for rapidly and collaboratively making changes to software with tools directly accessible in the running application; and (3) providing a streamlined API for extending Spatialstrates with new elements.

While web-based XR technology has limitations in areas such as space anchoring, it *empowers new design participants* by lowering barriers to entry. Web development skills are remarkably widespread,[9] making Spatialstrates accessible to a broad developer community. This is rare in XR development [8], which is currently dominated by 3D game engines.

Spatialstrates offers an *expressive match* for web developers by building directly on standard technologies. React, used by 39.5% of all respondents in the 2024 Stack Overflow Developer Survey [41], enables developers to fluidly transition between 2D and 3D content creation using familiar patterns, avoiding representational mismatch. The transclusion of spaces conceptually blends iframes with shared folders (as in Dropbox or Google Drive), creating a mental model that feels familiar to web users and developers alike.

Finally, Spatialstrates addresses a *problem not previously solved* by providing a fully web-based cross-reality platform with built-in mechanisms for enabling **congruency** (⇄) in cross-dimensional collaboration spaces. This congruency ensures that spatial content operations, avatar positions, and pointing gestures translate meaningfully across 2D and 3D environments .

[9]60% of respondents in Stack Overflow's 2024 developer survey report experience with JavaScript compared to 27% with C# (used in Unity) and 23% with C++ (used in Unreal Engine).

6.2 Towards an Open Platform for Cross-Reality Applications

Most XR applications are bespoke, with features implemented for the specific scenarios they explore. This fragmentation of XR development presents several challenges. XR applications are typically built as standalone software, requiring exclusive hardware access and offering minimal interoperability with other applications. Moreover, developers must recreate basic features when building applications for both desktop and XR environments, duplicating effort and increasing development costs.

Spatialstrates addresses these challenges by providing a single development paradigm for 2D and 3D. Developers can reuse application logic and create two views for each element type—one for 2D and one for 3D— reducing development overhead. Such a web-centered, interoperability-first approach aligns with Butcher et al. [8], who question whether native game engines are necessary for all XR applications; this paper aptly demonstrates how many essential XR features can be realized using open web standards.

This approach stands in stark contrast to the current dominant paradigm where XR applications are siloed within the ecosystems of game engines like Unity and Unreal [8]. While these engines offer powerful capabilities and up-to-date hardware compatibility, they constrain development to their specific workflows, tools, and licensing models. Spatialstrates demonstrates that web technologies—with their lower barriers to entry, platform independence, interoperability, and built-in collaboration capabilities—can serve as a viable alternative for many XR applications, particularly those focused on knowledge work, collaboration, and data visualization.

Presently, it may be hard to imagine how conventional applications such as video editors or word processors may benefit from collaborative continuous 2D or 3D interaction in XR. However, such tools may evolve in the future to leverage collaborative XR and spatial hypermedia, where spatial and desktop interaction complement each other. Spatialstrates represents our latest effort in reimagining software as *substrates*.[10] Rather than conventional applications, substrates are malleable, can be recombined, and evolve over time. With Videostrates [30], we have shown how video editing can be rethought as substrates-based software. We aim to explore Spatialstrates as a substrate for collaborative cross-reality video editing and other domains of desktop computing in the future.

6.3 Sharing Alike Across 2D and 3D

The avatars in Spatialstrates represent an initial step toward group awareness [21, 22] features across 2D and 3D environments. While effective in many scenarios, these awareness mechanisms still face limitations in certain situations. For example, the representation of a 2D cursor in 3D space provides limited positional information, and the projection of a 3D avatar to 2D can sometimes be ambiguous. More work is needed to develop richer awareness cues that maintain their semantic meaning when translated between dimensions.

The transclusion of space raises important privacy considerations. Our architecture scenario demonstrated how Samantha was able to remove one of Mike's models that he had included in their collaborative space. When moving or resizing a slice boundary, users may inadvertently reveal objects they had intentionally placed outside the shared area. This highlights how component access and permissions in Spatialstrates currently focus more on information hiding and encapsulation rather than privacy and security.

Future work should explore more sophisticated access control and permission models for spatial hypermedia. Kim et al. [27] studied this in depth, proposing territory-based permission models for collaborative 2D tabletop environments that could be adapted for cross-dimensional spaces. Rajaram et al. [42] elicit access-controlled sharing techniques for multi-user AR through scenario-based threat modeling. Such models and techniques could provide more nuanced control over shared content while maintaining the flexibility and fluidity that characterize effective collaboration.

Moreover, an important future direction is to explore alternative designs around congruency. While we focused on how to make user actions appear spatially congruent to collaborative partners, there can also be productivity benefits to the intentional design of asymmetry. For instance, currently, the relative positions and orientations of objects are consistent across both 2D and 3D with no possibility of divergence based on, e.g., user task, role or preferences. By weakening the requirement of objectivity, we can allow the subjective views of users to intentionally diverge [50]. For instance, avatar size disparities can provide awareness of different space zoom levels [16] or, in cross-reality scenarios, designers can exploit that 2D users can manipulate some content types more efficiently than XR users, and vice versa.

6.4 Limitations and Future Work

As a research prototype platform, Spatialstrates lacks various quality of life features found in established desktop and XR applications. E.g., the controller menu could be improved by using more advanced layouts for the buttons or even 3D icons. We also did not implement support for plane detection yet, which might help arranging elements on walls and surfaces like desks. Similarly, our implementation of avatars relies on 3D models of a Quest 3 headset and generic hand models, which could be improved by using full-body avatars, such as those available in the Meta Avatars SDK.

By virtue of being an interpreted environment running in a browser, WebXR applications such as Spatialstrates face performance limitations compared to native game engines. We observed occasional frame rate drops in complex scenes, particularly when multiple users interact with numerous elements simultaneously. While modern browsers continue to optimize JavaScript execution and WebGL rendering, this performance gap likely remains a constraint for computationally intensive applications. Future work could explore hybrid approaches that leverage WebAssembly for critical components while maintaining the accessibility and interoperability benefits of web technologies. We also believe that as WebXR adoption increases, so will its performance.

Spatialstrates only implements flat projection planes. Research has shown that users naturally arrange elements in cylindrical layouts around themselves [3], causing inevitable overlap when projected onto a flat plane. Curved projection planes could address this issue but introduce new challenges, such as handling edge cases

[10]Klokmose et al. [28] define substrates as: "*software artifacts that embody content, computation and interaction, effectively blurring the distinction between documents and applications. Substrates can evolve over time and shift roles, acting as what are traditionally considered documents in one context and applications in another, or a mix of the two.*" For a recent discussion and conceptual model of substrates, see [35].

where curves meet in full cylindrical arrangements. Our current PCA-based optimization algorithm considers only element positions when updating the projection plane. Future versions should account for element size and rotation while supporting curved projection geometries.

While Spatialstrates supports some code modifications at runtime, it lacks a fully consistent live reprogramming experience. Some changes require browser reloads, disrupting development workflow. Varv [6] offers a promising solution through its live and declarative programming model, as demonstrated in the Mirrorverse [19] video conferencing platform. Extending Varv to work with XR technologies such as React Three Fiber would enhance Spatialstrates' reprogrammability liveness. Though we consider this integration beyond our current scope, it remains a compelling direction for future work.

Finally, Spatialstrates' current implementation has architectural limitations to its spatial transclusion functionality. All spaces are embedded within a single webstrate, simulating rather than truly implementing independent spatial components. This approach prevents proper per-space access control, as Webstrates only supports access control at the document level. Advancing Spatialstrates toward a production-ready platform would require restructuring the architecture so each space is its own webstrate. Furthermore, migrating to MyWebstrates [29], a decentralized version of the Webstrates platform, would enhance architectural flexibility and enable more robust permission models for collaborative spatial computing.

7 CONCLUSION

We have presented Spatialstrates, an implementation of spatial hypermedia reimagined using current consumer-level XR hardware and modern open web technologies. Spatialstrates demonstrates how spatial hypermedia principles can unite the divided worlds of 2D desktop and 3D immersive environments through composability (🧩) of heterogeneous content, cooperability (👥) across devices and dimensions, and congruency (⇄) of spatial representations. Our four scenarios—brainstorming, architectural design, molecular science, and immersive analytics—validate these properties in practical contexts that bridge information spaces. By building on standards and open platforms, Spatialstrates offers an environment for exploring spatial hypermedia across 2D desktop and immersive 3D. As XR hardware becomes increasingly mainstream, the path forward for spatial computing is not merely an incremental improvement but a fundamental transformation that will advance our field far beyond what has come before and leading toward a far better blended and extended computing experience than we have previously known.

ACKNOWLEDGMENTS

This work was partly supported by the Villum Investigator grant VL-54492 by Villum Fonden, the Aarhus University Research Foundation (AUFF) grant AUFF-E-2022-9-33, and the Independent Research Fund Denmark (DFF) grant 10.46540/3104-00008B. Any opinions, findings, and conclusions or recommendations expressed here are those of the authors and do not necessarily reflect the views of the funding agency.

REFERENCES

[1] Jonas Auda, Uwe Gruenefeld, Sarah Faltaous, Sven Mayer, and Stefan Schneegass. 2023. A Scoping Survey on Cross-Reality Systems. *Comput. Surveys* 56, 4, Article 83 (Oct. 2023), 38 pages. https://doi.org/10.1145/3616536

[2] E. Barba, B. MacIntyre, and E.D. Mynatt. 2012. Here We Are! Where Are We? Locating Mixed Reality in The Age of the Smartphone. *Proc. IEEE* 100, 4 (April 2012), 929–936. https://doi.org/10.1109/JPROC.2011.2182070

[3] Andrea Batch, Andrew Cunningham, Maxime Cordeil, Niklas Elmqvist, Tim Dwyer, Bruce H. Thomas, and Kim Marriott. 2020. There Is No Spoon: Evaluating Performance, Space Use, and Presence with Expert Domain Users in Immersive Analytics. *IEEE Transactions on Visualization and Computer Graphics* 26, 1 (2020), 536–546. https://doi.org/10.1109/TVCG.2019.2934803

[4] Marcel Borowski, Peter W. S. Butcher, Janus Bager Kristensen, Jonas Oxenbøll Petersen, Panagiotis D. Ritsos, Clemens N. Klokmose, and Niklas Elmqvist. 2025. DashSpace: A Live Collaborative Platform for Immersive and Ubiquitous Analytics. *IEEE Transactions on Visualization and Computer Graphics* PP, 99 (2025), 1–13. https://doi.org/10.1109/TVCG.2025.3537679

[5] Marcel Borowski, Janus Bager Kristensen, Rolf Bagge, and Clemens N. Klokmose. 2021. *Codestrates v2: A Development Platform for Webstrates.* Technical Report. Aarhus University. https://pure.au.dk/portal/en/publications/codestrates-v2-a-development-platform-for-webstrates(66e1d4d9-27da-4f6b-85b3-19b0993caf22).html

[6] Marcel Borowski, Luke Murray, Rolf Bagge, Janus Bager Kristensen, Arvind Satyanarayan, and Clemens Nylandsted Klokmose. 2022. Varv: Reprogrammable Interactive Software as a Declarative Data Structure. In *Proceedings of the ACM Conference on Human Factors in Computing Systems.* ACM, New York, NY, USA, 492:1–492:20. https://doi.org/10.1145/3491102.3502064

[7] Niels Olof Bouvin and Clemens Nylandsted Klokmose. 2016. Classical Hypermedia Virtues on the Web with Webstrates. In *Proceedings of the ACM Conference on Hypertext and Social Media.* ACM, New York, NY, USA, 207–212. https://doi.org/10.1145/2914586.2914622

[8] Peter W. S. Butcher, Andrea Batch, David Saffo, Blair MacIntyre, Niklas Elmqvist, and Panagiotis D. Ritsos. 2024. Is Native Naïve? Comparing Native Game Engines and WebXR as Immersive Analytics Development Platforms. *IEEE Computer Graphics and Applications* 44, 3 (2024), 91–98. https://doi.org/10.1109/MCG.2024.3367422

[9] Peter W. S. Butcher, Nigel W. John, and Panagiotis D. Ritsos. 2021. VRIA: A Web-Based Framework for Creating Immersive Analytics Experiences. *IEEE Transactions on Visualization and Computer Graphics* 27, 7 (2021), 3213–3225. https://doi.org/10.1109/TVCG.2020.2965109

[10] Simon Butscher, Sebastian Hubenschmid, Jens Müller, Johannes Fuchs, and Harald Reiterer. 2018. Clusters, Trends, and Outliers: How Immersive Technologies Can Facilitate the Collaborative Analysis of Multidimensional Data. In *Proceedings of the ACM Conference on Human Factors in Computing Systems.* ACM, New York, NY, USA, 1–12. https://doi.org/10.1145/3173574.3173664

[11] Bill Buxton. 2009. Mediaspace – Meaningspace – Meetingspace. In *Media Space 20 + Years of Mediated Life*, Steve Harrison (Ed.). Springer London, London, 217–231. https://doi.org/10.1007/978-1-84882-483-6_13

[12] Daniel Immanuel Fink, Johannes Zagermann, Harald Reiterer, and Hans-Christian Jetter. 2022. Re-locations: Augmenting Personal and Shared Workspaces to Support Remote Collaboration in Incongruent Spaces. *Proceedings of the ACM on Human-Computer Interaction* 6, ISS, Article 556 (2022), 30 pages. https://doi.org/10.1145/3567709

[13] Bernhard Fröhler, Christoph Anthes, Fabian Pointecker, Judith Friedl, Daniel Schwajda, Andreas Riegler, Shailesh Tripathi, Clemens Holzmann, Manuel Brunner, Herbert Jodlbauer, Hans-Christian Jetter, and Christoph Heinzl. 2022. A Survey on Cross-Virtuality Analytics. *Computer Graphics Forum* 41, 1 (2022), 465–494. https://doi.org/10.1111/cgf.14447

[14] Jens Emil Grønbæk, Banu Saatçi, Carla F. Griggio, and Clemens Nylandsted Klokmose. 2021. MirrorBlender: Supporting Hybrid Meetings with a Malleable Video-Conferencing System. In *Proceedings of the ACM Conference on Human Factors in Computing Systems.* ACM, New York, NY, USA, 451:1–451:13. https://doi.org/10.1145/3411764.3445698

[15] Jens Emil Sloth Grønbæk, Ken Pfeuffer, Eduardo Velloso, Morten Astrup, Melanie Isabel Sønderkær Pedersen, Martin Kjær, Germán Leiva, and Hans Gellersen. 2023. Partially Blended Realities: Aligning Dissimilar Spaces for Distributed Mixed Reality Meetings. In *Proceedings of the ACM Conference on Human Factors in Computing Systems.* ACM, New York, NY, USA, 456:1–456:16. https://doi.org/10.1145/3544548.3581515

[16] Jens Emil Sloth Grønbæk, Juan Sánchez Esquivel, Germán Leiva, Eduardo Velloso, Hans Gellersen, and Ken Pfeuffer. 2024. Blended Whiteboard: Physicality and Reconfigurability in Remote Mixed Reality Collaboration. In *Proceedings of the ACM Conference on Human Factors in Computing Systems.* ACM, New York, NY, USA, 798:1–798:16. https://doi.org/10.1145/3613904.3642293

[17] Kaj Grønbæk, Jannie F. Kristensen, Peter Ørbæk, and Mette Agger Eriksen. 2003. "Physical hypermedia": organising collections of mixed physical and digital material. In *Proceedings of the ACM Conference on Hypertext and Hypermedia.*

ACM, New York, NY, USA, 10–19. https://doi.org/10.1145/900051.900056

[18] Kaj Grønbæk, Peter Posselt Vestergaard, and Peter Ørbæk. 2002. Towards geo-spatial hypermedia: Concepts and prototype implementation. In *Proceedings of the ACM Conference on Hypertext and Hypermedia*. ACM, New York, NY, USA, 117–126. https://doi.org/10.1145/513338.513370

[19] Jens Emil Grønbæk, Marcel Borowski, Eve Hoggan, Wendy E. Mackay, Michel Beaudouin-Lafon, and Clemens N. Klokmose. 2023. Mirrorverse: Live Tailoring of Video Conferencing Interfaces. In *Proceedings of the ACM Symposium on User Interface Software and Technology*. ACM, New York, NY, USA, 14:1–14:14. https://doi.org/10.1145/3586183.3606767

[20] Jan Gugenheimer, Evgeny Stemasov, Julian Frommel, and Enrico Rukzio. 2017. ShareVR: Enabling Co-Located Experiences for Virtual Reality between HMD and Non-HMD Users. In *Proceedings of the ACM Conference on Human Factors in Computing Systems*. ACM, New York, NY, USA, 4021–4033. https://doi.org/10.1145/3025453.3025683

[21] Carl Gutwin and Saul Greenberg. 1998. Design for Individuals, Design for Groups: Tradeoffs between Power and Workspace Awareness. In *Proceedings of the ACM Conference on Computer-Supported Cooperative Work and Social Computing*. ACM, New York, NY, USA, 207–216. https://doi.org/10.1145/289444.289495

[22] Carl Gutwin and Saul Greenberg. 1998. Effects of Awareness Support on Groupware Usability. In *Proceedings of the ACM Conference on Human Factors in Computing Systems*. ACM, New York, NY, USA, 511–518. https://doi.org/10.1145/274644.274713

[23] Tom Horak, Sriram Karthik Badam, Niklas Elmqvist, and Raimund Dachselt. 2018. When David Meets Goliath: Combining Smartwatches with a Large Vertical Display for Visual Data Exploration. In *Proceedings of the ACM Conference on Human Factors in Computing Systems*. ACM, New York, NY, USA, 1–13. https://doi.org/10.1145/3173574.3173593

[24] Xincheng Huang and Robert Xiao. 2024. SurfShare: Lightweight Spatially Consistent Physical Surface and Virtual Replica Sharing with Head-mounted Mixed-Reality. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 7, 4, Article 162 (Jan. 2024), 24 pages. https://doi.org/10.1145/3631418

[25] Sebastian Hubenschmid, Jonathan Wieland, Daniel Immanuel Fink, Andrea Batch, Johannes Zagermann, Niklas Elmqvist, and Harald Reiterer. 2022. ReLive: Bridging In-Situ and Ex-Situ Visual Analytics for Analyzing Mixed Reality User Studies. In *Proceedings of the ACM Conference on Human Factors in Computing Systems*. ACM, New York, NY, USA, 24:1–24:20. https://doi.org/10.1145/3491102.3517550

[26] Hans-Christian Jetter, Jan-Henrik Schröder, Jan Gugenheimer, Mark Billinghurst, Christoph Anthes, Mohamed Khamis, and Tiare Feuchtner. 2021. Transitional interfaces in mixed and cross-reality: A new frontier?. In *Companion Proceedings of the ACM Conference on Interactive Surfaces and Spaces*. ACM, New York, NY, USA, 46–49. https://doi.org/10.1145/3447932.3487940

[27] KyungTae Kim, Waqas Javed, Cary Williams, Niklas Elmqvist, and Pourang Irani. 2010. Hugin: a framework for awareness and coordination in mixed-presence collaborative information visualization. In *Proceedings of the ACM Conference on Interactive Tabletops and Surfaces*. ACM, New York, NY, USA, 231–240. https://doi.org/10.1145/1936652.1936694

[28] Clemens N. Klokmose, James R. Eagan, Siemen Baader, Wendy Mackay, and Michel Beaudouin-Lafon. 2015. Webstrates: Shareable Dynamic Media. In *Proceedings of the ACM Symposium on User Interface Software and Technology*. ACM, New York, NY, USA, 280–290. https://doi.org/10.1145/2807442.2807446

[29] Clemens Nylandsted Klokmose, James R. Eagan, and Peter van Hardenberg. 2024. MyWebstrates: Webstrates as Local-first Software. In *Proceedings of the ACM Symposium on User Interface Software and Technology*. ACM, New York, NY, USA, 42:1–42:12. https://doi.org/10.1145/3654777.3676445

[30] Clemens N. Klokmose, Christian Remy, Janus Bager Kristensen, Rolf Bagge, Michel Beaudouin-Lafon, and Wendy Mackay. 2019. Videostrates: Collaborative, Distributed and Programmable Video Manipulation. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology* (New Orleans, LA, USA) *(UIST '19)*. Association for Computing Machinery, New York, NY, USA, 233–247. https://doi.org/10.1145/3332165.3347912

[31] David Ledo, Steven Houben, Jo Vermeulen, Nicolai Marquardt, Lora Oehlberg, and Saul Greenberg. 2018. Evaluation Strategies for HCI Toolkit Research. In *Proceedings of the ACM Conference on Human Factors in Computing Systems*. ACM, New York, NY, USA, 36:1–36:17. https://doi.org/10.1145/3173574.3173610

[32] Benjamin Lee, Xiaoyun Hu, Maxime Cordeil, Arnaud Prouzeau, Bernhard Jenny, and Tim Dwyer. 2021. Shared Surfaces and Spaces: Collaborative Data Visualisation in a Co-located Immersive Environment. *IEEE Transactions on Visualization and Computer Graphics* 27, 2 (2021), 1171–1181. https://doi.org/10.1109/TVCG.2020.3030450

[33] Blair MacIntyre, Alex Hill, Hafez Rouzati, Maribeth Gandy, and Brian Davidson. 2011. The Argon AR Web Browser and standards-based AR application environment. In *Proceedings of the IEEE International Symposium on Mixed and Augmented Reality*. Los Alamitos, CA, USA, IEEE Computer Society, 65–74. https://doi.org/10.1109/ISMAR.2011.6092371

[34] Blair MacIntyre, Hafez Rouzati, and Martin Lechner. 2013. Walled Gardens: Apps and Data as Barriers to Augmenting Reality. *IEEE Computer Graphics and Applications* 33, 3 (2013), 77–81. https://doi.org/10.1109/MCG.2013.51

[35] Wendy E. Mackay and Michel Beaudouin-Lafon. 2025. Interaction Substrates: Combining Power and Simplicity in Interactive Systems. In *Proceedings of the CHI Conference on Human Factors in Computing Systems (CHI '25)*. ACM, Yokohama, Japan, 1–16. https://doi.org/10.1145/3706598.3714006

[36] Catherine C. Marshall and Frank M. Shipman. 1995. Spatial hypertext: designing for change. *Commun. ACM* 38, 8 (1995), 88–97. https://doi.org/10.1145/208344.208350

[37] Elisabeth Mayer, Francesco Chiossi, and Sven Mayer. 2024. Crossing Mixed Realities: A Review for Transitional Interfaces Design. In *Proceedings of Mensch und Computer*. ACM, New York, NY, USA, 629–634. https://doi.org/10.1145/3670653.3677481

[38] Miro 2025. *Miro Online Collaborative Whiteboard*. Miro. https://miro.com

[39] Theodor Holm Nelson. 1995. The Heart of Connection: Hypermedia Unified by Transclusion. *Commun. ACM* 38, 8 (1995), 31–33. https://doi.org/10.1145/208344.208351

[40] Dan R. Olsen. 2007. Evaluating user interface systems research. In *Proceedings of the ACM Symposium on User Interface Software and Technology*. ACM, New York, NY, USA, 251–258. https://doi.org/10.1145/1294211.1294256

[41] Stack Overflow. 2024. 2024 Stack Overflow Developer Survey. https://survey.stackoverflow.co/2024/ Accessed: 2025-04-03.

[42] Shwetha Rajaram, Chen Chen, Franziska Roesner, and Michael Nebeling. 2023. Eliciting Security & Privacy-Informed Sharing Techniques for Multi-User Augmented Reality. In *Proceedings of the ACM Conference on Human Factors in Computing Systems*. ACM, New York, NY, USA, 98:1–98:17. https://doi.org/10.1145/3544548.3581089

[43] Nico Reski, Aris Alissandrakis, and Andreas Kerren. 2022. An Empirical Evaluation of Asymmetric Synchronous Collaboration Combining Immersive and Non-Immersive Interfaces Within the Context of Immersive Analytics. *Frontiers in Virtual Reality* 2, Article 743445 (2022), 29 pages. https://doi.org/10.3389/frvir.2021.743445

[44] Jonathan C. Roberts, Panagiotis D. Ritsos, Sriram Karthik Badam, Dominique Brodbeck, Jessie Kennedy, and Niklas Elmqvist. 2014. Visualization Beyond the Desktop – The Next Big Thing. *IEEE Computer Graphics and Applications* 34, 6 (Nov. 2014), 26–34. https://doi.org/10.1109/MCG.2014.82

[45] R. M. Rohrer and E. Swing. 1997. Web-based information visualization. *IEEE Computer Graphics and Applications* 17, 4 (July 1997), 52–59. https://doi.org/10.1109/38.595269

[46] Luis Romero and Nuno Correia. 2003. HyperReal: a hypermedia model for mixed reality. In *Proceedings of the ACM Conference on Hypertext and Hypermedia*. ACM, New York, NY, USA, 2–9. https://doi.org/10.1145/900051.900055

[47] David Saffo, Andrea Batch, Cody Dunne, and Niklas Elmqvist. 2023. Through Their Eyes and In Their Shoes: Providing Group Awareness During Collaboration Across Virtual Reality and Desktop Platforms. In *Proceedings of the ACM Conference on Human Factors in Computing Systems*. ACM, New York, NY, USA, 383:1–383:15. https://doi.org/10.1145/3544548.3581093

[48] Arvind Satyanarayan, Dominik Moritz, Kanit Wongsuphasawat, and Jeffrey Heer. 2017. Vega-Lite: A Grammar of Interactive Graphics. *IEEE Transactions on Visualization and Computer Graphics* 23, 1 (2017), 341–350. https://doi.org/10.1109/TVCG.2016.2599030

[49] Jan-Henrik Schröder, Daniel Schacht, Niklas Peper, Anita Marie Hamurculu, and Hans-Christian Jetter. 2023. Collaborating Across Realities: Analytical Lenses for Understanding Dyadic Collaboration in Transitional Interfaces. In *Proceedings of the ACM Conference on Human Factors in Computing Systems*. ACM, New York, NY, USA, 97:1–97:16. https://doi.org/10.1145/3544548.3580879

[50] Dave Snowdon, Chris Greenhalgh, and Steve Benford. 1995. What you see is not what I see: Subjectivity in virtual environments. *Framework for Immersive Virtual Environments (FIVE'95)* 1 (1995).

[51] Mark Stefik, Daniel G. Bobrow, Gregg Foster, Stan Lanning, and Deborah G. Tatar. 1987. WYSIWIS Revised: Early Experiences with Multiuser Interfaces. *ACM Transactions on Information Systems* 5, 2 (1987), 147–167. https://doi.org/10.1145/27636.28056

[52] Balasaravanan Thoravi Kumaravel, Fraser Anderson, George Fitzmaurice, Bjoern Hartmann, and Tovi Grossman. 2019. Loki: Facilitating Remote Instruction of Physical Tasks Using Bi-Directional Mixed-Reality Telepresence. In *Proceedings of the ACM Symposium on User Interface Software and Technology*. ACM, New York, NY, USA, 161–174. https://doi.org/10.1145/3332165.3347872

[53] Balasaravanan Thoravi Kumaravel and Björn Hartmann. 2022. Interactive Mixed-Dimensional Media for Cross-Dimensional Collaboration in Mixed Reality Environments. *Frontiers in Virtual Reality* 3 (2022), 19 pages. https://doi.org/10.3389/frvir.2022.766336

[54] Wai Tong, Meng Xia, Kam Kwai Wong, Doug A. Bowman, Ting-Chuen Pong, Huamin Qu, and Yalong Yang. 2023. Towards an Understanding of Distributed Asymmetric Collaborative Visualization on Problem-solving. In *Proceedings of the IEEE Conference on Virtual Reality*. IEEE Computer Society, Los Alamitos, CA, USA, 387–397. https://doi.org/10.1109/VR55154.2023.00054

[55] W3C. 2017. *WebXR*. [Online]. Available https://www.w3.org/TR/webxr/. Accessed: 07/11/22.

[56] Cheng Yao Wang, Hyunju Kim, Payod Panda, Eyal Ofek, Mar Gonzalez Franco, and Andrea Stevenson Won. 2024. MRTransformer: Transforming Avatar Non-verbal Behavior for Remote MR Collaboration in Incongruent Spaces. In *Adjunct Proceedings of the IEEE International Symposium on Mixed and Augmented Reality Adjunct*. IEEE Computer Society, Los Alamitos, CA, USA, 545–548. https://doi.org/10.1109/ISMAR-Adjunct64951.2024.00157

[57] Xiyao Wang, Lonni Besançon, David Rousseau, Mickaël Sereno, Mehdi Ammi, and Tobias Isenberg. 2020. Towards an Understanding of Augmented Reality Extensions for Existing 3D Data Analysis Tools. In *Proceedings of the ACM Conference on Human Factors in Computing Systems*. ACM, New York, NY, USA, 528:1–528:13. https://doi.org/10.1145/3313831.3376657

[58] Emily Wong, Adélaïde Genay, Jens Emil Grønbœk, and Eduardo Velloso. 2025. Spatial Heterogeneity in Distributed Mixed Reality Collaboration. In *Proceedings of the ACM Conference on Human Factors in Computing Systems*. ACM, New York, NY, USA, to appear.

A ADDING ELEMENTS TO SPATIALSTRATES

This code example demonstrates the code required to create a custom `MyElement` element type, which renders a red box in 3D and a red rectangle in 2D. The Varv concept defines the new element type and inherits the properties of a generic element—called a "Movable." We then provide a 3D representation using the `<Movable>` tag from Spatialstrates, and a 2D representation by providing a custom shape type for tldraw.

```
{
  "concepts": {
    "MyElement": {
      "extensions": {
        "inject": [ "Movable" ]
      }
    }
  }
}
```

Listing 1: The Varv concept definition of the example element.

```
import React from 'react';
const { useMemo } = React;
import { Varv, useProperty } from '#VarvReact';
import { Movable } from '#Movable .default';

function MyElement3D() {
  const handle = useMemo(() => (
    <mesh>
      <boxGeometry args={[0.3, 0.3, 0.3]} />
      <meshStandardMaterial color="red" />
    </mesh>
  ), []);

  return <Movable handle={handle} upright={false} />;
}

export function Main() {
  const [conceptType] = useProperty('concept::name');
  return conceptType === 'MyElement' ? <MyElement3D /> :
       null;
}
```

Listing 2: The 3D representation of the example element.

```
import React from 'react';
import { HTMLContainer, } from 'tldraw';
import { CANVAS_SCALE } from '#Spatialstrates .canvas-
     utils';
import { MovableShapeUtil, MovableVarvScope } from '#
     Movable .movable-shape';

function MyElementShape({ shape }) {
  return <HTMLContainer className="clemens-shape" style =
       {{
    transform: 'translate(-50%, -50%)',
    width: shape.props.w + 'px',
    height: shape.props.h + 'px',
    backgroundColor: 'red'
  }}>
    <div>My Element</div>
  </HTMLContainer>;
}

class MyElementShapeUtil extends MovableShapeUtil {
  static type = 'MyElement';

  getDefaultProps() {
    return Object.assign(super.getDefaultProps(), {
      w: CANVAS_SCALE * 0.3,
      h: CANVAS_SCALE * 0.3
    });
  }

  component(shape) {
    return <MovableVarvScope shape={ shape }>
      <MyElementShape shape={ shape } />
    </MovableVarvScope>;
  }
}

export const Main = MyElementShapeUtil;
```

Listing 3: The 2D representation of the example element.

B REPROGRAMMING LOGIC

This code implements the `StickyNoteCounter` element type (Figure 9) that counts the number of notes in its proximity (as described in Section 5.3. Lines 24, 25, and 36–45 were added/modified to add the additional functionality of counting sticky notes for each color.

```
1  const frameGeometry = new RoundedBoxGeometry(0.3, 0.3, 0.005, 1);
2  const frameMaterial = new MeshStandardMaterial({ color: 'hsl(49, 0%, 60%)', metalness: 0.2, roughness: 0.5 });
3
4  const MAX_DISTANCE = 0.5;
5
6  function StickyNoteCounter() {
7    const [text, setText] = useState('');
8    const [position] = useProperty('position');
9    const [space] = useProperty('space');
10
11   useEffect(() => {
12     if (!space || !position) return;
13
14     const runAsnyc = async () => {
15       const stickyNoteUUIDs = await VarvEngine.lookupInstances(['StickyNote'], FilterAction.constructFilter({
16         property: 'space',
17         equals: space
18       }));
19
20       const stickyNoteConcept = await VarvEngine.getConceptFromType('StickyNote');
21       const stickyNotes = [];
22       for (const uuid of stickyNoteUUIDs) {
23         const position = await stickyNoteConcept.getPropertyValue(uuid, 'position');
24         const color = await stickyNoteConcept.getPropertyValue(uuid, 'color');
25         stickyNotes.push({ uuid, position, color });
26       }
27
28       const stickyNotesInRange = stickyNotes.filter(note => {
29         const notePosition = new Vector3(...note.position);
30         const currentPosition = new Vector3(...position);
31         return notePosition.distanceTo(currentPosition) <= MAX_DISTANCE;
32       });
33
34       let result = `There are ${stickyNotesInRange.length} sticky notes nearby.`;
35
36       const colorCounts = stickyNotesInRange.reduce((acc, note) => {
37         acc[note.color] = (acc[note.color] || 0) + 1;
38         return acc;
39       }, {});
40
41       if (Object.keys(colorCounts).length > 0) {
42         result += '\n\nColors:' + Object.entries(colorCounts).map(([color, count]) => {
43           return `\n${color}: ${count}`;
44         }).join('');
45       }
46
47       setText(result);
48     };
49
50     runAsnyc();
51   }, [position, space]);
52
53   const handle = useMemo(() => <mesh
54       geometry={ frameGeometry }
55       material = { frameMaterial }
56       position = { [0, 0.025, 0]}
57     />, []);
58
59   return <Movable handle={ handle } upright = { false} >
60     <Text position={ [0, 0.025, 0.003] }
61         maxWidth = { 0.27}
62         textAlign = 'center'
63         anchorX = 'center'
64         anchorY = 'middle'
65         color = 'black'
66         fontSize = { 0.02} >
67     { text }
68     </Text>
69   </Movable>;
70 }
```

Listing 4: The Sticky Note Counter element for 3D space.

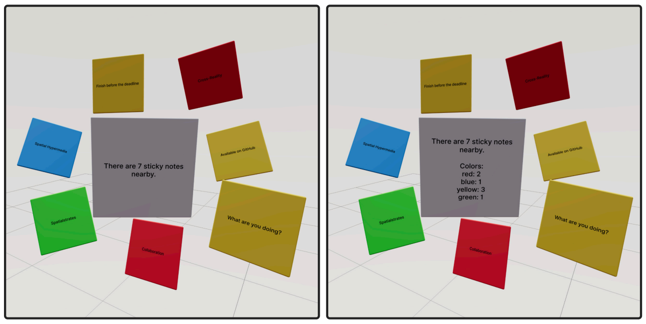

Figure 9: Sticky Note Counter. The logic of elements can be reprogrammed live. The left side shows a basic version with the count of close by notes, and the right side shows an extended version that also counts the notes by color.