Choose Your Reference Frame Right: An Immersive Authoring Technique for Creating Reactive Behavior

Sevinc Eroglu
sevinc.eroglu@rwth-aachen.de
RWTH Aachen University
Aachen, Germany

Patric Schmitz
patric.schmitz@cs.rwth-aachen.de
RWTH Aachen University
Aachen, Germany

Kilian Sinke
kilian.sinke@rwth-aachen.de
RWTH Aachen University
Aachen, Germany

David Anders
david.anders@rwth-aachen.de
RWTH Aachen University
Aachen, Germany

Torsten W. Kuhlen
kuhlen@vr.rwth-aachen.de
RWTH Aachen University
Aachen, Germany

Benjamin Weyers
weyers@uni-trier.de
University of Trier
Trier, Germany

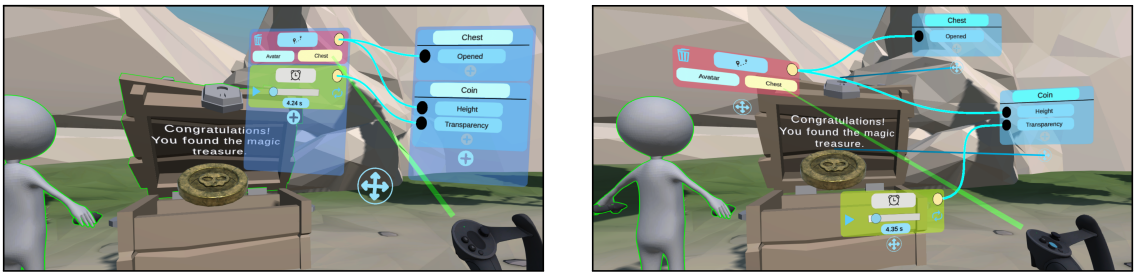


Figure 1: A user creates reactive behavior by *modulation mapping*. *Surround-referenced* layout (left): menu elements are located on a panel that travels with the user. *Object-referenced* layout (right): menu elements are attached directly to scene objects.

ABSTRACT

Immersive authoring enables content creation for virtual environments without a break of immersion. To enable immersive authoring of reactive behavior for a broad audience, we present modulation mapping, a simplified visual programming technique. To evaluate the applicability of our technique, we investigate the role of reference frames in which the programming elements are positioned, as this can affect the user experience. Thus, we developed two interface layouts: "surround-referenced" and "object-referenced". The former positions the programming elements relative to the physical tracking space, and the latter relative to the virtual scene objects. We compared the layouts in an empirical user study (n = 34) and found the surround-referenced layout faster, lower in task load, less cluttered, easier to learn and use, and preferred by users. Qualitative feedback, however, revealed the object-referenced layout as more intuitive, engaging, and valuable for visual debugging. Based on the results, we propose initial design implications for immersive authoring of reactive behavior by visual programming. Overall, modulation mapping was found to be an effective means for creating reactive behavior by the participants.

VRST '24, October 9–11, 2024, Trier, Germany

ACM ISBN 979-8-4007-0535-9/24/10
https://doi.org/10.1145/3641825.3687744

CCS CONCEPTS

• **Human-centered computing** → **Virtual reality**; **User interface design**; **User studies**.

KEYWORDS

Virtual Reality, Immersive Authoring, Visual Programming, Spatial Reference Frames, Empirical Evaluation

ACM Reference Format:
Sevinc Eroglu, Patric Schmitz, Kilian Sinke, David Anders, Torsten W. Kuhlen, and Benjamin Weyers. 2024. Choose Your Reference Frame Right: An Immersive Authoring Technique for Creating Reactive Behavior. In *30th ACM Symposium on Virtual Reality Software and Technology (VRST '24), October 9–11, 2024, Trier, Germany*. ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3641825.3687744

1 INTRODUCTION

Interactivity is an important aspect of virtual environments (VE), enhancing user engagement and creating compelling experiences [39, 52, 57]. When authoring VEs, it is thus necessary to define how scene objects react to user interactions or the state of other objects in the scene. Furthermore, the progression of time may result in dynamic changes to the VE. These dynamic and interactive state changes may collectively be termed *reactive behavior* [60].

In traditional workflows, reactive behavior is created offline using desktop-based programming environments, requiring programming skills and repeatedly entering and exiting the VE for testing. Immersive authoring tools address these challenges by enabling users to create and edit scenarios directly in VR [22, 38, 58]. Recently,

immersive authoring tools that expand upon visual programming gained attention [26, 60]. Some focus on programming with basic building blocks [37, 47], while others simplify for end-users by providing predefined functions [3, 18] and abstractions [60].

Inspired by these works, we aim to further simplify the process of creating reactive behavior while immersed. To this end, we present *modulation mapping* to define immediate effects on scene entities. Our method defines a workflow, in which a developer prepares application-specific degrees of freedom to control object behavior, exposed as object properties. In the VE, the user interactively maps input sources to these predefined properties by drawing connections. Input sources can be user input, the progression of time, properties of other objects, or spatial relations between objects. To make the technique approachable by novices, we only allow direct connections from sources to target properties. To retain expressiveness, behaviors can be interactively customized using visually represented mapping functions chosen from a set of templates.

Possible use cases for *modulation mapping* include teaching scenarios, where instructors may define cause-and-effect relations between objects to illustrate processes in, e.g., mechanics or chemistry. In scientific visualization, domain experts may explore data by mapping user interactions, such as joystick movements to visualization parameters. In game design and virtual storytelling, creators may enrich the scene by adding interactive elements (see Fig. 4).

Because our method enables users to define reactive behavior for objects that are located within the virtual scene, there is an inherent spatial relation between the programming elements and the objects. Therefore, an important factor for our approach's applicability is the spatial reference frame in which the programming elements are positioned. While the role of reference frames in information visualization [9, 19, 23] and menu interactions [15, 34] has been explored, the impact on the immersive authoring of reactive behavior through visual programming remains uninvestigated.

To support the design of immersive authoring techniques, our research investigates the role of reference frames on the applicability of modulation mapping for creating reactive behavior. Therefore, we designed a *surround-referenced* and an *object-referenced* interface layout, previously used by other immersive authoring environments [13, 18, 47] (see Section 2.2 for more detailed terminology). In a study ($n = 34$), we compared these layouts, revealing a significant effect of reference frame choice on the technique's applicability. From our findings, we derive initial interface design recommendations to guide future research on interface layouts for immersive creation of reactive behaviors. In summary, we contribute

- *Modulation mapping* as a simplified immersive visual programming technique to create reactive behavior.
- An empirical evaluation of *surround-* vs. *object-referenced* interface layouts in the context of modulation mapping.
- Initial *design implications* for immersive authoring of reactive behavior by visual programming.

2 RELATED WORK

2.1 Immersive Authoring of Reactive Behavior

Immersive content authoring may include different tasks, such as 3D modeling [1, 21, 28, 42], animation [2, 6, 55], scene arrangement [5, 16, 22, 56], and defining reactive behavior. Our work does not address all aspects of immersive authoring, but focuses on adding reactive behavior to a previously created virtual environment. In this section, we give an overview of similar methods and tools that enable the immersive definition of reactive behavior.

Making objects react to user input or other objects is typically achieved by programming. Desktop-based authoring environments use text-based or visual programming languages (VPL), the latter being simpler to use for creators who are not trained programmers [46]. Such simplified programming interfaces are desirable in VR due to ergonomics and the limitations of text entry [17, 30].

Several immersive VPL [18, 29, 35, 51, 58, 60] expand upon the dataflow programming paradigm, which models the flow of data and interactions between objects as directed graphs [48]. Steed and Slater [51] proposed an early dataflow VPL with three types of nodes (source, filter, receptor) that users connect to define behavior. *Ivy* by Ens et al. [18] is a spatially situated dataflow programming environment for connections between virtual representations of real-world sensors and actuators. *FlowMatic* [60] is a similar work based on functional reactive programming that reduces complexity by offering abstractions. *RealityFlow* offers a multi-user dataflow VPL that can be used on desktop, mobile, and VR platforms [37].

The modulation mapping technique that we present realizes a dataflow VPL approach to create reactive behavior. Compared to previous work, we aim to further simplify the authoring of reactive behavior for a wide range of users. To this end, we propose to prepare complex behavior offline by programmers and to expose it to the authoring environment as high-level object properties, following a similar motivation as in [3]. However, we do not model behavior as discrete events and actions specified as logical conditions, but model the immediate effects of continuously varying inputs on the affected target properties. By offering a special type of mappings, we enable modeling of hybrid continuous-discrete signal flows within a unified interface design concept (see Section 3.2).

2.2 Reference Frames for 3D User Interfaces

The layout of 3D user interface (UI) elements in the VE can be classified based on the reference frame relative to which they are positioned [20, 24]. This positioning is crucial for determining accessibility and occlusion [10]. In the following, we give an overview of how previous work has classified and evaluated reference frames.

Feiner et al. [23] pioneered 2D window placement in augmented reality (AR) with world-, display-, and *surround-fixed* windows. World-fixed windows are fixed to locations or objects, display-fixed ones to head-worn displays, and surround-fixed ones are located at a fixed position in the user's surroundings. Bowman et al. [10] adapted and extended the terminology for graphical menu placement to world-referenced, object-referenced, head-referenced, body-referenced, and device-referenced, the latter referring to a physical display surface, such as a workbench or handheld AR system. In line with Feiner et al. [23], we adopt the *surround-* terminology for UIs positioned in the surroundings of the user, i.e., the physical tracking space. However, we use the term *-referenced* instead of *-fixed* in line with Bowman et al. [10] for an interface that can be repositioned freely within this reference frame. When the user walks in the tracked area, a *surround-referenced* interface will appear static, similar to a world-referenced layout. However,

when a travel technique such as steering [10] or teleportation [11] is used, it follows the movement of the user. Similar to Bowman et al. [10] we use *object-referenced* for UIs attached to a scene object.

The influence of different reference frames for interface layouts has previously been studied. Billinghurst et al. [9] compared head- and body-fixed information displays on a wearable computer, demonstrating that body-fixed positioning enabled faster information retrieval. Polys et al. [41] further investigated the impact of information layout (object- and head-referenced), screen size, and field-of-view on user performance in VEs. They conclude that the consistent visibility of the head-referenced layout significantly improved user performance and satisfaction. Similarly, Bernatchez et al. [8] found that 3D floating menus are best used with reference frames following the user's position and facing the user. Das and Borst [15] further evaluated contextual (object-referenced) and world-referenced menus for projection-based VR systems, and reported that contextual menus were faster. Ens et al. [20] studied world- and body-referenced multiple display layouts for multi-tasking on head-worn AR displays, and reported that body-referenced layouts can lead to higher selection errors due to reaching motion. Lediaeva and LaViola [34] further compared body (arm, hand, and waist) to world-referenced menus in VR, finding that participants preferred world-referenced menu placement.

Immersive authoring of reactive behavior by dataflow VPL requires a graphical representation of connections between nodes. This introduces additional information and interaction in 3D space compared to previously explored spatial menus. A comparison of reference frames for immersive dataflow VPL interfaces has not been addressed in the literature. Such an analysis is crucial for understanding the impact of different reference frames in the dataflow VPL paradigm. Considering previous work, we choose a *surround-referenced* layout that maintains consistent visibility by traveling with the user, which may enhance accessibility and performance [8, 41]. We compare it to an *object-referenced* layout that positions interface elements in the VE, which may improve contextual and spatial awareness [15] and user experience [18, 34]. By systematically investigating the efficiency and overall user experience of these layouts in the context of creating reactive behavior, our research contributes insights into the design and optimization of immersive dataflow VPL interfaces for immersive authoring.

3 TECHNIQUE DESIGN

Our goal is to design an immersive authoring technique for creating reactive behavior that is easily accessible to a wide range of users. To achieve this, we drew inspiration from dataflow VPL, which represent programs as networks of nodes and connections that naturally depict data and control flow. This visual representation aligns well with how humans conceptualize processes [54], making it easier to understand the transformation of data, and is thus considered intuitive and end-user friendly [59]. To devise our technique, we developed the following design considerations:

D1 Structural Simplicity: Dataflow VPL programs can become structurally complex with numerous tangled connections between nodes, making interpretation difficult [49]. Therefore, our technique should enable users to define reactive behavior without the structural complexity of VPL.

D2 Effortless Customization: Users can have difficulty to understand low-level primitives and compose them into high-level behavior [40]. To make the technique approachable, customizing the influence on a target property should be possible without using explicit mathematical operators.

D3 Unified Mechanism: VPL that offer multiple concepts to choose from can negatively impact the ability to solve problems easily [7]. Thus, for conceptual simplicity, the same mechanism used to customize mappings should enable modeling of conditional and stateful behaviors.

D4 Spatial Reference Frame: As the nodes are closely related to scene objects, it is crucial to investigate their placement relative to the objects or the user's position [15, 41]. The positioning of the interface elements should provide a good user experience in terms of efficiency, cognitive load, and user-friendliness when creating reactive behavior while being immersed.

Based on these considerations, the following sections explain our conceptual decisions and UI designs.

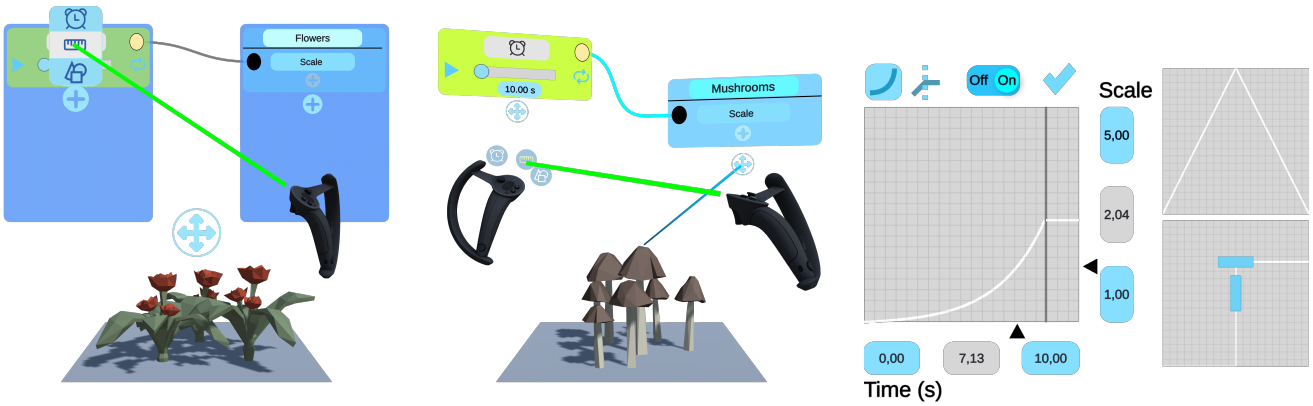
3.1 Modulation Mapping

To achieve a low structural complexity that is more approachable to novice users (**D1**), we propose to restrict the mapping between *source* and *target* properties to direct point-to-point connections. To customize the influence (**D2**), a mapping function can be applied to each connection to create non-linear effects, discontinuous transitions, or to model conditional and stateful behavior (**D3**) (see Section 3.2). In our workflow, non-trivial behavioral logic is prepared by a programmer and its parameters are exposed to the content author as object properties. The method thus shifts the level of abstraction towards the authoring of scenarios based on ready-to-use template functionality, in contrast to programming by the composition of basic building blocks. Novice users are not required to learn a modeling language but benefit from the high level of abstraction of the provided templates.

We propose three types of *source* properties: *Object Property*, *Time*, and *Relation*. To achieve conceptual simplicity while being easily extensible, we choose to model most inputs as object properties. To create temporal and context-aware behaviors, two additional types of sources are necessary: *Time* sources expose additional playback controls, and *Relation* sources can reference multiple objects.

- ***Time***: Progression of time as a continuous input enables modeling of temporal dynamics, e.g., changing object properties over time or controlling previously recorded animations.
- ***Relation***: Spatial relations between scene objects, such as the distance or angle, can express contextually responsive behaviors. A virtual agent might, e.g., change its mood (anxious, angry, or scared) whenever the user looks directly at it.
- ***Object Property***: Any other scalar-valued input that can influence the scenario. This can, e.g., include universal properties of objects in the scene (scale, movement speed), attributes specific to a class of object (rotation speed of a fan, battery charging status), scene properties (music loudness, sunlight intensity), or external input sources (user interaction, network input).

As an example of external input sources, we enable user input by introducing an avatar as a proxy object, carrying the left- and right-hand controllers. It can be placed into the scene by dragging



(a) *Surround-referenced*: Object properties are located on a panel with a two-column interface to create connections. All menu elements are added by choosing from a drop-down list.

(b) *Object-referenced*: Object properties are attached directly to scene objects and connections are created between them. Other menu elements are instantiated via buttons on the left controller.

(c) The mapping function interface (left) is used to choose a template function, adjust the source and target ranges, enable or disable the mapping, and toggle the *One-shot* option. Example template functions (right): *Triangle, Step.*

Figure 2: Our *surround-referenced* and *object-referenced* interface designs to define modulation mappings. All interface elements can be repositioned freely using a handle (blue crossed arrows) relative to their respective frame of reference.

a body-referenced miniature avatar attached to the user's chest. This way, the user's position, controller positions, and additional controller inputs (buttons, joysticks) can be used as input sources like any other object in the scene. It further enables to observe effects from a third-person perspective, e.g., by moving the avatar closer or farther away from an affected object (see Fig.4-bottom).

Target properties can similarly be defined as any scalar-valued parameter that affects an individual object's behavior (virtual agent mood, crowd simulation parameter) or visual appearance (scale, transparency), or global scene properties (sky color tint, rain intensity). Each target property is influenced by a single source property. To combine multiple inputs, an operator (add, multiply, average) could have been utilized. This would, however, introduce another layer of structural complexity. To keep the method easily approachable by novices, we chose to offer only a single input per target.

3.1.1 User Interface Designs. To choose a suitable spatial layout for our user interface elements (**D4**), we reviewed previous work and found the impact of reference frames on immersive authoring through visual programming not addressed in the literature. Therefore, we developed two interface designs for creating modulation mappings using a *surround-referenced* and an *object-referenced* layout that we evaluate in a comparative user study (see Section 4).

Both UIs utilize the same menu elements to represent source and target properties. We chose adapted 2D menus as a familiar interface to improve the overall user experience [10]. The *Time* element contains a draggable input field for duration, a play button, and a looping toggle. The *Relation* element has two input fields for selecting objects (see Fig. 1). The *Object Property* element contains an input field for object selection and a button (+) to add properties. Interaction is realized by a ray-based selection technique. Object selection can be performed by pointing (long range), grabbing (close range), or via a drop-down menu. The latter enables to select objects without graphical representation (e.g. scene lighting or background music). To create mappings, the user drags connecting lines between the ports of the respective properties.

Surround-Referenced Layout: Our motivation for the *surround-referenced* layout was to ensure that users can interact with minimal effort [41] and no travel is necessary to draw connections. We designed an interface on a panel comprising two distinct columns: the left for source properties and the right for target properties (see Fig. 2a). The objective of the two-column view was to facilitate point-to-point connections that can be traced with minimal cognitive effort. Since all modulation mappings are co-located on a single panel, the interface needs to be kept within the user's reach. Therefore, the interface is anchored relative to the user's surrounding (i.e. the tracking space), such that it follows the user when traveling.

To create modulation mappings, the desired source and target properties have to be added to the respective column. Pressing + on the source panel opens a drop-down list in which *Time*, *Relation*, or *Object Property* can be selected. After selection, a corresponding menu element is created on the panel. Pressing + on the target panel creates a menu element and initiates object selection.

Object-Referenced Layout: Our motivation for the *object-referenced* layout, used in previous works [13, 18, 47], was to maintain the spatial relationship between the programming elements and the scene objects. Interacting spatially by reaching into the scene to draw connections was found intuitive in similar use cases and could improve the overall user experience [18].

Each object has its own *Object Property* menu element that can be opened by selecting the object. The element appears above the object and can be repositioned by dragging its handle. Pressing + opens a drop-down list for selecting source and target properties, visually distinguished by source properties having a port on the right, and target properties on the left. For *Time*, *Relation*, and objects without graphical representation, we designed a hand-held

menu that contains 3D buttons with corresponding icons. Selecting an icon instantiates the corresponding menu element (see Fig. 2b).

3.2 Mapping Functions

A mapping function can be applied to customize how the source property affects the target property. To facilitate this, we designed a 2D graph interface (see Fig. 2c-left). Hovering over a connecting line of a mapping displays a graph icon. Selecting this icon opens the graph interface in front of the mapping panel for the *surround-referenced* or the target object's menu for the *object-referenced* case.

We provide template functions with visual representations for users to customize modulation mappings without overwhelming them, especially those with limited mathematical knowledge (**D2**). Our templates include *Linear Increase* and *Decrease* for proportional changes, *Exponential Increase* and *Decrease* to create non-linear dynamic behavior, and *Triangle* and *Inverted Triangle* for cyclical patterns like oscillations or repeating effects. To realize discrete conditions and events (**D3**), we provide a *Step* and its inverted function. The *Step* can be used to model conditional behavior, where the output value switches depending on whether a threshold is exceeded. We offer custom draggable handles for direct control of the threshold and target value (see Fig. 2c-right).

All functions can be toggled to act as a *One-shot* function, disabling the mapping after the threshold is reached to create stateful behavior. For example, a group of plants could grow when the water level rises due to rain. Using a *One-shot* mapping, the plants will retain their size after the rain stops. The user can design and test the effects of the *One-shot* function while the graph view is open; once closed, the mapping triggers once and deactivates when the threshold is crossed, until reactivated by the on/off switch.

The interface includes a graph view of the mapping function and draggable input fields for the source value range $[s_{min}, s_{max}]$ and target value range $[t_{min}, t_{max}]$, depicted along the X- and Y-axis, respectively. To compute the target value t, the source property s is normalized to the interval $[0, 1]$, transformed by the mapping function $f : [0, 1] \rightarrow [0, 1]$, and mapped to the target value range.

$$t(s) = \begin{cases} t_{min} & \text{if s} \leq s_{min} \\ t_{max} & \text{if s} \geq s_{max} \\ t_{min} + f(\frac{s - s_{min}}{s_{max} - s_{min}}) * (t_{max} - t_{min}) & \text{else} \end{cases} \quad (1)$$

The user can observe the current value via black arrowheads at both axes, as well as numerical feedback in gray text boxes. These are continuously updated to reflect a changing source value. This direct feedback enables users to understand how the source value is translated to the target value. Thereby, the user can test the effects of the mapping function on the scene while designing it in VR.

4 EMPIRICAL EVALUATION

To provide an immersive authoring technique by visual programming that offers a good user experience, it is important to use a suitable spatial layout for its programming elements (**D4**). Since previous work did not investigate the role of reference frames in this context, our main research investigates the benefits and limitations of two interface layouts regarding the applicability of modulation mapping. Therefore, we designed an empirical user study to evaluate the *surround-referenced* (**SR**) and *object-referenced* (**OR**) layouts in the context of immersive authoring of reactive behavior. We determine the applicability of our technique by the efficiency, cognitive load, and user-friendliness. Our study design was guided by the following hypotheses:

Regarding efficiency:

- The mean task completion time will be lower for **SR** (**H1a**).
- The differences in task completion times between **SR** and **OR** depend on the task complexity (**H1b**).

Regarding cognitive load:

- The mean score for task load will be lower for **SR** (**H2**).

We operationalize user-friendlines based on the measures stated in the following hypotheses:

- We expect the mean reported clutter to be higher for **OR** (**H3**), because user interface elements attached to scene objects can obstruct the user's view [33].
- To gain insight into how users perceive the layouts, we formulate the following undirected hypotheses. The mean scores for the six User Experience Questionnaire (UEQ) scales (**H4**), ease of use (**H5**), ease of learning (**H6**), and preference (**H7**) will be different based on the interface layout.

4.1 Study Design and Tasks

We devised a within-subject design with two subsequent parts to investigate our hypotheses regarding the *object-referenced* and *surround-referenced* layouts as two levels of one factor. In the first part of the study (see Section 4.1.1), participants were asked to create modulation mappings in abstract scenarios that contained only primitive objects with simple colors. The object properties were denoted with letters from A to F (see Fig. 3). Our motivation was to evaluate task completion times (**H1**), task load (**H2**), and perceived clutter (**H3**) in a controlled setting without additional biases induced by the semantics and visual complexity of real-world scenarios. We aimed to get a clearer view of the interface design's performance without too much influence by the conducted task.

In the second part, participants were asked to create modulation mappings within a realistic scenario (Section 4.1.2) without time constraints. This part of the study aimed to resemble real-world use cases more closely, so that we could gain insight into both interfaces' practical applicability and gather empirical data with a higher level of external validity compared to the first part of the study. Without time pressure, participants can thoroughly evaluate the interfaces to provide feedback on their experience (**H4**), ease of use (**H5**), ease of learning (**H6**), and overall preference (**H7**).

4.1.1 Abstract Scenarios. To test **H1**, we created a second factor that varies the complexity of the task. When evaluating and assessing different testing scenarios, we found that the object arrangement has a major influence on the task complexity. Therefore, while we use abstract objects with a minimal appearance to avoid distractions, we choose their positioning to resemble typical scene arrangements that can be encountered in immersive authoring. Individual objects may, for example, be laid out next to each other and easily reachable, be partially obstructed, or have different sizes. Objects may also be located at a distance from each other, such that users have to change their field of view to understand their spatial relation. We considered previous work [50] and went through an iterative process in which we included VR experts from our lab

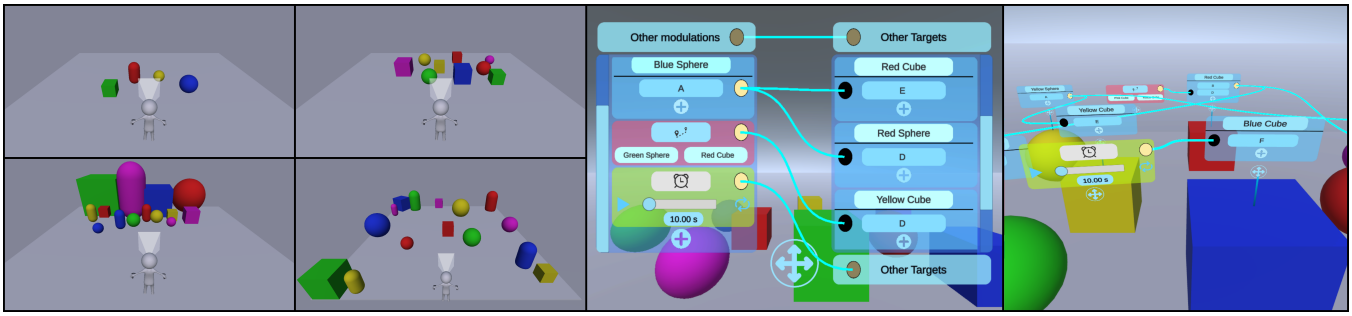


Figure 3: Overview of part one of the study, showing abstract scenes (left) in row-wise order: *Simple, PartlyOccluded, VaryingScale, Distant*. Exemplary mappings in the *PartlyOccluded* scene using the SR (center) and OR (right) interface layouts.

Task 1 Modulate the scale of the two groups of flowers by the background music loudness.

Task 2 Lower the bridge based on the distance of the avatar to the gate.

Task 3 Change the mood of the sky and the rain intensity based on the distance of the avatar to the totem.

Task 4 Open the chest as the avatar approaches with the right controller.

Task 5 The treasure inside of the chest should fade in and out continuously.

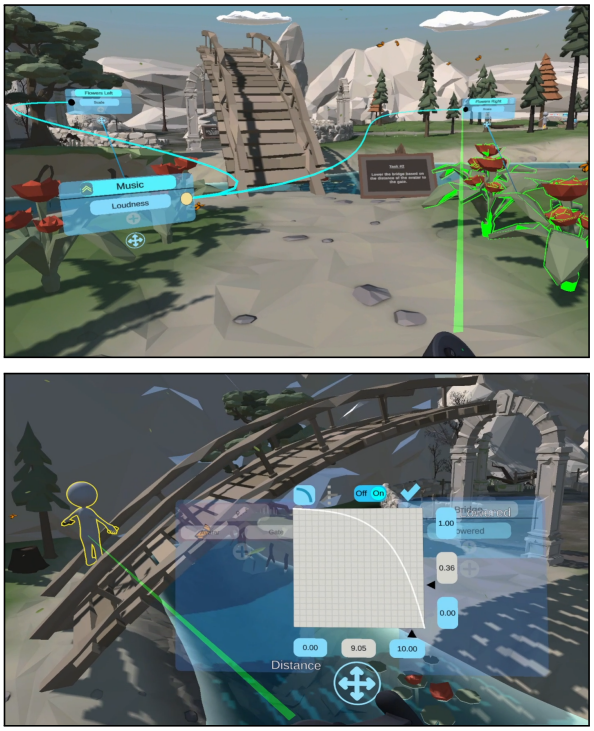


Figure 4: Realistic scenarios in the user study: Task descriptions (top); example view of *Task 1*, OR (center) and *Task 2*, SR with mapping function interface (bottom).

with the goal of producing a set of scenes with representative object arrangements. Eventually, we derived four scenes that we denote as *Simple*, *Partly Occluded*, *Varying Scale*, and *Distant* (see Fig. 3).

The specific objects involved in each task and the corresponding number of menu elements were chosen to reflect the quality of each scene. In the *Simple* scene, 4 objects are laid out next to each other. To complete the task, 6 menu elements and 4 connections have to be created. For the *Partly Occluded* scene, we arranged 9 objects such that the menu elements partially occlude each other, and users have to create 9 menu elements and 6 connections. The *Varying Scale* scene contains 15 objects, where small ones are located in front of bigger ones to avoid the selection of fully occluded objects. The task involves creating 9 menu elements and 5 connections between objects of different sizes. Regarding *Distant*, there are 15 objects and participants have to create 8 menu elements and 10 connections. The mappings were chosen such that connections need to be made between far-apart objects, requiring participants to travel and repeatedly change their field of view.

We used hand-based steering rather than teleportation to enable users to draw connections while traveling. In all scenes, objects are placed within an area of 10 by 10 meters. Instructions are listed on a panel attached to the left controller, which might read: $GreenCube(A) \rightarrow RedCube(D)$. The specific instructions that were used in the study and a Unity package that contains the abstract scenarios can be found at https://zenodo.org/records/10627368.

In this first part of the study, the mapping function interface (see Fig 2c) was disabled. Since the interface is identical for both layouts, it would introduce unnecessary variance to the measured effects we investigate by **H1** to **H3**. Regarding **H2** and **H3**, we look at the main effect caused by the layout, resulting in a one-factorial (layout) within-subject design. Participants ranked questionnaires regarding task load and perceived visual clutter after completing all four tasks for each respective interface layout.

4.1.2 Realistic Scenarios. We created a virtual scenario for participants to perform five tasks (see Fig. 4) that consisted of adding reactive behaviors to different scene objects. The tasks were designed to cover the key functionalities of our presented technique. This includes the utilization of object properties as sources and targets, distance relations between objects, time-modulated behaviors, and custom mapping functions. All tasks were performed in the same order to construct a coherent narrative. To validate

our hypotheses **H4** to **H7**, we chose a one-factorial within-subject design, where participants ranked subjective questionnaires after completing all five tasks in each condition.

4.2 Apparatus

The study took place in our lab, with participants seated on a rotatable chair. We used a Valve Index HMD with its controllers, tracked by four Lighthouse 2.0 base stations. The experimental platform was developed in Unity 2021.3.5f1, running on a Windows 10 with a 3.7GHz Intel Core i9-10900X CPU and an NVIDIA RTX 3090 GPU.

4.3 Procedure

Upon arrival, participants signed a written informed consent form and completed a pre-study questionnaire about demographics and experience in VR, programming, and visual programming.

In part one of the study, participants went through a familiarization phase before executing the tasks with each layout. They received instructions from a panel and watched demonstration videos in VR. They practiced creating modulation mappings for up to 15 minutes. Participants then performed the tasks in the four scenes (see Section 4.1.1). Once the first menu element had been instantiated, completion times were recorded and the participant could start traveling. The order of the layouts and scenes was counterbalanced using a Latin Square. After completing all four scenes for the given layout, participants answered NASA-TLX [25] and a 7-point Likert scale custom questionnaire to rate visual clutter.

In part two of the study, participants first entered another familiarization phase to try out the mapping function graph and learn to place the avatar into the scene. Afterward, they were asked to create reactive behaviors in a realistic scenario (see Section 4.1.2). After completing all tasks for a given layout, participants answered UEQ [32], a 7-point Likert scale questionnaire on preference, ease of learning and use, and freely commented on the recently used layout. The layouts were assigned in the same order as in part one.

Lastly, they provided free comments on the modulation mapping technique and answered a 7-point Likert scale questionnaire regarding its effectiveness and ease of learning the mapping function graph interface. The entire procedure took approximately 90 min.

4.4 Participants

We recruited 36 participants on the university campus through various communication channels, but we had to exclude two due to technical issues. The remaining 34 participants (7 female, 26 male, 1 non-binary) were aged between 20 and 41 (M = 27 years old, SD = 4.85), all right-handed with normal or corrected-to-normal vision. Their experience levels in VR applications ranged from 2 novices, 6 beginners, 9 advanced, to 17 experts, in programming from 1, 1, 7, to 25, and in visual programming from 8, 14, 9, to 3.

5 RESULTS

To determine statistically significant differences between the layouts for varying task complexity, we analyzed the measurements using 2x4 factorial repeated-measures ANOVA. We further computed a two-sided paired-sample t-test, concerning variables that were measured only once per condition without a second factor. When sphericity assumptions were not met, we carried out Mauchly's tests and applied Greenhouse-Geisser correction. Additionally, when we observed a statistically significant overall effect, we conducted Bonferroni-corrected post-hoc tests for pairwise comparisons.

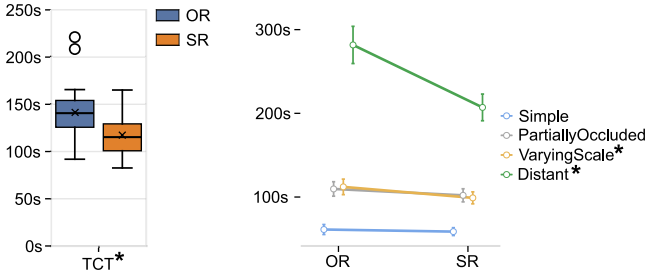


Figure 5: Mean task completion time (TCT): Box plots of the main effect (left). Marginal means per condition show the interaction between layout and task (right), whiskers indicate the 95% confidence interval.

Effect sizes for ANOVA were assessed with η_p^2 (thresholds: .01, .06, and .14 for small, medium, and large effects), while t-test effect sizes were computed using Cohen's d (thresholds: .2, .5, and .8 for small, medium, and large effects) [14]. For all statistical tests, we assume a significance level of $p < .05$, marked as (*) in the figures. Box plots show the median (-), mean (x) and interquartile range (IQR), with whiskers indicating $IQR * 1.5$.

5.1 Objective Measures

Regarding **task completion time**, we observed a significant main effect of layout ($F(1, 33) = 56, p < .001, \eta_p^2 = .629$), supporting **H1a**. We further observed a significant interaction effect between layout and scene ($F(1.47, 48.44) = 40.1, p < .001, \eta_p^2 = .549$). Post-hoc tests revealed two significant differences between the layouts in *Varying Scale* ($p = .039, d = .598$) and *Distant* ($p < .001, d = 1.296$), supporting **H1b**. To understand the underlying factors, we recorded the time spent interacting with menus, dragging connecting lines, traveling, and the amount of head rotation per task (See Fig. 5).

Menu **interaction time** includes pressing buttons, scrolling, and repositioning menus. We observed no significant main effect of layout ($F(1, 33) = 1.88, p = .180, \eta_p^2 = .054$).

Concerning **drag time**, we observed a significant main effect of layout ($F(1, 33) = 149.8, p < .001, \eta_p^2 = .819$), and a significant interaction effect between layout and scene ($F(1.27, 41.93) = 79.2, p < .001, \eta_p^2 = .706$). Post-hoc tests revealed significant differences between layouts in all scenes ($p_{all} < .001, d_{all} \geq 1.26$).

For **travel time**, we observed a significant main effect of layout ($F(1, 33) = 84.6, p < .001, \eta_p^2 = .719$), and a significant interaction effect between layout and scene ($F(1.20, 39.59) = 68.2, p < .001, \eta_p^2 = .674$). Post-hoc tests revealed significant differences between layouts in all scenes except *Simple* ($p_{all\setminus Simple} < .001, d_{all\setminus Simple} \geq .903$).

Regarding **head rotation**, we found a significant main effect of layout ($F(1, 33) = 60.9, p < .001, \eta_p^2 = .648$), and a significant interaction effect between layout and scene ($F(1.46, 48.07) = 104.8, p < .001, \eta_p^2 = .760$). Post-hoc tests showed significant differences between layouts only in *Distant* ($p_{Distant} < .001, d_{Distant} = 1.83$).

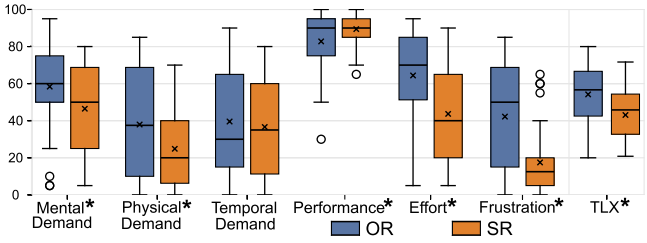


Figure 6: Box plots of NASA-TLX sub-scores and total scores.

	Statistics			Mean (SD)	
UEQ Scales	*t*	*p*	*d*	**OR**	**SR**
Attractiveness	2.50	0.017	0.43	1.56 (1.08)	1.99 (0.56)
Perspicuity	2.58	0.014	0.44	1.74 (0.82)	2.14 (0.71)
Efficiency	4.73	< 0.001	0.81	0.87 (1.30)	2.00 (0.68)
Dependability	4.15	< 0.001	0.71	1.41 (0.85)	1.99 (0.68)
Stimulation	1.05	0.302	0.18	1.68 (1.02)	1.85 (0.67)
Novelty	-2.66	0.012	-0.46	1.46 (0.83)	0.94 (1.06)

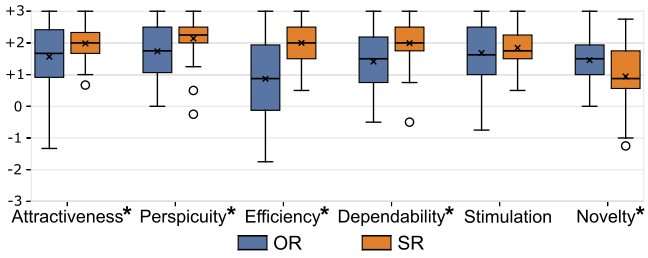


Figure 7: UEQ sub-scales: Paired sample t-test statistics, mean (standard deviation), and distributions for each layout.

5.2 Subjective Measures

We conducted a two-sided paired-sample t-test for the following variables since they were measured only once per layout.

The mean **task load** is significantly lower for SR than OR ($T(33) = 5.19, p < .001, d = .891$), supporting **H2**. To investigate the factors determining task load, we examine the NASA TLX sub-scales (see Fig. 6) and except Temporal Demand ($T(33) = .96, p = .343$), we observed significant differences. SR is significantly lower in Mental Demand ($T(33) = 3.94, p < .001, d = .68$), Physical Demand ($T(33) = 2.51, p = .017, d = .43$), Effort ($T(33) = 4.39, p < .001, d = .75$) and Frustration ($T(33) = 5.52, p < .001, d = .947$) and higher in Performance ($T(33) = 3.17, p = .003, d = .54$) compared to OR.

Regarding **user experience**, significant differences were observed in five out of six scales of the UEQ, except Stimulation, which partially supports **H4**. SR significantly outperformed OR in Attractiveness, Perspicuity, Efficiency, and Dependability, while OR significantly outperformed SR in Novelty. See Fig. 7 for the distribution, statistical test results, and descriptive statistics.

Concerning the **subjective questionnaires**, significant differences were found in **Visual Clutter** ($T(33) = -6.80, p < .001, d = -1.165$), **Ease of Learning** ($T(33) = 2.04, p = .049, d = .35$), **Ease of Use** ($T(33) = 2.94, p = .006, d = .504$), and **Preference** ($T(33) = 2.32, p = .027, d = .398$), supporting **H3, H5, H6, H7**. Participants also rated the **Effectiveness** of the modulation mapping technique ($M = 6.38, SD = 1.10$) and the **Ease of Learning** of the mapping function ($M = 6.32, SD = .97$). See Fig. 8 for the results.

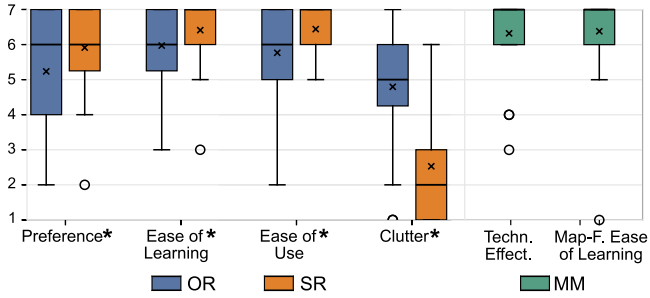


Figure 8: 7-point Likert questionnaire, with items ranging from very strongly disagree(1) to very strongly agree(7). Comparison of the layouts (left), and evaluation of the modulation mapping (MM) technique (right) regarding its effectiveness and the ease of learning of the mapping function interface.

5.3 Qualitative Results

We analyzed the comments on layouts and modulation mapping using thematic analysis [12] and present the findings in this section.

Regarding *object-referenced*, 12 participants found it **intuitive** and **engaging** to interact directly with objects in the scene to define behavior. P9 stated, *"It is really intuitive and feels immersive since I was able to place the mapping behavior directly to the objects"*. Five participants (P12, P20, P25, P33, P34) stated that selecting properties and connecting ports from a **distance can be challenging**, especially with objects far apart. The layout was considered valuable for tasks like **visual debugging** (P9, P12, P15, P26), however, some participants also noted potential issues with **clutter** (P15, P24, P26, P33, P35) when dealing with a large number of connections. Moreover, P7 suggested automated UI scale adjustment and layouting.

Regarding *surround-referenced*, eight participants (P2, P9, P15, P17, P20, P26, P31, P33) appreciated the **efficiency** and **ability to access** the mappings **without requiring travel**. However, seven participants (P1, P20, P21, P24, P26, P30, P33) stated that with numerous mappings, the interface can become **cluttered**, making it harder to navigate and **maintain an overview**. P31 stated, *"The only downside is having to scroll more for objects at the top and I can imagine that in more complex scenarios the overview can get lost when you can't see the mappings at once"*. P1 and P19 suggested **grouping** options for better list organization.

Regarding the *modulation mapping technique*, 15 participants found authoring reactive behaviors by creating direct connections **easy to use** and **intuitive**. Some participants appreciated the real-time feedback and customization of behaviors via predefined mapping functions. P11 and P35 found the technique beneficial to expert programmers: VR expert P11 stated, *"not only novices but also experienced programmers can benefit from the technique for quick testing of the behaviors without diving into the code or taking the HMD off"*.

Four participants (P19, P25, P32, P27) proposed **combining elements of both layouts**. P26 stated, *"I liked using surround-referenced, however, I was missing the connection to the virtual environment from the mappings"*. Some participants suggested drawing indicator lines from the SR panel to the respective scene objects upon hover. In addition, two VR experts (P27, P31) suggested to

enable instantiating multiple mapping panels that can be placed next to scene objects and summoned when needed.

6 DISCUSSION

We found **SR** to outperform **OR** in terms of faster task completion. The results further revealed an interaction effect between layout and scene. While we observed significant differences in task completion time for *Varying Scale* and *Distant* scenes, there were no significant differences for *Simple* and *Partly Occluded*. Based on our secondary results, a possible explanation could be that the **OR** layout requires additional interactions (travel, head movement) or poses precision challenges when drawing connecting lines (drag time), which is more evident when objects are at a distance or have varying scales.

Regarding task load, participants found creating modulation mappings using **OR** significantly more mentally and physically demanding, and frustrating compared to using **SR**. We speculate that the increased head movement and travel for drawing connections between distant objects led to split attention, increasing the mental load compared to the **SR** condition, where interactions could be performed co-located on the SR panel [53]. Participants expressed frustration when ray-casting precision issues caused failed connections, requiring more physical effort. With **SR**, the interface elements are often reachable in close proximity, thereby the mapping creation may be achieved with lower effort.

Placement of menus can clutter a VE and occlude the user's view [10]. Since in the **OR** layout, menus are placed in the environment, we hypothesized that **OR** clutters the scene more than **SR**. The results support our hypothesis, showing that **OR** significantly cluttered the view more. This suggests that **SR** provided a more streamlined interaction, contributing to better user experience and task performance. These results are in line with previous work [4, 43, 44] and our own findings regarding **H1a** and **H4**.

Concerning user experience, **SR** was rated higher than **OR** for attractiveness, perspicuity, efficiency, and dependability. Furthermore, **SR** was found easier to learn, easier to use, and preferable over **OR**. We believe that **SR**, by placing interface elements within arm's reach, provided better overview and easier access, thereby reducing mental and physical load. This, in turn, improved the overall user experience. Our findings are in line with previous work [27], showing that interfaces with lower cognitive load lead to higher user satisfaction. When users find an interface easy to learn (**H5**), easy to use (**H6**), and not mentally demanding (**H2**), they tend to have a more positive overall perception of the interface (**H4**, **H7**).

Given the high ratings for our technique's effectiveness, ease of learning the mapping function, and overall positive feedback, modulation mapping shows potential for enabling users to easily and effectively create reactive behavior in immersive environments. Our technique appears to be a promising and valuable addition to existing immersive authoring tools.

6.1 Design Implications

Based on our empirical results, we propose initial design implications regarding the choice of reference frame for user interfaces in immersive authoring by visual programming.

We suggest to use *surround-referenced* interface layouts where efficiency and streamlined workflows are a priority, especially for scenes with a large number of objects. However, consider to organize the property lists clearly by providing tools to navigate and maintain an overview of connections. This type of interface would be most suitable for behaviors that do not strongly rely on spatial relationships between objects. Additional visual feedback, such as drawing indicator lines between mappings and the respective scene objects upon hover, may further support the authoring process.

The *object-referenced* interface layout was found intuitive, engaging, and valuable for visual debugging. However, it can become cluttered with numerous objects. Therefore, use it for scenes with fewer objects and behaviors that strongly rely on their spatial configuration. Consider (semi-)automatic layouting [45], grouping, and hiding of objects to minimize clutter [18]. These could be guided, e.g., by estimating the cognitive load during task execution [36]. Consider selection techniques that address the ray-casting precision issues [31] to select properties and connect nodes at a distance.

Generally, we believe that a *hybrid* approach that combines elements of both *surround-* and *object-referenced* layouts may leverage their respective strengths and mitigate their weaknesses. We see interesting opportunities for future research in immersive reactive behavior authoring that explores how both approaches can be linked in a synergistic way.

7 LIMITATIONS AND FUTURE WORK

Our study design considers task complexity as a factor influencing efficiency. To vary task complexity, we altered the spatial arrangement of the objects alongside the number of required interactions. We found both related factors relevant to achieve a representative and diverse set of tasks, yet we plan to investigate them separately in future trials. Further, we only explored static horizontal arrangements of objects. Future research could examine authoring reactive behavior in scenes with moving, or vertically arranged objects. In this initial design, we used ray-casting for selection and hand-directed steering for travel. Since interaction techniques impact task performance and user experience, our future research will evaluate different selection and travel techniques in this context.

While our participants completed all tasks successfully and provided overall positive feedback on the modulation mapping technique, we plan to conduct further user studies regarding its effectiveness in different authoring scenarios and application domains.

8 CONCLUSION

We presented modulation mapping, a simplified approach for immersive authoring of reactive behavior by dataflow visual programming. In a comparative user study, we compared a *surround-* to an *object-referenced* interface layout regarding task efficiency and user experience. From the results, we derived an initial set of interface design implications for future research and development of similar immersive authoring techniques. Participants found modulation mapping an intuitive and effective way of authoring reactive behavior while immersed. At the same time, experts in VR and programming found the technique a valuable addition to a developer's workflow for authoring immersive environments.

REFERENCES

[1] Adobe. 2020. Top 3D sculpting tools for virtual reality authoring. https://www.adobe.com/products/medium.html. (Accessed on 06/03/2024).

[2] Rahul Arora, Rubaiat Habib Kazi, Danny M Kaufman, Wilmot Li, and Karan Singh. 2019. Magicalhands: Mid-air hand gestures for animating in vr. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology*. 463–477.

[3] Valentino Artizzu, Gianmarco Cherchi, Davide Fara, Vittoria Frau, Riccardo Macis, Luca Pitzalis, Alessandro Tola, Ivan Blecic, and Lucio Davide Spano. 2022. Defining Configurable Virtual Reality Templates for End Users. *Proceedings of the ACM on Human-Computer Interaction* 6, EICS (2022), 1–35.

[4] Felipe Bacim, Eric D Ragan, Siroberto Scerbo, Nicholas F Polys, Mehdi Setareh, and Brett D Jones. 2013. The effects of display fidelity, visual complexity, and task scope on spatial understanding of 3D graphs.. In *Graphics Interface*, Vol. 2. 25–32.

[5] Camille Barot, Kevin Carpentier, Marie Collet, Andrea Cuella-Martin, Vincent Lanquepin, Mathilde Muller, Esteban Pasquier, Loic Picavet, Arthur Van Ceulen, and Kevin Wagrez. 2013. The wonderland builder: Using storytelling to guide dream-like interaction. In *2013 IEEE Symposium on 3D User Interfaces (3DUI)*. IEEE, 201–202.

[6] Rüdiger Beimler, Gerd Bruder, and Frank Steinicke. 2013. Smurvebox: A smart multi-user real-time virtual environment for generating character animations. In *Proceedings of the Virtual Reality International Conference: Laval Virtual*. 1–7.

[7] Brigham Bell, John Rieman, and Clayton Lewis. 1991. Usability testing of a graphical programming system: things we missed in a programming walkthrough. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 7–12.

[8] Marc Bernatchez and Jean-Marc Robert. 2007. A study on the impact of spatial frames of reference on human performance in virtual reality user interfaces. In *2007 IEEE International Conference on Systems, Man and Cybernetics*. IEEE, 2600–2605.

[9] Mark Billinghurst, Jerry Bowskill, Nick Dyer, and Jason Morphett. 1998. An evaluation of wearable information spaces. In *Proceedings. IEEE 1998 Virtual Reality Annual International Symposium (Cat. No. 98CB36180)*. IEEE, 20–27.

[10] Doug Bowman, Ernst Kruijff, Joseph J LaViola Jr, and Ivan P Poupyrev. 2004. *3D User interfaces: theory and practice, CourseSmart eTextbook*. Addison-Wesley.

[11] Evren Bozgeyikli, Andrew Raij, Srinivas Katkoori, and Rajiv Dubey. 2016. Point & teleport locomotion technique for virtual reality. In *Proceedings of the 2016 annual symposium on computer-human interaction in play*. 205–216.

[12] Virginia Braun and Victoria Clarke. 2012. *Thematic analysis*. American Psychological Association.

[13] Edwige Chauvergne, Martin Hachet, and Arnaud Prouzeau. 2023. Authoring Interactive and Immersive Experiences Using Programming by Demonstration. In *Proceedings of the 34th Conference on l'Interaction Humain-Machine*. 1–13.

[14] Jacob Cohen. 2013. *Statistical power analysis for the behavioral sciences*. Routledge.

[15] Kaushik Das and Christoph W Borst. 2010. An evaluation of menu properties and pointing techniques in a projection-based VR environment. In *2010 IEEE Symposium on 3D User Interfaces (3DUI)*. IEEE, 47–50.

[16] Josen Daniel O De Leon, Romelio P Tavas, Rodolfo A Aranzanso, and Rowel O Atienza. 2016. Genesys: A Virtual Reality scene builder. In *2016 IEEE Region 10 Conference (TENCON)*. IEEE, 3708–3711.

[17] John Dudley, Hrvoje Benko, Daniel Wigdor, and Per Ola Kristensson. 2019. Performance envelopes of virtual keyboard text input strategies in virtual reality. In *2019 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. IEEE, 289–300.

[18] Barrett Ens, Fraser Anderson, Tovi Grossman, Michelle Annett, Pourang Irani, and George Fitzmaurice. 2017. Ivy: Exploring spatially situated visual programming for authoring and understanding intelligent environments. In *Proceedings - Graphics Interface*.

[19] Barrett Ens, Juan David Hincapié-Ramos, and Pourang Irani. 2014. Ethereal planes: a design framework for 2D information space in 3D mixed reality environments. In *Proceedings of the 2nd ACM symposium on Spatial user interaction*. 2–12.

[20] Barrett M Ens, Rory Finnegan, and Pourang P Irani. 2014. The personal cockpit: a spatial interface for effective task switching on head-worn displays. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 3171–3180.

[21] Sevinc Eroglu, Patric Schmitz, Carlos Aguilera Martinez, Jana Rusch, Leif Kobbelt, and Torsten W. Kuhlen. 2020. Rilievo: Artistic scene authoring via interactive height map extrusion in VR. In *ACM SIGGRAPH 2020 Art Gallery, SIGGRAPH 2020*. https://doi.org/10.1145/3386567.3388577

[22] Sevinc Eroglu, Frederic Stefan, Alain Chevalier, Daniel Roettger, Daniel Zielasko, Torsten W Kuhlen, and Benjamin Weyers. 2021. Design and evaluation of a free-hand vr-based authoring environment for automated vehicle testing. In *2021 IEEE Virtual Reality and 3D User Interfaces (VR)*. IEEE, 1–10.

[23] Steven Feiner, Blair MacIntyre, Marcus Haupt, and Eliot Solomon. 1993. Windows on the world: 2D windows for 3D augmented reality. In *Proceedings of the 6th annual ACM symposium on User interface software and technology*. 145–155.

[24] Vicki Ha, Jim Wallace, Ryder Ziola, and Kori Inkpen. 2006. My MDE: configuring virtual workspaces in multi-display environments. In *CHI'06 Extended Abstracts on Human Factors in Computing Systems*. 1481–1486.

[25] Sandra G Hart. 2006. NASA-task load index (NASA-TLX); 20 years later. In *Proceedings of the human factors and ergonomics society annual meeting*, Vol. 50. Sage publications Sage CA: Los Angeles, CA, 904–908.

[26] Martin Hedlund, Adam Jonsson, Cristian Bogdan, Gerrit Meixner, Elin Ekblom Bak, and Andrii Matviienko. 2023. BlocklyVR: Exploring Block-based Programming in Virtual Reality. In *Proceedings of the 22nd International Conference on Mobile and Ubiquitous Multimedia*. 257–269.

[27] Paul Jen-Hwa Hu, Pai-Chun Ma, and Patrick YK Chau. 1999. Evaluation of user interface designs for information retrieval systems: a computer-based experiment. *Decision support systems* 27, 1-2 (1999), 125–143.

[28] Bret Jackson and Daniel F. Keefe. 2016. Lift-Off: Using Reference Imagery and Freehand Sketching to Create 3D Models in VR. *IEEE Transactions on Visualization and Computer Graphics* (2016). https://doi.org/10.1109/TVCG.2016.2518099

[29] Dominic Kao, Christos Mousas, Alejandra J. Magana, D. Fox Harrell, Rabindra Ratan, Edward F. Melcer, Brett Sherrick, Paul Parsons, and Dmitri A. Gusev. 2020. Hack.VR: A Programming Game in Virtual Reality. arXiv:2007.04495 [cs.HC]

[30] Pascal Knierim, Valentin Schwind, Anna Maria Feit, Florian Nieuwenhuizen, and Niels Henze. 2018. Physical keyboards in virtual reality: Analysis of typing performance and effects of avatar hands. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. 1–9.

[31] Marcel Krüger, Tim Gerrits, Timon Römer, Torsten Kuhlen, and Tim Weissker. 2024. IntenSelect+: Enhancing Score-Based Selection in Virtual Reality. *IEEE Transactions on Visualization and Computer Graphics* (2024).

[32] Bettina Laugwitz, Theo Held, and Martin Schrepp. 2008. Construction and evaluation of a user experience questionnaire. In *HCI and Usability for Education and Work: 4th Symposium of the Workgroup Human-Computer Interaction and Usability Engineering of the Austrian Computer Society, USAB 2008, Graz, Austria, November 20-21, 2008. Proceedings 4*. Springer, 63–76.

[33] Joseph J LaViola Jr, Ernst Kruijff, Ryan P McMahan, Doug Bowman, and Ivan P Poupyrev. 2017. *3D user interfaces: theory and practice*. Addison-Wesley Professional.

[34] Irina Lediaeva and Joseph LaViola. 2020. Evaluation of body-referenced graphical menus in virtual environments. In *Graphics Interface 2020*.

[35] Gun A Lee, Claudia Nelles, Mark Billinghurst, and Gerard Jounghyun Kim. 2004. Immersive authoring of tangible augmented reality applications. In *Third IEEE and ACM International Symposium on Mixed and Augmented Reality*. IEEE, 172–181.

[36] David Lindlbauer, Anna Maria Feit, and Otmar Hilliges. 2019. Context-aware online adaptation of mixed reality interfaces. In *Proceedings of the 32nd annual ACM symposium on user interface software and technology*. 147–160.

[37] John T Murray. 2022. RealityFlow: Open-Source Multi-User Immersive Authoring. In *2022 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW)*. IEEE, 65–68.

[38] Michael Nebeling, Katy Lewis, Yu Cheng Chang, Lihan Zhu, Michelle Chung, Piaoyang Wang, and Janet Nebeling. 2020. XRDirector: A Role-Based Collaborative Immersive Authoring System. In *Conference on Human Factors in Computing Systems - Proceedings*. https://doi.org/10.1145/3313831.3376637

[39] Donald A. Norman. 2002. *The Design of Everyday Things*. Basic Books, Inc., USA.

[40] John F Pane and Brad A Myers. 1996. *Usability issues in the design of novice programming systems*. Carnegie-Mellon University. Department of Computer Science.

[41] Nicholas F Polys, Seonho Kim, and Doug A Bowman. 2005. Effects of information layout, screen size, and field of view on user performance in information-rich virtual environments. In *Proceedings of the ACM symposium on Virtual reality software and technology*. 46–55.

[42] Kevin Ponto, Ross Tredinnick, Aaron Bartholomew, Carrie Roy, Dan Szafir, Daniel Greenheck, and Joe Kohlmann. 2013. SculptUp: A rapid, immersive 3D modeling environment. In *2013 IEEE Symposium on 3D User Interfaces (3DUI)*. 199–200. https://doi.org/10.1109/3DUI.2013.6550247

[43] Eric D Ragan, Doug A Bowman, Regis Kopper, Cheryl Stinson, Siroberto Scerbo, and Ryan P McMahan. 2015. Effects of field of view and visual complexity on virtual reality training effectiveness for a visual scanning task. *IEEE transactions on visualization and computer graphics* 21, 7 (2015), 794–807.

[44] Ruth Rosenholtz, Yuanzhen Li, and Lisa Nakano. 2007. Measuring visual clutter. *Journal of vision* 7, 2 (2007), 17–17.

[45] Kadek Ananta Satriadi, Barrett Ens, Maxime Cordeil, Tobias Czauderna, and Bernhard Jenny. 2020. Maps around me: 3d multiview layouts in immersive spaces. *Proceedings of the ACM on Human-Computer Interaction* 4, ISS (2020), 1–20.

[46] Brenden Sewell. 2015. *Blueprints visual scripting for unreal engine*. Packt Publishing Ltd.

[47] Solirax. 2018. Neos Wiki. https://wiki.neosvr.com/Main_Page. (Accessed on 06/03/2024).

[48] Tiago Boldt Sousa. 2012. Dataflow Programming: Concept, Languages and Applications. In *7th Doctoral Symposium on Informatics Engineering*.

[49] Tiago Boldt Sousa. 2012. Dataflow programming concept, languages and applications. In *Doctoral Symposium on Informatics Engineering*, Vol. 130.

[50] Anthony Steed. 2006. Towards a general model for selection in virtual environments. In *3D User Interfaces (3DUI'06)*. IEEE, 103–110.

[51] Anthony Steed and Mel Slater. 1996. Dataflow representation for defining behaviours within virtual environments. In *Proceedings - Virtual Reality Annual International Symposium*. https://doi.org/10.1109/vrais.1996.490524

[52] Jonathan Steuer, Frank Biocca, Mark R Levy, et al. 1995. Defining virtual reality: Dimensions determining telepresence. *Communication in the age of virtual reality* 33 (1995), 37–39.

[53] John Sweller, Jeroen JG Van Merrienboer, and Fred GWC Paas. 1998. Cognitive architecture and instructional design. *Educational psychology review* 10 (1998), 251–296.

[54] Barbara Tversky. 2013. Visualizing thought. In *Handbook of human centric visualization*. Springer, 3–40.

[55] Daniel Vogel, Paul Lubos, and Frank Steinicke. 2018. AnimationVR-interactive controller-based animating in virtual reality. In *2018 IEEE 1st Workshop on Animation in Virtual and Augmented Environments (ANIVAE)*. IEEE, 1–6.

[56] Jia Wang, Owen Leach, and Robert W Lindeman. 2013. DIY World Builder: an immersive level-editing system. In *2013 IEEE Symposium on 3D User Interfaces (3DUI)*. IEEE, 195–196.

[57] Lei Zhang, Doug A Bowman, and Caroline N Jones. 2019. Exploring effects of interactivity on learning with interactive storytelling in immersive virtual reality. In *2019 11th International Conference on Virtual Worlds and Games for Serious Applications (VS-Games)*. IEEE, 1–8.

[58] Lei Zhang and Steve Oney. 2019. Studying the Benefits and Challenges of Immersive Dataflow Programming. In *Proceedings of IEEE Symposium on Visual Languages and Human-Centric Computing, VL/HCC*. https://doi.org/10.1109/VLHCC.2019.8818856

[59] Lei Zhang and Steve Oney. 2019. Studying the benefits and challenges of immersive dataflow programming. In *2019 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*. IEEE, 223–227.

[60] Lei Zhang and Steve Oney. 2020. FlowMatic: An Immersive Authoring Tool for Creating Interactive Scenes in Virtual Reality. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology*. 342–353.