ScalAR: Authoring Semantically Adaptive Augmented Reality Experiences in Virtual Reality

Xun Qian
School of Mechanical Engineering, Purdue University
West Lafayette, IN, USA
qian85@purdue.edu

Fengming He
School of Electrical & Computer Engineering, Purdue University
West Lafayette, IN, USA
he418@purdue.edu

Xiyun Hu
School of Mechanical Engineering, Purdue University
West Lafayette, IN, USA
hu690@purdue.edu

Tianyi Wang
School of Mechanical Engineering, Purdue University
West Lafayette, IN, USA
wang3259@purdue.edu

Ananya Ipsita
School of Mechanical Engineering, Purdue University
West Lafayette, IN, USA
aipsita@purdue.edu

Karthik Ramani
School of Mechanical Engineering, Purdue University
West Lafayette, IN, USA
ramani@purdue.edu

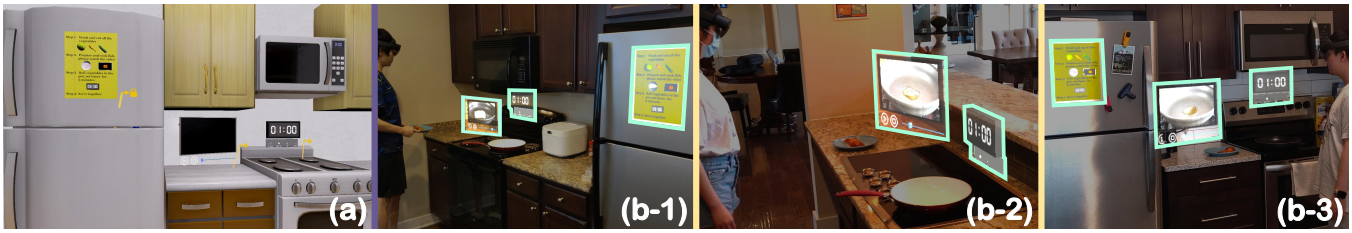

Figure 1: An overview of ScalAR. (a) A VR authoring environment that consists of various virtual scenes synthesized from the collected real-world sample scenes. A designer is immersed in the virtual environment and authors the semantic-level associations of the AR contents in each scene, then adjusts and validates the design by traversing across all the given scenes. (b-1 to b-3) ScalAR adaptively renders the AR contents in different target environments using the *semantic adaptation model* fit from the designer's demonstrations.

ABSTRACT

Augmented Reality (AR) experiences tightly associate virtual contents with environmental entities. However, the dissimilarity of different environments limits the adaptive AR content behaviors under large-scale deployment. We propose ScalAR, an integrated workflow enabling designers to author *semantically adaptive* AR experiences in Virtual Reality (VR). First, potential AR consumers collect local scenes with a semantic understanding technique. ScalAR then synthesizes numerous similar scenes. In VR, a designer authors the AR contents' semantic associations and validates the design while being immersed in the provided scenes. We adopt a decision-tree-based algorithm to fit the designer's demonstrations as a *semantic adaptation model* to deploy the authored AR experience in a physical scene. We further showcase two application scenarios authored by ScalAR and conduct a two-session user study where the quantitative results prove the accuracy of the AR content rendering and the qualitative results show the usability of ScalAR.

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

ACM ISBN 978-1-4503-9157-3/22/04.
https://doi.org/10.1145/3491102.3517665

CCS CONCEPTS

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

KEYWORDS

Augmented Reality, Virtual Reality, Semantic Understanding, Immersive Authoring, Adaptation

ACM Reference Format:
Xun Qian, Fengming He, Xiyun Hu, Tianyi Wang, Ananya Ipsita, and Karthik Ramani. 2022. ScalAR: Authoring Semantically Adaptive Augmented Reality Experiences in Virtual Reality. In *CHI Conference on Human Factors in Computing Systems (CHI '22), April 29-May 5, 2022, New Orleans, LA, USA.* ACM, New York, NY, USA, 18 pages. https://doi.org/10.1145/3491102.3517665

1 INTRODUCTION

Augmented Reality (AR) technology has been broadly applied to education [90, 98], medical [74], entertainment [63, 69] and engineering [10, 62] areas. One substantial advantage of these AR experiences is the tight associations between the functionalities of the digital assets and the affordance of the physical environments that are established by the precise placements and bindings defined during designing processes. Although existing authoring tools support designers to create AR contents that are tracking-based (i.e., can be fixed in mid-air using the tracking capability of

AR capable devices [75]), marker-based (i.e., can be attached to fiducial markers [27, 73]), or geometry-based (i.e., can be aligned with edges/planes/meshes [18, 65]), few authoring systems enable creating *semantically adaptive* AR experiences which possess semantic understanding [12, 32] of the surrounding environments and appropriately adjust the spatial relationships between the involving digital contents and the relevant physical objects.

Unlike *tracking-based*, *marker-based*, and *geometry-based* AR experiences, which can be easily deployed in different scenes as long as the same tracking features exist, it is challenging to author *semantically adaptive* AR experiences due to the complexity of granting consistent spatial and semantic associations under diversified deployment scenarios. For instance, a designer may want to place an AR painting that is hung on the wall behind the sofa while holding the similar size of the table in front of the sofa. Meanwhile, the same AR experience is expected to be consumed both in a living room with a sofa and two tables, and another living room with two chairs but no table. To this end, we are motivated to explore an authoring workflow that supports designers to define the semantic relationships of the AR contents, validate and test the design in abundant scenarios, and produce AR experiences that AR consumers can broadly share with each other.

The metaphor of immersive authoring provides designers with an in-situ and ad-hoc authoring experience [36, 88, 95]. While being situated in an environment, designers can exploit the surroundings as spatial and contextual references to endow semantic awareness to the AR contents. However, immersive authoring only allows designers to customize AR experiences for one specific scene. The AR contents do not hold explicit relationships with the physical entities, which implies that any slight variation in a different environment may easily invalidate the design. In addition, it is time and effort consuming for designers to be physically present in one or more target scenes. On the other hand, Virtual Reality (VR) has been broadly adopted owing to the immersive authoring capability [59, 97]. Typically, designers can create AR experiences in VR as well by referring to the virtual replica of the physical environments [17, 70, 89]. Meanwhile, the flexibility of VR enables designers to travel across different scenarios free from temporal and spatial limitations [38, 92]. We envision that designers can efficiently tailor and validate their designs in numerous virtual environments to grant AR experiences with semantic-level adaptation capability.

We present ScalAR, an integrated authoring workflow that allows designers to create AR experiences that can be generalized to various environments while using existing and synthesized scenes as semantic references. ScalAR is composed of three interconnected parts: (1) An AR scanning application that supports collecting sample physical scenes from local environments, (2) a VR authoring studio for creating AR experiences, and (3) an AR client that deploys the AR experiences in the environments. Once an AR designer has a basic idea of where the AR experience should be deployed (e.g., office, kitchen, living room, etc.), the designer can invite potential AR consumers to capture their local scenes as samples. The AR scanning application records the spatial layouts of the scenes leveraging a scene understanding algorithm. To provide the designer with diverse validation scenarios, we synthetically generate more samples using a genetic algorithm. Then, these samples are imported into the VR authoring studio and are represented as realistic virtual scenes (Figure 1a). While being immersed in a virtual scene, the designer can exploit environmental affordance as references and define the semantic behaviors of the AR contents with the help of the VR authoring interface. The VR authoring studio also allows for rapid navigation across multiple virtual scenes to enable the designer to modify and validate the design. Upon the completion of the authoring, ScalAR follows a decision-tree-based algorithm to fit the designer's demonstrations in all the provided scenes as a *semantic adaptation model*. Finally, in the AR client, the model is utilized to deploy the authored AR experience based on the AR consumers' local layouts (Figure 1b-1 to b-3). In summary, we highlight our contributions as follows:

- An integrated system workflow for collecting physical scenes, defining and validating *semantically adaptive* AR experiences in synthetically generated VR environments, and deploying the experiences in different physical scenes.
- An AR interface for scanning the physical environment, an immersive VR authoring studio for manipulating AR contents and traversing across multiple scenes, and an AR interface for deploying the AR experiences.
- A decision-tree-based algorithm that fits an AR designer's demonstrations as a *semantic adaptation model* for adaptively rendering the AR contents in different deploying environments.

2 RELATED WORKS

2.1 *Semantically Adaptive* AR Experiences

Many AR experiences have been proposed in various application areas for improving working efficiency, industrial productivity, and quality of life [4]. An AR experience precisely superimposes virtual contents on the real world to intuitively deliver digital augmentation with respect to the affordance and functionalities of the attaching physical entities. More importantly, the AR experience is expected to adaptively maintain the same associations in diverse and dissimilar deploying environments. Considering the techniques that are utilized to achieve this capability, we classify AR experiences into four major types: ***tracking-based***, ***marker-based***, ***geometry-based***, and ***semantic-based*** AR experiences (Figure 2).

Tracking-based AR experiences utilize the tracking capability of AR to fix 3D digital contents in the real world. Spatial [75] renders virtual avatars next to AR consumers to improve the social presence during remote collaborations. Mobi3DSketch [43] allows for creating 3D sketches floating in mid-air. These AR experiences can be deployed anywhere using an AR-capable device since they usually focus on realistic 3D visualization and standalone functionalities. Yet, the digital augmentation has limited spatial or semantic associations with the deploying environments.

Marker-based AR experiences use intermediate tangible carriers to connect digital assets with the physical world. AR contents can be attached onto spatially placed fiducial markers [19, 68, 71, 85, 86]. Meanwhile, ARIoT [39] and Scenariot [35] leverage smart objects as the spatial and functional references to render AR icons and interfaces in-situ. Consumers can visualize the digital contents as long as the reference markers exist in the target scene. Yet, the additional settings and indirect spatial associations hinder the spatiality and flexibility of these experiences.

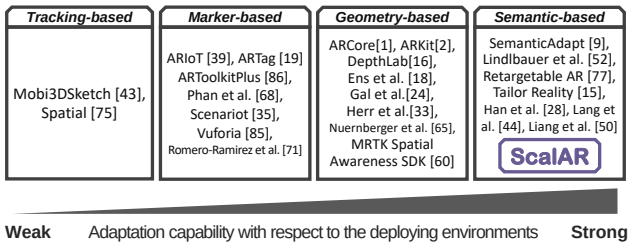

Figure 2: The four types of AR experiences. *Tracking-based*: Fix AR contents in mid-air through the tracking capability of AR. *Marker-based*: Attach AR contents to fiducial markers or smart objects. *Geometry-based*: Place AR contents by referring to the geometric features in the environment. *Semantic-based*: Render AR contents considering the semantic information of the physical entities. ScalAR aims at the authoring of *semantic-based* AR experiences.

Geometry-based AR experiences bind AR contents with the geometric features of the physical entities that are extracted by various computational algorithms. Typically, AR contents can be aligned with edges [65], surfaces [1, 2, 33], depth maps [16], and 3D meshes [18, 24, 60], which largely enriches application scenarios and reduces deployment difficulties. However, impreciseness and ambiguity of the AR content placements may be introduced due to the duplicated and unexpected features detected in the environments (e.g., an AR potted plant could be unreasonably rendered on a sofa plane, or on a mistakenly detected plane on a floor lamp).

Semantic-based AR experiences rely on both the semantic meanings and spatial properties of physical entities to render AR contents by leveraging semantic understanding techniques [12, 32]. Han et al. [28] achieves realistic trajectory simulation of the AR contents by assigning different material properties to the engaging physical objects. Retargetable AR [77] adjusts AR avatars' behaviors according to the functionalities of the nearby objects. Researchers further implement optimization algorithms to fulfill the adaptation. Liang et al. [50] develops an algorithm to synthesize AR pets' behaviors based on the indoor layouts. Lang et al. [44] and SemanticAdapt [9] optimally place MR assets and interfaces around users considering the semantic and functional associations with respect to the physical objects, while Lindlbauer et al. [52] utilizes the semantic implications behind human activities to adjust the level-of-detail of MR interfaces. Tailor Reality [15] further embraces human's visual perception of different entities as the semantic references to restructure physical layouts in MR. Compared with *geometry-based* approaches, the additional identity information partially resolves the ambiguity of rendering AR contents in complicated scenes. More importantly, these AR experiences associate the functionalities of the AR contents with the affordance of the surrounding objects so that they can be accurately deployed to different environments. Considering the semantic-level perception and adaptation capability, we identify this type of AR experiences as ***semantically adaptive*** AR experiences.

In most *semantic-based* systems, the adaptation is limited to the object-level semantic associations that can be explicitly interpreted by the pre-designed computational algorithms. Yet, spatial variation and duplication/absence of the physical objects in different scenes may generate unexpected corner cases that reduce the performance of the algorithms. For instance, an AR cat that is designed to jump from a sofa to the coffee table may not behave reasonably in a room where there is no coffee table in front of the sofa or there is only a dining table but a real cat usually does not tend to jump onto it. In contrast, we aim to adopt an ad-hoc authoring process to eliminate the implicit semantic nuances and expand the semantic-level AR content associations to the entire environment. However, although multiple authoring tools have been proposed for creating *tracking-based* (e.g., Unity3D [80] supports designers to create 3D holograms for AR-capable devices), *marker-based* (e.g., Vuforia [85] allows for creating AR contents attached on image targets), and *geometry-based* AR experiences (e.g., ARKit [2] assists rendering virtual assets on the 3D mesh of the environments), few works support authoring *semantically adaptive* AR experiences owing to the complexity of accurately defining the semantic-level associations while maintaining the adaptation capability in diverse deploying scenes. To this end, we strive to design an authoring system enabling designers to create *semantically adaptive* AR experiences that extensively utilize the semantic information of the physical environments and actively adjust the AR content behaviors in different scenarios.

2.2 AR Authoring Tools

Authoring AR experiences requires designers to explicitly assign spatial behaviors of the virtual contents with respect to the physical environments. Desktop-based AR authoring tools [25, 46, 55] have shown compelling capabilities (e.g., Unity3D [80] and Unreal [81] support designers to build cross-platform AR applications and design complicated AR animations and interactions). Yet, since designers are isolated from the target environments, the spatial and semantic associations relevant to the physical entities, which are one of the critical characteristics of AR experiences, may not be accurately and smoothly defined. On the other hand, borrowing the metaphor of the immersive authoring [47], prior arts bring designers to the target environments during authoring. By leveraging physical objects as spatial references, designers can create AR contents that are precisely aligned with the surroundings [36, 45]. SemanticPaint [82] and SceneCtrl [95] immerse users into the semantic scans of the environments to facilitate labeling and modifying physical scenes in AR, while Pronto [49] enables designers to add AR contents into videos by immersive enactions. In addition, spatially sensitive AR animations and interactions can be created through in-situ demonstrations by referring to the physical entities [5, 6, 48, 93]. However, these works cannot grant the adaptation capability addressed in the previous section since the AR content behaviors are defined through in-situ manipulation without direct virtual-to-physical bindings. Therefore, while acknowledging the advantages of the immersive authoring metaphor, we aim to develop a system that allows designers to provide immersive manipulations of the AR contents in diversified scenarios and distills the demonstrations as a mathematical model for adaptively rendering the AR contents in different environments.

2.3 Using VR for Authoring

Researchers have leveraged Virtual Reality (VR) as authoring platforms to create VR/AR applications. One substantial attribute of VR is the immersive experience within the 3D domain (e.g., immersive VR games [67] and designs [78]). Due to the high similarity between AR and VR in terms of the immersiveness, prior works have proposed to author AR experiences in the virtual replica of the physical environments to fully leverage the environmental affordance [17, 70]. XRDirector [61] introduces a cross-device immersive authoring system to collaboratively create interactive AR/VR experiences. Meanwhile, many works have explored the capability of transferring the spatial and contextual information in the real world to VR through immersive demonstration in the physical environments [8, 29, 51, 76]. On the other hand, the flexibility of the VR domain allows effortless travel across infinite spaces and empowers users with more capabilities beyond physical limitations. Some works immerse trainees in numerous virtual environments to rapidly learn skills [72, 96]. In VR, users can experience adaptive real-walking that is not limited to the current situated physical spaces [56, 83]. Spacetime [92] introduces a novel interaction form allowing VR users to edit virtual objects without the limitation of space and time. Similarly, Remixed Reality [53] provides users with the freedom to travel in space and time in a virtual environment. Furthermore, Jetter et al. [38] and Horizon Workrooms [34] construct versatile and adaptive VR spaces that support realistic sketching and typing experiences. FlowMatic [97] supports programmers to create interactive virtual scenes through an immersive visual programming tool. Most recently, DistanciAR [89] allows designers to remotely author on-site AR experiences while being situated in a VR scene.

Inspired by these prior arts, we embrace VR as the main authoring interface. Our system constructs the authoring environment by collecting physical layouts from the real world and rendering them in VR using realistic CAD models. This way, the affordance of the original environment can be mostly preserved. Designers can conduct the same immersive authoring as they do in an AR environment. Moreover, since VR can free users from spatial limitations, we synthesize more virtual scenes and enable designers to rapidly assign and adjust the AR content behaviors under diversified conditions without leaving the current physical location. Additionally, by introducing the immersive programming metaphor adopted in prior works [97], designers can explicitly define the semantic-level associations of the AR contents, which further facilitates our system to convert designers' demonstrations as the adaptation models to render the AR experiences in different physical environments.

3 SCALAR

3.1 Characteristics of *Semantically Adaptive* AR Experiences

It is important to first clarify the characteristics of the *semantically adaptive* AR experiences considered in this work. First, we aim to author an AR experience that is deployed in a physical scene with a series of spatially distributed objects that hold specific affordance. For example, an AR decoration experience is deployed in a *living room* with a virtual curtain to cover the window, a virtual fireplace beneath the TV, and several virtual paintings hung on the wall. Typically, the AR contents are spatially aligned with multiple physical objects to achieve the semantic-level digital augmentation (e.g., in Figure 1b, an AR display is precisely placed on the surface of a countertop next to a range in order to facilitate AR consumers to watch video tutorials while preparing and cooking foods). To this end, two levels of associations should be explicitly defined for a *semantic adaptive* AR experience:

- ***Identity association*** implies which physical entities the AR content should be bound to in different scenes. For example, in Figure 1b, the AR display is bound with the countertop that is next to the range when there is more than one countertop, otherwise directly bound with the range.
- ***Spatial association*** specifies the ***spatial attributes*** (i.e., the 6DOF transform and the 3DOF scale of an physical/virtual element) of the AR content relative to the binding physical entities' ***geometric features*** (i.e., the vertices, edges, faces, and volume of the object's 3D bounding box) following the prior AR content placement works [9, 31, 65]. E.g., in Figure 1b, the AR display is precisely aligned with the back edge on the countertop's top surface while the size is coupled with its width.

Furthermore, to ensure that the digital augmentation is consistently and accurately delivered in different scenes, the AR experiences should dynamically adjust the two-level associations in response to the potential cross-scene variations. Specifically, we classify the scene variations as:

- ***Spatial variation*** represents the change of the *spatial attributes* of the physical entities. For instance, the sizes of the countertops vary in Figure 1b-1 and 1b-3. Or, a coffee table is located at different sides of a chair.
- ***Quantity variation*** refers to the change of the quantity of each physical object. For example, in Figure 1b-2, no countertop exists. Or, the first scene consists of two chairs and one table, while the second only has one chair.

3.2 ScalAR System Design

Following the characteristics discussed above, we present ScalAR, an authoring workflow that enables an AR designer to define and adjust the two-level associations of each AR content within numerous VR scenes that are synthesized from the real environments with the two-level variations. Then, we fit the designer's demonstrations as a generalized model using a decision-tree-based algorithm for each AR content. Finally,the model is applied to render the corresponding AR content in different scenes. Now, we describe the workflow of ScalAR by the five consecutive steps (three major sections and two transition steps) shown in Figure 3.

After a designer decides to create an AR experience for a specific scene (e.g., an AR experience for *living room* environments), multiple potential AR consumers are invited to collect their local scenes including the identities and *spatial attributes* of all the present physical objects using a semantic understanding technique supported by our system (Figure 3a). For each collected scene, a scene synthesis algorithm is applied to generate a large collection of virtual scenes that consist of the same objects but vary in their *spatial attributes*, which is defined as a ***semantic group***. Typically, the scene variations are achieved in this process where the *spatial variations*

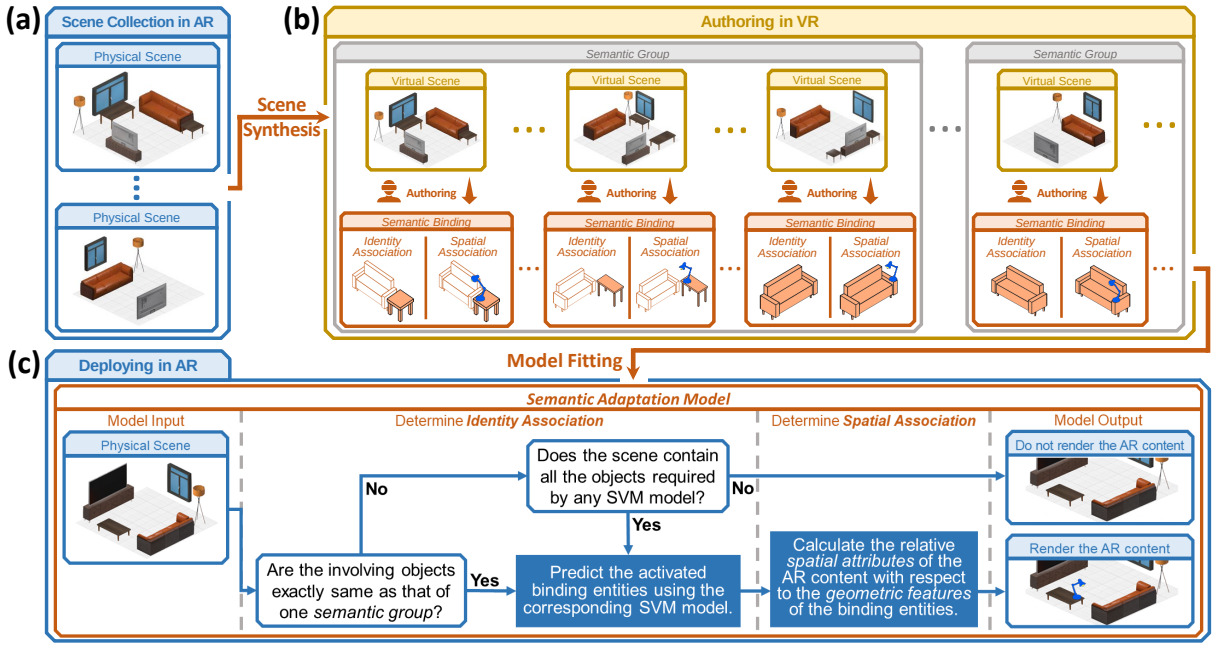

Figure 3: The workflow of ScalAR. (a) Potential AR consumers collect their local physical scenes. (a to b) ScalAR synthesizes plentiful virtual scenes that are highly similar to the collected scenes and groups them accordingly. (b) In VR, a designer defines both the *identity* and *spatial associations* of each AR content as *semantic bindings*, and adjusts them in all the provided virtual scenes. (b to c) We then use the authored *semantic bindings* to fit a *semantic adaptation model*. (c) When deploying the AR experience in a physical scene, the *semantic adaptation model* takes the physical layout as the input and consecutively determines the two-level associations of the AR content.

happen within each *semantic group*, while the *quantity variations* exist across different *semantic groups*. ScalAR then uploads both the real and generated scenes to the VR side and spatially distributes the involving objects using pre-assigned CAD models to construct the authoring environment (e.g., all the virtual scenes in Figure 3b).

Now in VR, the designer is immersed in one virtual scene to start the authoring process. ScalAR supports the designer to define the behavior of each AR content using ***semantic bindings*** that represent the two-level associations relative to the current scene. First, the designer informs our system of the *identity associations* by defining a *primary binding object* and more *reference binding objects* if applicable.

- A ***primary binding object*** represents the physical entity that dominates the semantic and spatial association of the AR content. For instance, in the first virtual scene in Figure 3b, the designer intends to place a virtual lamp on the *table* (filled) when there exist *tables* in the living room.
- A ***reference binding object*** serves as an additional reference of the *primary binding object* for two purposes. (1) Clarify the identity of the *primary binding object* when it is duplicated in the scene. For example, the first virtual scene in Figure 3b involves two *tables*. The designer assigns the *sofa* (unfilled) as the *reference binding object* to indicate that the AR lamp should always be placed on the *table* that is on the left side of the *sofa*. (2) Address the needs when the *spatial association* of the AR content is coupled with multiple objects. In the first virtual scene in Figure 3b, the relative *spatial attributes* of the AR lamp are expected to be associated with both the *table* and the *sofa*.

The designer then authors the *spatial association* by manipulating the AR content while referring to the *geometric features* of both the *primary binding object* and *reference binding objects*. In the first virtual scene in Figure 3b, the AR lamp is placed at the corner of the *table* surface that is closest to the *sofa*, and it is always facing towards the *sofa*.

Following the authoring in the initial scene, the designer starts to travel across all the other provided scenes within and across different *semantic groups* to adjust or re-define the *semantic bindings* in response to the provided scene variations. For instance, within the first *semantic group* of Figure 3b, the designer adjusts the *spatial association* of the AR lamp when the layout of the *sofa* and *tables* varies. When both the *tables* are far from the *sofa* as shown in the third virtual scene, the designer also changes the *identity association* to the *sofa* (filled) while placing the lamp on the armrest. Meanwhile, in another *semantic group* where no *table* exists, the designer assigns the lamp's *identity association* to the *sofa* (filled) and places it on the armrest. Typically, in order to reduce the workload, we pre-assign *semantic bindings* to the rest of the virtual scenes based on the authoring in the initial scene. This way, the designer only performs adjustments in the scenes with significant variations. The details of this feature will be explained in Section 4.3.

After the designer validates all the provided scenes, our system implements a decision-tree-based fitting algorithm to distill the

designer's demonstrations as a ***semantic adaptation model*** for rendering the corresponding AR content in a physical scene. Briefly, for each *semantic group*, we train an SVM model for *identity association* prediction using the virtual scene layouts as inputs and the activated *identity association* objects as outputs. Then, for each group of the demonstrations that holds the same *identity association*, we use the *spatial association* information to fit an AR content placement model in two steps: (1) select the *geometric features* of the *identity association* objects that dominate the AR content placement, and (2) calculate the average *spatial attributes* with respect to the corresponding *geometric features* for placing the AR content. The fitting details will be explained in Section 4.4.

Eventually, when deploying the authored AR experience, our system follows the *semantic adaptation model* (Figure 3c) to render each AR content given the new physical scene as the model input. Specifically, ScalAR (1) compares the involving objects with the required inputs of all the trained SVM models to select one applicable SVM model to determine the *identity association*. Note that, if no proper SVM model exists, our system will not render the AR content, and (2) fetches the corresponding AR content placement model, adopts the activated *geometric features*, and calculates the relative *spatial attributes* of the AR content to determine the *spatial association*. For instance, in Figure 3c, the AR lamp is accurately rendered using the detected *table* and *sofa* as the spatial references.

4 IMPLEMENTATION OF SCALAR

In this section, we explain the design and implementation of ScalAR step-by-step, namely, (1) physical scene collection, (2) virtual scene synthesis, (3) authoring interface in VR, (4) fitting of the *semantic adaptation model*, and (5) deployment of the AR experience.

4.1 Physical Scene Collection

Some prior *semantic-based* works [50, 77, 82] implement semantic segmentation [12, 26] to acquire the identity and spatial information of the physical environments. Yet, these approaches require massive training data and computational power. Instead, we adopt an integrated approach leveraging a 2D object detection network and a 3D scene understanding module to record the physical scene using any AR-enabled device with scene understanding capability. Specifically, a HoloLens 2 [58] is used in our implementation.

A potential AR consumer wears the AR-HMD and uses the AR-side system of ScalAR to complete the scene collection process. The consumer needs to complete two consecutive steps, *plane detection* and *object detection*, to collect the required data of the scene. An AR main menu that floats next to the consumer's left hand is used to select the steps and provide the corresponding textual guidance (Figure 4a-1). First, the consumer enters the *plane detection* step and simply walks inside the environment. ScalAR adopts a scene understanding module[1] embedded in the HoloLens 2 to detect and record the 6DOF of the physical planes (including walls, floors, and object surface planes if applicable) and to dynamically display them in AR (Figure 4b-1). After confirming that all the critical planes are detected, the consumer enters the *object detection* step and keeps walking in the environment while looking at the surrounding physical objects sequentially. In this step, ScalAR records the RGB images perceived by the AR device, the raycasts of the consumer's head transform, and the timestamps.

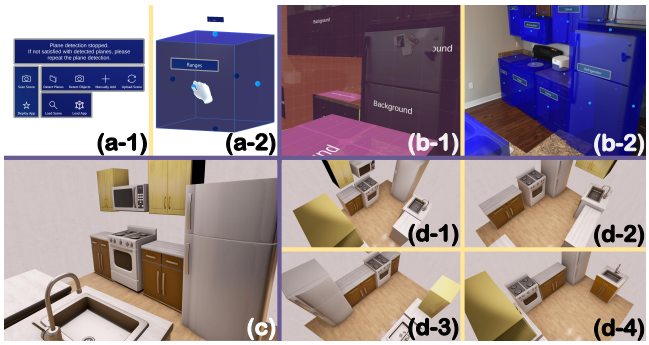

Figure 4: Physical scene collection and virtual scene synthesis. (a-1) The main menu of the AR-side system of ScalAR for collecting the physical scenes and deploying the AR experiences. (a-2) The 3D bounding box to represent the physical object. (b-1) The detected physical planes are dynamically displayed in the scene. (b-2) The labeled 3D bounding boxes are displayed in-situ after the data processing. (c) ScalAR uses pre-assigned CAD models as the virtual replica of the detected objects to build a realistic authoring environment. (d-1 to d-4) The synthesized virtual scenes with *spatial variations* and *quantity variations*.

Now, we explain how ScalAR infers the identities and *spatial attributes* of the involving objects. We pre-train an RGB-based 2D object detection neural network with the capability to detect 15 objects (refer to Section 4.6) that are relevant to our application scenarios and return the 2D bounding boxes. For each object label, we manually assign additional ***placement attributes***, namely, *wall-hanging* (e.g., cabinets and window blinds are always hung on the wall), *floor-stand* (e.g., ranges, dishwasher, and beds always stand on the floor), and *non-planar* (e.g., floor lamp does not have primary planes) to the applicable objects for two purposes: assist the generation of the 3D bounding boxes explained below and serve as the object properties required for scene synthesis described in Section 4.2. Every physical object has at least one *placement attribute*. We follow the proposed approaches [11, 88] to process the data. First, for each timestamp, we feed the RGB image to the object detection model to return the 2D bounding box of one object with the highest prediction score. Using the direction of the raycast, we project the center and four corners of the 2D bounding box from the image plane to the physical plane that the raycast intersects with. We mitigate the perspective distortion by averaging the opposite edges of the projected bounding box as the final width and height and centering this 2D rectangle at the projected center. Next, based on the object's *placement attribute* and whether the projected plane is vertical or horizontal, we extrude the projected bounding box from the projected plane to either the floor or the nearest wall. Since the recorded physical planes do not have labels, we identify the floor plane and the wall planes by finding the outermost and largest horizontal and vertical planes respectively. Finally, across

[1] https://docs.microsoft.com/en-us/windows/mixed-reality/design/scene-understanding

all the timestamps, we adopt an empirically set IoU threshold (refer to Section 4.6) to average all the 3D bounding boxes to one unique 3D bounding box for each detected object.

After the data processing, ScalAR displays the detected objects in-situ (Figure 4b-2). The consumer can adjust their *spatial attributes* through direct manipulation and add more physical objects by adding empty bounding boxes from the main menu (Figure 4a-2), assigning object labels, and adjusting the *spatial attributes*. Once the consumer finishes the entire scene collection, our system uploads the physical layout to our online server.

4.2 Virtual Scene Synthesis

Researchers have explored indoor scene synthesis with the optimization algorithms [22, 30, 41, 94]. We follow the same direction to adopt a genetic algorithm (GA) [40] based approach to synthesize more scenes for each collected physical scene to provide the AR designer with diverse scenarios that address the two-level scene variations. Specifically, we integrate Refixture[2] into our system where the *placement attributes* are pre-imported as the required parameters. ScalAR feeds each collected scene to the algorithm and saves the generated scenes at the backend. The implementation details of the GA algorithm are described in Section 4.6. Later, we group both the real and generated scenes with the same collection of involving objects into the same *semantic group*. Following this process, the *quantity variations* exist across the *semantic groups*, while the *spatial variations* lie within each *semantic group*. Typically, the total number of the synthesized scenes needs to fulfill the SVM model fitting requirements, which will be explained in Section 4.6.

ScalAR further sorts the *semantic groups* by the number of distinct identities of the involving objects. Meanwhile, within each *semantic group*, we cluster scenes into groups via the mean-shift clustering algorithm [22, 23] and sort the virtual scenes by minimizing the change of the *spatial attributes* of the objects between any two consecutive scenes. Finally, for each virtual scene, ScalAR applies the pre-assigned CAD models as the virtual replica of the involving objects and renders them according to the *spatial attributes* to construct the VR authoring environment. Figure 4c illustrates the virtualized scene of a collected physical scene, while Figure 4d-1 to d-4 showcases some of the synthesized scenes.

4.3 Authoring Interface in VR

By leveraging the advantages of immersive authoring and programming in VR, we develop the VR authoring studio allowing for both intuitive manipulation of the AR contents and abstract-level line connections to define the *semantic bindings* and additional dynamic behaviors of the AR contents. With ScalAR, a designer authors a *semantically adaptive* AR experience in VR with three modes: (1) ***Edit Mode*** to define and adjust AR content behaviors and *semantic bindings* in the current scene, (2) ***Traverse Mode*** to travel across different virtual scenes, and (3) ***Test Mode*** to check the usability of the AR experiences from an AR consumer's perspective. Similar to the AR-side interface, a left-hand-attached main menu is used to switch among the three authoring modes and toggle on/off other sub-menus for each mode (Figure 5a-1).

[2]https://jpc22.github.io

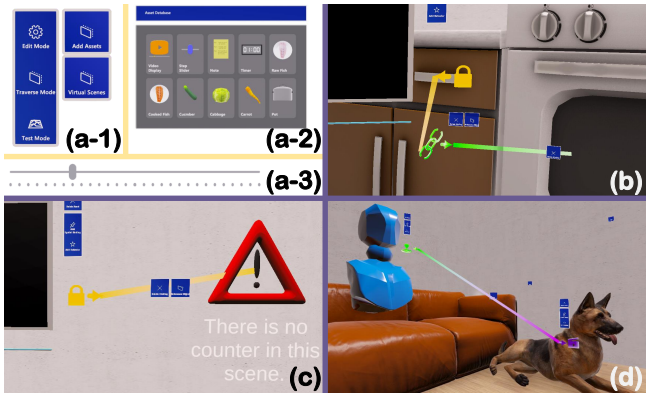

Figure 5: The VR authoring interface of ScalAR. (a-1) The main menu of the VR authoring studio of ScalAR with three buttons to switch among the three authoring modes and the sub-menus for *Edit Mode* and *Traverse Mode*. (a-2) The *asset menu* that holds all the pre-imported AR contents. (a-3) The *scene cursor* for navigating the virtual scenes in *Traverse Mode*. (b) A designer connects a *primary binding object* from the 'lock' icon of the AR content to the physical object and a *reference binding object* from the 'chain' icon of the *primary binding object* connection to another physical object through immersive programming. (c) The warning sign appeared next to the AR content when any of the binding objects is absent in the current virtual scene. (d) A designer creates a *trigger-behavior binding* from the 'avatar' model (*consumer asset*) to the 'animation' icon (*semantic animation*) of the AR content.

While being immersed in a virtual scene, the designer starts with the *Edit Mode* and drags a pre-defined AR content from the *asset menu* (Figure 5a-2) to the current scene. Then, the designer defines the current scene's *semantic binding*. For the *identity association*, by connecting a line from the 'lock' icon floating next to the AR content to the target environmental object through drag-and-drop interaction (Figure 5b), the designer can explicitly author the *primary binding object*. Meanwhile, the *reference binding object* can be defined in a similar way by creating the line connection from the 'chain' icon on the *primary binding object* connection to the target object (Figure 5b). The designer can always click the same button to toggle on/off the icon and the line connection for a clear view, and can delete an AR content or a line connection by clicking the delete button floating next to the UI elements. To define the *spatial association*, the designer can directly manipulate the AR content while utilizing the affordance of the object as references.

After finishing in one virtual scene, the designer enters the *Traverse Mode* to validate the initial design. We provide the designer with a ***scene cursor*** (Figure 5a-3) to travel across different scenes instead of repeatedly showing/hiding the physical objects. By dragging the *scene cursor*, the physical objects smoothly move, rotate and scale based on the next target virtual scene. To prevent the designer from adjusting the AR contents in every virtual scene to reduce the workload, we apply the *semantic binding* defined in the initial scene to the rest virtual scenes by fixing the *primary/reference*

*binding object*s and the relative *spatial attributes*. For instance, if a virtual display has been placed on the surface of a countertop next to the ranges in the first scene, it keeps following the moving countertop in the *Traverse Mode*. Meanwhile, if the *primary binding object* is duplicated in the current scene while the *reference binding object* is defined, our system automatically keeps the *primary binding object* as the one that holds the most similar relative *spatial attributes* with respect to the *reference binding object* in the initial design. For example, the virtual display always follows the countertop that is closest to the ranges if there is more than one countertop. Whenever the scene variation in one virtual scene causes significant impreciseness or semantic-level ambiguity of the AR content, the designer can stop traversing and re-enter the *Edit Mode* to either adjust the placement of the AR content or change the *identity association* under the current condition. Typically, the designer can deactivate the previous *primary binding object* by clicking the 'deactivate' button floating next to the connection line, and create a new connection as the activated *primary binding object*. Note that since the *reference binding object* is a property of the *primary binding object*, the *reference binding object* should be re-defined as well. ScalAR only records the activated *identity association* into the *semantic binding* of the current scene. For the *quantity variation* that happens across different *semantic group*s, if one of the binding objects does not exist, our system shows a warning sign (Figure 5c) forcing the designer to re-define the *semantic binding*. The *semantic binding*s in different *semantic group*s are recorded separately. So, if the designer enters another *semantic group* where the original binding objects reappear, the original *identity association* will be re-activated. The designer can repeat all the supported operations. During authoring, the designer validates all the provided scenes and conducts adjustments if necessary. Our system records all the *semantic binding*s even including that of the non-adjusted scenes as the data samples for the following model fitting process.

As addressed in the Related Works [9, 15, 52], an AR consumer is another important resource considered in *semantic-based* AR experiences. The spatial movement of the AR consumer is often related to the affordance of the physical objects (e.g., sit on the sofa, stand in front of the window). Meanwhile, AR contents may hold pre-designed animations that are spatially aligned with the environment surroundings [93]. Following the trigger-behavior metaphor that has been broadly implemented in AR/VR applications [87, 88, 97], ScalAR enables the designer to create dynamic AR content behaviors that can interactively react to the AR consumer. We introduce the ***consumer asset*** that represents the *spatial attributes* of an AR consumer and the ***semantic animation*** that represents the embedded dynamic behavior of an AR content. The designer can bind the *consumer asset* and the target of the *semantic animation* with the surroundings in the same way as other AR contents. Then, the designer can create a simple ***trigger-behavior binding*** using the same line connection operation between the *consumer asset* and the *semantic animation* (Figure 5d), implying that the *semantic animation* of the AR content is triggered when the AR consumer holds the designated *spatial attributes*. Leveraging the advantage of immersive authoring, our system enables the designers to role-play the AR consumer in the *Test Mode* to seamlessly test the correctness of the design in different virtual scenes.

4.4 Fitting of the *Semantic Adaptation Model*

For each AR content, ScalAR fits a *semantic adaptation model* using all the *semantic binding*s by implementing a decision-tree-based algorithm. First, within each *semantic group*, ScalAR trains an SVM-based *identity association* prediction model in order to determine the activated *identity association* objects given the deploying scene. Then, for each case of the activated *identity association*, ScalAR fits an AR content placement model for accurately rendering the AR contents with respect to the corresponding binding objects.

4.4.1 Fitting of the Identity Association Prediction Model. Within each *semantic group*, the designer may activate different *identity association*s in different scenes due to the *spatial variation*. We train an SVM model to predict which *identity association* should be activated based on the *spatial attributes* of the involving objects. Consider an AR content that has n available *primary binding object*s across L virtual scenes. Let $e_{j,l} \in \{0, 1\}$ denotes whether the AR content is bound to the j-th object in the l-th scene. We calculate a significance score $s_j = \sum_{l=1}^{L} e_{j,l}$ for each *primary binding object* and set the object with the highest significance score as the ***dominant object*** for this AR content. For example, the designer binds an AR display to a *countertop* in m virtual scenes, and to the *ranges* in the other $L - m$ virtual scenes, where $m > L - m$. So, the countertop becomes the *dominant object* for the AR display. Then, for each virtual scene, we construct a ***spatial feature vector*** consisting of the relative *spatial attributes* of the other $n - 1$ *primary binding object*s within the *dominant object*'s local coordinate system. In total, we have L *spatial feature vector*s and use them to train a multi-class SVM [91] with the one-versus-rest label encoding. After confirming the *primary binding object*, we also get its corresponding *reference binding object*s if applicable. We repeat the same procedure for the other *semantic group*s.

4.4.2 Fitting of the AR Content Placement Model. For every set of the *semantic binding*s that have the same activated *identity association*, we fit an AR content placement model to render the AR content relative to the binding physical objects. Similar to the prior works [3, 31, 65] that determine the spatial relationships between the virtual and physical objects by referring to the *geometric features* of the physical objects, our fitting process first decides which *geometric features* of the *identity association* objects the AR content should bind to, then calculates the *spatial attributes* of the AR content relative to the corresponding *geometric features*. Specifically, we categorize the types of the *geometric feature* bindings as:

- ***Point binding*** implies the AR content is associated with a vertex of the physical object's 3D bounding box. E.g., an AR content keeps a fixed distance from a *corner* of the *primary binding object* (Figure 6a), or faces towards the *centroid* of the object.
- ***Line binding*** implies the AR content is associated with an edge of the physical object's 3D bounding box. E.g., an AR content keeps a fixed distance from an *edge* of the *primary binding object* (Figure 6b), or revolves around an *edge* of the object.
- ***Plane binding*** implies the AR content is associated with a face of the physical object's 3D bounding box. E.g., an AR content moves on a *face* of the *primary binding object* (Figure 6c).

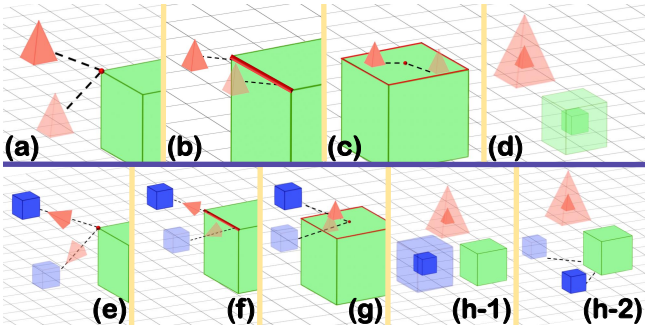

Figure 6: Examples of AR content spatial relationships with respect to physical entities. The opaque and translucent colors represent the same elements in different virtual scenes. Red pyramid: AR content; green cube: *primary binding object*; blue cube: *reference binding object*. Refer to the text for the detailed descriptions. (a) *Point binding*. (b) *Line binding*. (c) *Plane binding*. (d) *Volume binding*. (e) *Point binding* with *reference binding object*. (f) *Line binding* with *reference binding object* (g) *Surface binding* with *reference binding object*. (h-1 and h-2) *Volume binding* with *reference binding object*.

- ***Volume binding*** implies the volume of the AR content is associated with that of the physical object. E.g., an AR content's *size* increases as its *primary binding object* gets bigger (Figure 6d).

In order to determine whether an AR content holds a specific type of binding, we investigate the following three types of ***spatial relationship attributes*** of the *geometric features*. Note that the first two may happen for the *point/line/plane binding*s, while the third one is only valid for the *volume binding*s.

- ***Dis*** represents the Euclidean distance between the corresponding *geometric feature* and the AR content's centroid.
- ***rRot*** represents the angle between an AR content's primary axis and the ***relative vector*** that represents (1) the vector pointing from the AR content's *centroid* to the target vertex for the *point binding*, (2) the vector pointing from the AR content's *centroid* to the closest point on the target *edge* for the *line binding*, and (3) the normal vector of the target *face* for the *plane binding*.
- ***rSize*** represents a subset of the vector

$$[\frac{S_{x0}}{S_{x1}}, \frac{S_{x0}}{S_{y1}}, \frac{S_{x0}}{S_{z1}}, \frac{S_{y0}}{S_{x1}}, \frac{S_{y0}}{S_{y1}}, \frac{S_{y0}}{S_{z1}}, \frac{S_{z0}}{S_{x1}}, \frac{S_{z0}}{S_{y1}}, \frac{S_{z0}}{S_{z1}}]^T,$$

where (S_{x0}, S_{y0}, S_{z0}) represents the scale of the AR content and (S_{x1}, S_{y1}, S_{z1}) is the scale of the binding object.

During authoring, if a designer authors a *spatial association* of an AR content (e.g., an AR decoration is always placed at the corner of the TV), one or more *spatial relationship attributes* should keep unchanged across all the provided demonstrations. Now, we describe the fitting process to systematically interpret such invariance and extract the necessary information of the binding.

For all the AR content placements with the same *identity association*, more than one type of binding as well as more than one physical object's *geometric feature* may be feasible to represent the spatial relationship. For instance, the **Dis** of an AR vase may be fixed with respect to all the vertices (*point binding*) of a table and even some of the edges (*line binding*). Therefore, we develop an algorithm to extract the *geometric features* that consist of the most 'unchanged' *spatial relationship attributes* across all the demonstrations, namely, the ***target binding features***. In addition, it calculates the *spatial attributes* relative to the *target binding features* that are used to place the AR content, namely, the ***target binding values***. The algorithm takes all the M *geometric features* $[f_1, f_2, .., f_M]$ as well as the corresponding *spatial relationship attributes* in all the L virtual scenes as the inputs, where $f_m.\mathbf{Dis}[l]$ indicates the **Dis** attribute of the $m-th$ *geometric feature* in the $l-th$ scene. First, for each type of the *spatial relationship attributes*, we set a cut-off rule to extract the *geometric features* that the AR content may be bound. Typically, we extract the *geometric feature* f_m if $\max(|f_m.\mathbf{Attr}[i] - f_m.\mathbf{Attr}[j]|) < threshold_{\mathbf{Attr}}$, where $\mathbf{Attr} \in \{\mathbf{Dis}, \mathbf{rRot}, \mathbf{rSize}\}$, and $i, j \in \{1, 2, ..., L\}, i \neq j$ (the CutOffGeometricFeatures function in Figure 7). Considering that the **Dis** may also be affected by the scale of the binding object, we use both the normalized and unnormalized thresholds to evaluate the results. The thresholds are empirically set as $threshold_{\mathbf{Dis}} = 10cm$ (unnormalized), $0.15 \times scale\ radius$ (normalized), where the *scale radius* is half of the space diagonal of the bounding box, $threshold_{\mathbf{rRot}} = 20°$, $threshold_{\mathbf{rSize}} = 20\%\ per\ axis$, following [42, 65]. Although these works adopt the total difference (50% of the axis length) along the three dimensions as the **rSize** threshold, we use an average threshold (20%) for each individual axis to enforce a stricter restriction on aspect ratio variation. Then, within each of the three sets of the extracted *geometric features*, we compute the standard deviation of the corresponding *spatial relationship attributes* across all the demonstrations and select the *geometric feature* with the lowest standard deviation as the *target binding feature* respectively. Finally, we use the corresponding average value as the *target binding values* (the CalculateTargetBinding function in Figure 7).

Note that each of the three subsets may be empty due to the significant variations across the data and the lack of an apparent pattern in the designer's demonstrations, which prevents us from determining the *target binding features/values*. In this case, we check whether the *reference binding object*s exist. If so, the *reference binding object*s are utilized to constrain the DOF of the AR content with respect to the *primary binding object*s and to confirm the *target binding features*. Here, we demonstrate several examples when considering the *reference binding object*s: (1) an AR content keeps a fixed distance from a *primary binding object*'s *geometric features*, and faces along the vector pointing from the *primary binding object* to the *reference binding object* (Figure 6e, f), (2) an AR content fixed on a plane of the *primary binding object* tracks the motion of a *reference binding object* (Figure 6g), and (3) The scale of the AR content is affected by either the distance between the *reference binding object* and the *primary binding object* (Figure 6h-1) or the scale of the *reference binding object* (Figure 6h-2). When a *spatial relationship attribute* gets an empty set after running the CutOffGeometricFeatures function, we adopt the same function using the *geometric features* of the *reference binding object* $[f_1', f_2', ..., f_M']$ as the inputs. The values used for the *reference binding object* are in the *primary binding object*'s local coordinate system. When more than one *reference binding object* exists, we regard them as one object and average all the *spatial attributes* for the calculation.

Algorithm: Fit Target Binding

```
GetTargetBindingFeaturesAndValues ([f_1, f_2, ..., f_M], [f_1', f_2', ..., f_M'], attribute = Dis):
    targetBindingFeature ← null
    targetBindingValue ← null
    /* Check primary binding object */
    F ← CutOffGeometricFeatures ([f_1, f_2, ..., f_M], attribute)
    if F.Length ≠ 0 then
        targetBindingFeature, targetBindingValue ← CalculateTargetBinding (F, attribute)
    else
        /* Check reference binding object */
        if [f_1', f_2', ..., f_M'] ≠ null then
            F' ← CutOffGeometricFeatures [f_1', f_2', ..., f_M'], attribute)
            if F'. Length ≠ 0 then
                targetBindingFeature, targetBindingValue ← CalculateTargetBinding (F', attribute)
    /* Otherwise */
    if targetBindingFeature, targetBindingValue == null then
        targetBindingFeature, _ ← GetTargetBindingFeaturesAndValues ([f_1, f_2, ..., f_M], rRot)
        targetBindingValue ← random value
    return targetBindingFeature, targetBindingValue

CutOffGeometricFeatures ([f_1, f_2, ..., f_M], Attr):
    F ← array[]
    for m ← 1, M do
        currentGeoFeatureArr ← array[][]
        for i, j ← 1, L do
            currentGeoFeatureArr[i][j] ← abs(f_m.Attr[i] − f_m.Attr[j])
        if max (currentGeoFeatureArr) < threashold.Attr then
            Add f_m into F
    return F

CalculateTargetBinding (F, Attr):
    σ_min ← ∞; f̂, μ̂ ← null
    for f in F do
        μ = (1/L) Σ_{l=1}^{L} f.Attr[l]
        σ = sqrt((1/L) Σ_{l=1}^{L} (μ − f.Attr[l])^2)
        if σ < σ_min then
            σ_min ← σ; f̂ ← f; μ̂ ← μ
    return f̂, μ̂
```

Figure 7: The fitting algorithm that predicts the *target binding feature* and calculates the *target binding value* using the *geometric features* of the *identity association* object as the inputs. Here, we use the Dis attribute as an example. The other two *spatial relationship attributes*, rRot and rSize, are predicted using the same approach.

Suppose we still fail to get the *target binding feature* for a *spatial relationship attribute*. We use the non-empty *target binding feature* of other *spatial relationship attributes* as the *target binding feature* and set the *target binding value* as a random value. Overall the algorithm to fit the AR content placement model is illustrated as the GetTargetBindingFeaturesAndValues function in Figure 7.

4.5 Deployment of the AR Experience

In this section, we describe how to apply the *semantic adaptation model* during the deployment of the authored AR experience. Typically, given a physical scene, how ScalAR renders each AR content.

After ScalAR finishes fitting the *semantic adaptation model*, all the model parameters including (1) the involving object labels needed for each SVM-based *identity association* prediction model, (2) the *target binding features* and *target binding values* for each case of the activated *identity association*, and (3) the physical object labels involved in the *trigger-behavior binding*s are uploaded to the online server. When deploying an AR experience, a consumer first conducts the scanning process described in Section 4.1, or clicks the *load scene* button on the left-hand menu (Figure 4a-1) if the scene has already been scanned. Then, the AR consumer deploy the AR experience by clicking the *load app* button.

ScalAR follows the process of the loaded *semantic adaptation model* shown in Figure 3c to determine the placement of each AR content with respect to the current physical layout. Specifically, ScalAR first checks whether the system has a *semantic group* that contains exactly the same set of the physical objects as the current layout. If so, the model adopts the corresponding SVM model to predict the activated *identity association*. Otherwise, our system searches a *semantic group* where the authored *identity association*s include a subset of the physical objects present in the current scene and adopts the corresponding SVM model. For example, ScalAR tries to render an AR decoration in a scene with { *TV*, *table*, *bookshelf* }, but no *semantic group* matches it. However, ScalAR has a *semantic group* that involves { *TV*, *table*, *sofa*, *floor lamp* }. And during the authoring of this AR content, the designer has activated either *TV* or *table* as the *identity association* in different conditions. Here, our system uses the SVM model trained from this *semantic group* to predict *identity association*. On the contrary, if no such *semantic group* can be found, the system assumes the AR content is not suitable for the current scene and will not render it. This is because for *semantic-based* AR experiences, the semantic-level augmentation is primarily expressed by accurately placing a series of AR contents next to the relative physical objects within a physical environment. For instance, the AR contents in an AR cooking tutorial are tightly associated with the physical objects in a kitchen environment such as ranges, refrigerators, and countertops. Thus, these AR contents will not be reasonable to be rendered in a living room with sofas, TVs, and bookshelves. After predicting the activated *identity associations* in the current scene, ScalAR fetches the *target binding features* and *target binding values* to render the AR content. Note that since the *target binding values* are relative to the *target binding features*, the absolute *spatial attributes* to eventually render the AR content will be calculated accordingly. For instance, in one AR content placement model, an AR bird uses the upper-left *corner* of the TV (from the *primary binding object*) as the *target binding object* with a **Dis** value as the *target binding value*. Meanwhile, it contains the *centroid* of the sofa (from the *reference binding object*) as another *target binding object* with a **rRot** to constrain the rotation of the bird. In the deploying scene, given the *spatial attributes* of the TV and sofa, our system renders the AR bird to stand at the upper-left *corner* while facing towards the *centroid* of the sofa. To avoid collision between the AR content and the physical objects, we further adjust the content placement to exclude the space occupied by the physical objects. Afterwards, the *trigger-behavior binding*s are constructed if both the connected AR contents have been placed.

4.6 System Hardware and Software Setup

We built the AR-side system on HoloLens 2 [58], and the VR-side system on Oculus Quest 2 [66]. We adopted the bare-hand interaction supported by both devices as the user input modality. Both systems were developed with Unity3D (2019.4.16f1) and the user interfaces were supported by Microsoft Mixed Reality Toolkit (MRTK)[3]. The AR-VR data transfer (the online server mentioned in the paper) was enabled by Azure Storage service[4]. For the physical scene collection, the 2D object detection model was trained to detect 15 indoor furnishings including dishwasher, microwave, refrigerator, sink, ranges, countertops, cabinets, bed, blinds, wardrobe, sofa, TV, table, bookshelf, and chair for study and research use. For each object,

[3]https://github.com/microsoft/MixedRealityToolkit-Unity
[4]https://docs.microsoft.com/en-us/azure/storage

the researchers selected approximately 2000 images from ImageNet [14] and Epic-kitchens [13]. The training took 1.2 hours on Azure Custom Vision platform[5] with a precision of 95.1%, recall of 95.1%, and mAP of 90.6%. The IoU threshold was set to 0.5 for the 3D bounding box generation. The scene collection runs at 30 frame per second on a HoloLens 2, while the post-processing requires less than 10 seconds. Regarding the scene synthesis, the GA approach employed a population of 100 individuals, a mutation probability of 30, and a maximum number of 20 iterations, which takes 25 seconds on average for each collected scene. The values for the initial population were generated randomly and all the GA parameters were determined empirically in order to fulfill the criterion. To ensure a satisfactory fitting result of the *identity association* prediction model, we empirically synthesized 90 virtual scenes for each *semantic group* to train the SVM classifier, which was determined by the dimension of *spatial feature vectors* fed into the SVM training in our case [91]. The SVM classifier was trained and run on a local PC (Intel Core i7-9700K 3.6GB, 32GB RAM, NVIDIA RTX2080 GPU).

5 APPLICATION SCENARIOS

With ScalAR, AR designers are allowed to create *semantically adaptive* AR contents that can be deployed in multiple physical environments. Here, we showcase two application scenarios that leverage the flexibility of the VR authoring environment and the semantic-level adaptation enabled by our system.

5.1 AR Interior Decoration Collection

The advents of mobile computing lower the barrier to spatially and temporally distribute the workforce while keeping the productivity. ScalAR enables designers to create AR experiences in VR without leaving their local places. This capability raises substantial advantages of facilitating different designers to create scalable AR experiences under the same semantic topic, which significantly expedites the dissemination of their domain expertise. We address this enhancement through an AR interior decoration collection experience. Three interior designers with different design styles create a living room decoration AR application (Figure 8). With ScalAR, each designer is immersed in numerous living room environments with key elements such as sofa, TV, and window. They place the pre-designed virtual assets according to the scale, location, and overall layout of the environments. Then, AR consumers who locate in different places can experience the three designs to decorate their living rooms.

5.2 Interactive AR Pet

The role played by AR consumers in the AR experiences is essential, especially in AR games[6], and interactive AR experiences [48, 49]. ScalAR embraces the trigger-behavior metaphor to achieve dynamic reactions of the AR contents with respect to the semantic-sensitive human behaviors. Here, we showcase an interactive AR pet application authored with ScalAR (Figure 9). A designer places three *consumer assets* in the virtual living room environment and binds them to the physical objects respectively. Through trigger-behavior

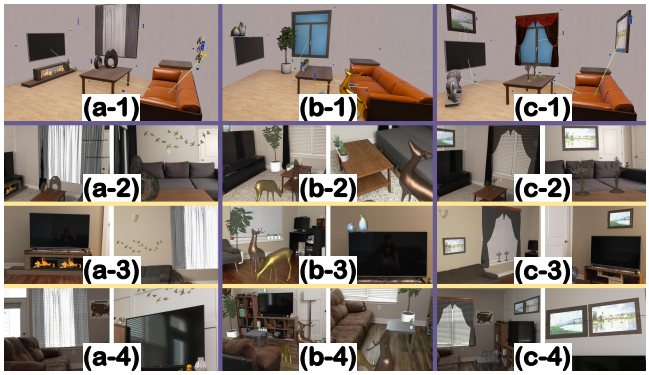

Figure 8: AR Interior Decoration Collection. (a-1) The modern style design authoring: a fireplace on the floor beneath the TV, a decoration on the table in front of the sofa, a decoration hung on the wall behind the sofa, and a curtain covering the window. (a-2 to a-4) The deployments in three living rooms. (b-1) The natural style design authoring: a bird decoration attached at the corner of the TV, a tall plant is placed on the floor beneath the window, a plant decoration is placed on the table in front of the sofa, a deer statue on the floor next to the sofa, and a bird decoration hung on the wall behind the sofa. (b-2 to b-4) The deployments in three living rooms. (c-1) The classic style design authoring: a painting hung on the wall behind the TV, another painting hung on the wall behind the sofa, a classic statue on the floor beneath the TV, a pair of candlesticks on the table in front of the sofa, and a classic curtain covering the window. (c-2 to c-4) The deployments in three living rooms.

connections, the designer defines three *semantic animations* of a pre-designed AR pet dog (sit, walk-towards, and jump-to) where the last two require the designer to bind the *target position* AR content in the environment using the same method as other AR contents adopt. Then, in different AR environments, AR consumers can enjoy the AR pet with interactive responses where not only the AR content behaviors are semantically represented, but the AR consumers can intuitively trigger the corresponding behaviors (e.g., the dog will walk to the consumer and jump onto the table as the consumer sits on the sofa).

6 USER STUDY

Participants. Our system consists of two major platforms. The AR-side application supports local AR consumers to collect the physical scenes and use the AR experiences, while the VR-side studio focuses on the authoring of the AR content behaviors. Thus, we invited two groups of users respectively, the *AR users* (AUs) and *VR users* (VUs), to evaluate our system. For the AR-side, our system targets any novice user who wants to consume the AR experiences. We recruited 16 users (11 males, 5 females, aging from 19 to 39). 14 out of 16 had used AR/VR applications (e.g., phone-based AR/VR games) while the rest two had heard about AR/VR. None of them had used our system before the user study. For the VR-side system, by introducing the immersive authoring and spatial programming

[5]https://azure.microsoft.com/en-us/services/cognitive-services/custom-vision-service
[6]https://www.asobostudio.com

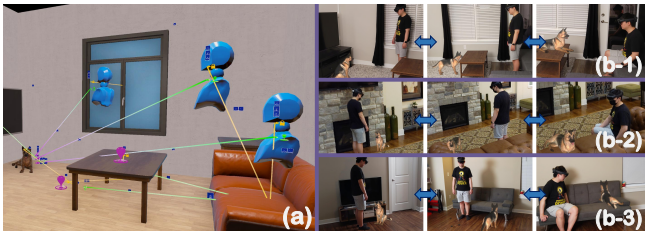

Figure 9: Interactive AR Pet. (a) The authoring details: the dog initially lies on the floor beneath the TV, (1) when the AR consumer stands in front of the window which is next to the TV, the dog sits up, (2) when the AR consumer stands in front of the sofa, the dog walks to the front of the table in front of the sofa, (3) when the AR consumer sits on the sofa, the dog jumps onto the table, while if no table exists, the dog jumps on the sofa next to the AR consumer. (b-1 to b-3) The deployments in three living rooms.

metaphors, ScalAR does not limit the target users to professional designers or programmers. Although the main purpose of the user study was not to compare the user experience between experts and novices, we invited both designers and novice users as the VUs to get more diverse feedback. 12 users (8 males, 4 females, aging from 23 to 31) were recruited as the VUs. Four of them had AR/VR development experience (e.g., Unity3D, ARCore, and Vuforia) for more than 12 months and self-identified as AR/VR designers, two had taken VR development courses using Unity3D and Oculus, and the rest six had experienced AR/VR games/applications such as VRChat and Pokemon Go. None of them had used our system.

User Study Sessions	Session 1	Session 2		
	Semantic Adaptation Model Evaluation	Scene Collection	VR Authoring	AR Deployment
User Groups	VUs	AUs	VUs	AUs
Completed Order of the Sessions	2nd	1st	3rd	4th

Table 1: User Study Arrangement. Considering the logistics of the tasks, the Session 2 was divided into three sub-sessions. We first conducted the scene collection sub-session with the AUs to create sample scenes used for the VR authoring sub-session. Then, the VUs completed the model evaluation session, and authored the AR experiences. Finally, the AUs consumed the authored experiences.

Procedure. We conducted a two-session user study to evaluate (1) the performance of the *semantic adaptation model* and (2) the overall usability of ScalAR. The arrangement of the entire user study is shown in Table 1. Since the second session focused on the end-to-end experience evaluation, it was split into three sub-sessions representing the three major parts of the ScalAR workflow.

The two AR-side sub-sessions, scene collection and AR deployment, were completed in two visits and took an hour in total. Each AU was paid with a $10 e-gift card. One of our researchers brought the AR device (HoloLens 2) to each AU's home and guided the AUs to complete the tasks respectively. For the scene collection, we first asked the AUs to complete a HoloLens built-in tutorial to get familiar with the freehand interactions. Then, the AUs used ScalAR to collect their physical scenes. We screen-recorded the AUs' operations and recorded the necessary data as described in the following section. The AR deployment sub-session happened after the VR-side user study. We asked each AU to deploy and use the AR experiences authored by the VUs and recorded the first-person view of the entire usage process. The two consecutive VR-side sub-sessions lasted for 2 hours, and each VU was paid with a $20 e-gift card. After a VU came, we first guided the VU to get familiar with the bare-hand drag-and-drop and click operations enabled by Oculus Quest 2. Then, the VU completed the tasks in VR and the first-person view was screen-recorded. After the AUs used the AR experience, we sent the AR-side usage videos together with the video of the VU's authoring process back to the VU to ask for the feedback in terms of the deployment performance. After each sub-session, both the AUs and VUs completed a Likert-type survey (scaled 1-5) regarding the user experience of the system features and the overall deployment performance. At the end of both sides of the user study, we conducted a conversation-type interview to collect the subjective feedback and asked the users to filled a standard System Usability Scale (SUS) questionnaire.

6.1 Session 1: *Semantic Adaptation Model* Accuracy Evaluation

6.1.1 Task Description. This study session aimed to assess the performance of the fitting algorithm and the feasibility of the *semantic adaptation model* with an AR decoration experience for *bedroom* environments. Only the VR studio of ScalAR was used by the VUs to generate data for the quantitative analysis. The scene collection was done by one of our researchers who collected a *bedroom* environment with nine physical objects: a bed, a window, a door, a wardrobe, a floor, and 4 walls, and ScalAR synthesized 120 virtual scenes, where 90 were included in the authoring system and 30 were used for the test. To comprehensively reflect the considerations of the *semantically adaptive* AR experience authoring, we designed six AR content placement tasks: (1) a cat sleeps at the corner of the bed mattress, (2) a curtain covers the entire window, (3) a painting is hung on the wall behind the bed, (4) a plant is placed on the floor below the window, (5) a mirror is hung next to the window when the window is facing towards the wardrobe, otherwise, on the wardrobe, and (6) a toy dog is placed on the bed when the bed is near the window, otherwise, on the window (Figure 10a). The first two tasks only required single *primary binding object*s, the next two required single *primary binding object*s and *reference binding object*s. The last two tasks required the VUs to activate different *primary binding object*s under different layouts. Specifically, the performance of the SVM model fitting was evaluated using the last two tasks, while the overall fitting accuracy was evaluated using all the six tasks. In order to get controllable quantitative data, one researcher explicitly described each task to the VUs during the study. For the tasks that may involve ambiguities such as the 'corner of the bed', the researcher further clarified them to ensure all the VUs had the same goals. The VUs authored and adjusted the AR contents based on their understanding of every virtual scene. After the VUs finished authoring, we fit the *semantic adaptation model*s and used the test scenes to evaluate the spatial accuracy of the AR content rendering. Note that the virtual scenes were synthesized from one

physical scene without *quantity variation*. Thus, for each VU, one single *semantic adaptation model* was fit per task.

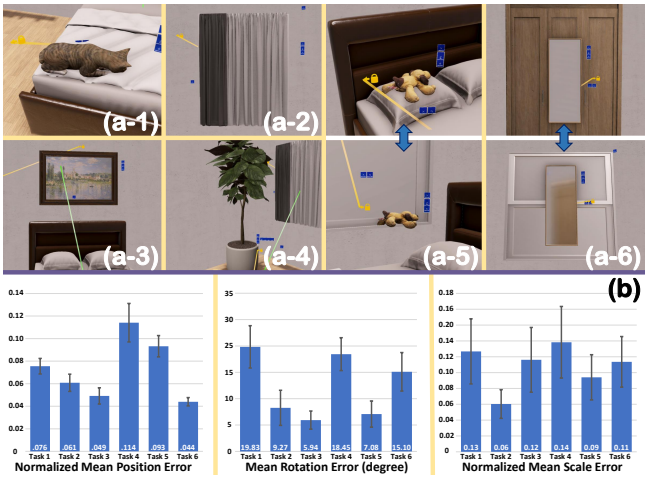

Figure 10: User Study Session 1. (a-1 to a-6) The six AR content placement tasks. (b) The results of the AR content placement accuracy.

In terms of the evaluation criteria, following commonly adopted processes for object manipulation in the virtual domain [7, 57], the researcher who described the tasks to the VUs manually placed the AR contents in all the 30 test scenes to provide the ground truths. We calculated the errors following the procedures in [57] for all the six tasks. For the last two tasks, we calculated the accuracy of the trained SVM models. In the previous works [20, 42, 84], the object manipulation is considered as successful if the errors are below certain thresholds. Here, we used the proposed thresholds: $threshold_{pos} = 0.15$, $threshold_{rot} = 18°$, $threshold_{scale} = 0.5$. Both the position error and scale error were normalized with half of the ground-truth object's diagonal as the unit.

6.1.2 Result and Discussion. All the 12 VUs successfully completed the six tasks. For each task, within the 90 virtual scenes provided, the VUs manually adjusted 8.63 scenes on average (SD=2.18) using 11.79 minutes (SD=2.91). The quantitative evaluation results are illustrated in Figure 10b. Specifically, the average position error was 0.072 ($SD = 0.027$) and the average rotation error was 12.61° ($SD = 5.98°$) across all the six tasks. Both the results were below the thresholds respectively, which indicated that the *semantic adaptation model* could accurately predict the AR content placement. For the position error, task 4 ($AVG = 0.114$, $SD = 0.017$) had worse performance than others due to the difficulty to accurately refer to the projection of window centroid on the floor. The high error value of task 5 ($AVG = 0.093$, $SD = 0.009$) might attribute to the different spaces that different VUs left between the mirror and window when the window was facing the wardrobe. The rotation errors in task 1 ($AVG = 19.83°$, $SD = 8.03°$), task 4 ($AVG = 18.45°$, $SD = 6.23°$), and task 6 ($AVG = 15.10°$, $SD = 7.31°$) increased because the VUs were free to rotate the AR content and not all the AR contents' axes were expected to have stable **rRot**s relative to the corresponding objects. The scale errors were normalized with respect to the ground truths and obtained as the sum of the three axes [42]. And all the scale errors were lower than the threshold. For task 2 ($AVG = 0.06$, $SD = 0.04$) and task 5 ($AVG = 0.094$, $SD = 0.06$), in which AR content sizes were highly aligned with the physical object size, our model could accurately estimate the scale change. Although the VUs could also coarsely adjust the AR content scale as the physical object scale changed in other tasks such as task 4 ($AVG = 0.14$, $SD = 0.05$), they found it hard to scale the AR content without precise measurement of the physical objects. The average scale error was 0.108 ($SD = 0.03$), showing that our model could reliably estimate the scale change trend. Meanwhile, the overall accuracy of the SVM classification result was 96.5%, where the accuracy results were 97.2% and 95.8% for task 5 and task 6 respectively, which proved the feasibility of the SVM training process. Overall, the *semantic adaptation model*s could precisely predict the activated *identity association*s and the AR contents' relative *spatial attributes* in different deploying scenes.

6.2 Session 2: System Usability Evaluation

6.2.1 Task Description. In this session, we evaluated the end-to-end user experience of the three major sections of the ScalAR workflow using an AR cooking tutorial application for *kitchen* environments. First, for the scene collection, we concentrated on whether an end-user could accurately and easily scan a physical scene using the AR-side system of ScalAR. The AUs were asked to scan their *kitchen*s using our system (Figure 11a-1). And our system recorded the identities and *spatial attributes* of the originally inferred physical objects. Then, the AUs were encouraged to adjust the 3D bounding boxes and add the missing objects. After the AUs were satisfied with the results, we recorded the same information of the layouts, where the post-adjustment data were used as the ground truths to evaluate the performance of the scene collection algorithm of ScalAR. Next, we randomly selected 10 collected scenes to synthesize the virtual scenes for the VR authoring sub-session. Typically, ScalAR generated 270 virtual scenes with three *semantic group*s based on the collected scenes. Then in VR, the VUs were asked to author the cooking tutorial with five AR contents: a slider-like UI, a video display, a plate with fish, a menu, and a *consumer asset*. For the *semantic binding*s (Figure 11a-2), (1) the slider-UI was attached to the microwave oven when the range was below the microwave oven, otherwise above the range, (2) the video display was attached to the countertop near the range, (3) the starting and finishing points of the plate animation were attached to the refrigerator and the sink, (4) the menu was attached to the refrigerator. In order to control the cooking tutorial, the VUs were asked to create trigger-behavior connections from the slider-UI to (1) the animation behavior of the plate and (2) the play-video behavior of the video display (marinate fish). The position trigger of the *consumer asset* was connected to the second play-video behavior of the display (fry fish). After the authoring, the *semantic adaptation model*s fit from the VUs' demonstrations were sent to the AR-side system. And the AUs deployed and used the AR experiences in a random order (Figure 11a-3).

6.2.2 Physical Scene Collection Accuracy Result. All the 16 AUs successfully scanned their *kitchen* environments using ScalAR. Two

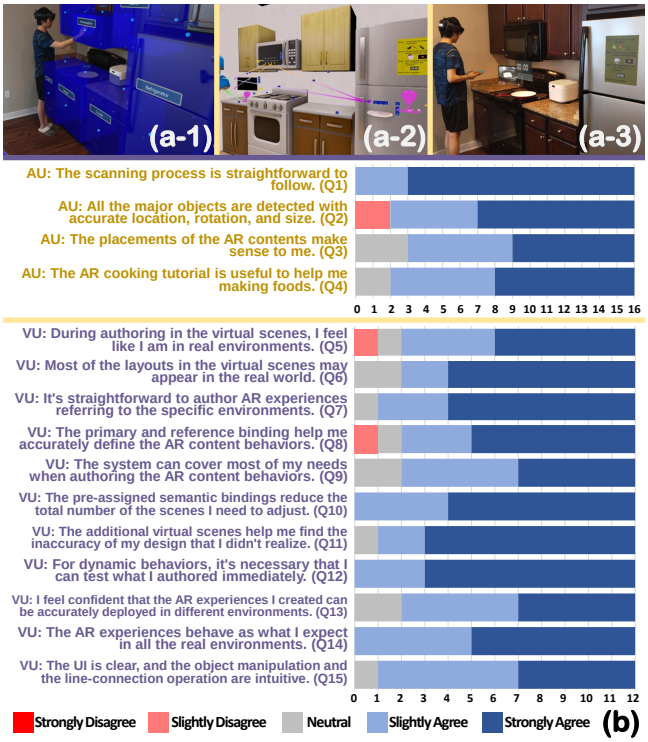

Figure 11: User Study Session 2. (a-1) The scene collection sub-session where the AUs can manually adjust the generated bounding boxes. (a-2) The VR authoring of the cooking tutorial task. (a-3) The AUs consume the authored AR experience. (b) The Likert-type questionnaire results for both the AUs and VUs.

AUs manually added one cabinet that was not detected by our system because the cabinet was hung above the refrigerator with a relative small size while the AUs did not notice it during scanning. Regarding the quantitative accuracy, we followed the criteria commonly used in the 3D object detection area [64, 79], and adopted the AU-adjusted layouts as the ground truths. Typically, we calculated the mean translation error (i.e., the error of the standard Euclidean distance) to be 0.046m (SD=0.13), the mean rotation error (i.e., the error of the geodesic distance) to be 1.87° (SD=1.04), and the mean scale error (i.e., the error of the logarithmic scaling factor) to be 0.09 (SD=0.14). Compared with the results using the general dataset in the above works, we achieved satisfactory results mainly because our system guided the AUs to observe the objects of interests with a closer distance and for a longer time to collect enough images with higher quality. Overall, the results indicated that the scene collection algorithm of ScalAR enabled the accurate scanning of the identity and spatial information of the physical environments for future VR authoring and AR deployment. The subjective feedback described later also supports the performance of the algorithm.

6.2.3 End-to-End User Experience Result. The Likert-type questionnaire results of both the AUs and VUs are shown in Figure 11b. Regarding the scene collection process, majority of the AUs were satisfied with the ease of the scanning process (Q1, AVG=4.81, SD=0.40) and the accuracy of the detected physical objects (Q2, AVG=4.31, SD=1.01). *"I just followed the instructions next to the menu to scan my room. No difficulties at all. (AU7)"* *"The detection was really accurate, it even separated the cabinets that were connected. (AU13)"* The two 'slightly disagree' votes were due to the missing detection of the cabinet mentioned in the previous section. Yet, the AUs were confident that ScalAR could detect those objects if they paid more attention to those cabinets and also welcomed the 'manually add object' function.

For the VR authoring experience, all the 12 VUs successfully completed the authoring using 31.75 minutes on average (SD=6.32). The VUs stopped at 15.83 virtual scenes (SD=2.59) to manually define and adjust the *semantic bindings*. The virtual scenes created by our system received complimentary feedback (Q6, AVG=4.50, SD=0.80). *"All of these virtual kitchens are very reasonable. Some kitchens don't have a microwave, some have a very large countertop, and some even don't have one. (VU5)"* Given the realistic and reasonable virtual environments, the VUs acknowledged the immersive feeling provided by our system (Q5, AVG=4.25, SD=0.97; Q7, AVG=4.58, SD=0.67). *"I clearly knew where to place those AR contents in the virtual kitchen, actually I could imagine walking inside a real kitchen. (VU11)"* Yet, one VU questioned that using the same CAD model for a physical object in every scene might reduce the feeling of the scene variation. This can be addressed by adding more available CAD models for each object and randomly picking one for each virtual scene. And we also discuss the detail-level diversity of each object in the next section. The majority of the VUs appreciated the additional virtual scenes for validating their designs (Q11: AVG=4.67, SD=0.65). *"I didn't realize that not all kitchens have a microwave until I saw that warning sign. (VU1)"* *"I adjusted the position and the size of the video display when I saw a very large countertop. It would be bad if a tiny video display was placed on a large table. (VU4)"* And our system essentially gave the VUs confidence in the accurate execution of the cooking tutorial (Q13: AVG=4.25, SD=0.79). We further asked about the detailed features to facilitate the authoring. Overall, the system covered most of the needs to author a *semantically adaptive* AR experience (Q9, AVG=4.25, SD=0.75). *"I think all the conditions have been considered in your system. I really like the idea of representing the user with a virtual model, and define this model's behaviors in different environments. (VU12)"* The design of the *primary/reference binding object* was welcomed by the VUs (Q8, AVG=4.33, SD=0.98). *"I really like the idea of reference binding. It's very common to hang a painting on the wall while using TV as reference for example. (VU11)"* But another VU with not much designing experience raised that *"If I use this system to design the AR app, the primary binding object may be enough because it is really easy to understand, but the reference is slightly complicated. (VU6)"* Following this comment, we need to consider different target user groups and provide additional assistance to non-expert users, which will be discussed in the next section. Supported by pre-assigning the *semantic bindings* to other scenes, about 6% of the provided scenes were manually adjusted by the VUs. And they felt relived that they did not have to adjust every virtual scene (Q10: AVG=4.67, SD=0.49). *"That the AR contents could follow the attached objects really reduce a lot of work. So, I could fully focus on those scenes with critical differences. (VU3)"* The *Test Mode* was receptively (Q12, AVG=4.75, SD=0.45). *"It's necessary to immediately try out my design. I could even try if the video display*

was too far from me when I put it next to the range. (VU3)" The VUs complimented the UI and intuitive operation enabled by immersive authoring (Q15, AVG=4.33, SD=0.65). *"Using lines to represent the binding relationship is really easy to understand. I think it aligns with a correct logic. (VU3)"*

The final deployment results received positive feedback from both groups of the users. After watching the deployment videos, most VUs acknowledged the success of the adaptive behaviors in different kitchens (Q14, AVG=4.58, SD=0.51). *"I feel more confident of using your system after seeing that my design works in so many rooms. (VU1)"* The AUs also enjoyed the semantic-level associations of the AR contents with the surroundings (Q3: AVG=4.25, SD=0.77; Q4: AVG=4.38, SD=0.72). *"I liked the animations in the tutorial most. Really easy to understand where to do what. (AU2)" "I really like that big video display on my countertop. It's convenient to watch tutorials when preparing and cooking foods. (AU9)"* The SUS survey results were 94 out of 100 (SD=3.46) for the AR-side and 87 out of 100 (SD=8.88) for the VR-side, which further illustrated the satisfactory usability of the entire workflow our system.

7 LIMITATIONS AND FUTURE WORKS

Object-level adaptation. Our system mainly addresses the adaptation capability regarding the variations of the indoor layouts. While using the 3D bounding boxes as the spatial references to place AR contents has been proved effective during the user study, some VUs raised that the adaptation performance may be unsatisfactory owing to the complicated and diversified geometrical configurations of the physical objects (e.g., different types of sofas with different sizes and locations of the armrest). Integrating object-level semantic understanding techniques [54, 77] could be a solution. By semantically segmenting the dense mesh of the physical objects, some functional planes can be extracted as the additional spatial references (e.g., sittable and leanable planes of a chair). We could allow the designers to select these features as the *identity association* resources in VR, and incorporate them with the current *semantic adaptation model* to further improve the system performance.

Feedback from the deployment results. As addressed in Section 4.5, if a new physical scene does not have the required objects for the classification process, we skip rendering the corresponding AR content because the essential reference environment affordance may not exist. VU10 raised that by adding a feedback from AR to VR, we could inform the designer of the failure cases by showing these scenes in the VR authoring environment. This way, the designer can manually re-define the *semantic binding*s and keep the design updated even when new physical scenes are collected and uploaded to the authoring environment. Leveraging the online server we currently adopt, granting the AR-side system with such capabilities could be one improvement direction.

Dynamic suggestions during authoring. ScalAR fits the *semantic adaptation model* after a designer validates the design in all the virtual scenes, and the user study has proved the accuracy of the fitting results. Yet, VU6 and VU2 suggested after adjusting the AR contents in several scenes, the system could provide some suggestions in terms of the potential *spatial association*s for the designer to explicitly inform our system. In addition, while the feature of pre-assigning the initial *semantic binding* to the other virtual scenes received complimentary feedback, VU6 mentioned that those associations were constantly set and sometimes may confuse designers. Adding an adaptive statistical classification step such as AdaBoost [21] and imitation learning [37] to the authoring process would be a possible direction to address these needs. By dynamically inferring the designer's adjustments, the system can provide visual hints such as highlighted points/lines/planes on the object to guide the designer to constrain the expected *spatial association*s of the AR contents and can adaptively adjust the pre-assigned *semantic binding*s to further reduce the workload.

Different levels of immersion in VR. We received complimentary feedback regarding the immersiveness of the authoring experience during the user study. Some VUs raised that *"Because I was standing at the center of the room, it was difficult for me to see all the virtual contents when I slid the cursor. (VU3)"* This concern has also been addressed in prior immersive authoring works [89]. Leveraging the versatility of VR, we could elevate the designer's head position during the *Traverse Mode* to provide a larger field of view. Meanwhile, different view modes other than the *scene cursor* approach may be introduced in ScalAR. For instance, (1) allowing designers to move the virtual scene instead of walking in the environment during authoring and (2) arranging all the virtual scenes in a matrix pattern, and provide the designers with a bird's eye view for more rapid validation of the design. In this manner, ScalAR balances the immersiveness and the abstraction by leveraging the limitless mobility enabled by VR.

More adaptation resources for expert designers. The provided system features were proved to cover most needs to author *semantically adaptive* AR content behaviors as addressed in the user study. Some VUs who identified as AR/VR developers recommended providing more capabilities to expert users following the current system design. For instance, we consider AR consumers' semantic-sensitive spatial movements such as 'sitting on the sofa' as the resources because the authoring logic is highly similar to the *semantic binding*s to the physical objects. VU10 mentioned ScalAR should provide more input modalities such as 'look at the TV' and 'open the refrigerator' as the resources since they also involve semantic implications with the surroundings. VU1 further suggested utilizing AR consumers' interactions with the physical entities to trigger AR content behaviors as addressed in [63, 88]. Therefore, our research opens up future directions to expand the semantic-level blends between the AR and physical domains while maintaining the adaptation capability in different environments through human authoring.

8 CONCLUSION

In this work, we presented ScalAR, an AR/VR integrated authoring system empowering designers to build AR experiences that could semantically adapt to different deploying environments. We first discussed the characteristics of *semantically adaptive* AR experiences to characterize the two-level associations between the AR contents and the physical environments, namely *identity association* and *spatial association*, as well as the two-level scene variations, *spatial variation* and *quantity variation*. Then, we proposed an authoring workflow with three consecutive sections: (1) In AR, potential consumers collect the target scenes relevant to the authoring topic. (2)

In VR, a designer defines and validates the two-level associations of the AR contents across plentiful virtual scenes through immersive authoring. We distill a *semantic adaptation model* through a decision-tree-based fitting algorithm using the designer's demonstrations. (3) Back in AR, the model maps the target physical scene to the corresponding semantic-level relationships of the AR contents. We further demonstrated two application scenarios, namely an AR interior decoration collection and an interactive AR pet, that exploited the capability of our system. With the user study, we first evaluated the performance of the fitting algorithm by investigating the quantitative behaviors of the *semantic adaptation model* fit from the users' demonstrations and elaborated the overall usability of our system through the complimentary qualitative user feedback. To sum up, we believe that ScalAR exposes a promising perspective of leveraging the immersiveness and flexibility of VR to create semantic-sensitive AR experiences that can be extensively deployed in diverse physical environments.

ACKNOWLEDGMENTS

We thank the reviewers for their invaluable feedback. This work is partially supported by the NSF under grants Future of Work at the Human Technology Frontier (FW-HTF) 1839971. We also acknowledge the Feddersen Chair Funds. Any opinions, findings, and conclusions expressed in this material are those of the authors and do not necessarily reflect the views of the funding agency.

REFERENCES

[1] ARCore 2021. ARCore. https://developers.google.com/ar.
[2] ARKit 2021. ARKit. https://developer.apple.com/augmented-reality/arkit.
[3] Eric A Bier. 1990. Snap-dragging in three dimensions. *ACM SIGGRAPH Computer Graphics* 24, 2 (1990), 193–204.
[4] Mark Billinghurst, Adrian Clark, and Gun Lee. 2015. A survey of augmented reality. (2015).
[5] Yuanzhi Cao, Xun Qian, Tianyi Wang, Rachel Lee, Ke Huo, and Karthik Ramani. 2020. An Exploratory Study of Augmented Reality Presence for Tutoring Machine Tasks. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. 1–13.
[6] Yuanzhi Cao, Tianyi Wang, Xun Qian, Pawan S Rao, Manav Wadhawan, Ke Huo, and Karthik Ramani. 2019. GhostAR: A time-space editor for embodied authoring of human-robot collaborative task with augmented reality. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology*. 521–534.
[7] Han Joo Chae, Jeong-in Hwang, and Jinwook Seo. 2018. Wall-based space manipulation technique for efficient placement of distant objects in augmented reality. In *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology*. 45–52.
[8] Lung-Pan Cheng, Eyal Ofek, Christian Holz, and Andrew D Wilson. 2019. Vroamer: generating on-the-fly VR experiences while walking inside large, unknown real-world building environments. In *2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*. IEEE, 359–366.
[9] Yifei Cheng, Yukang Yan, Xin Yi, Yuanchun Shi, and David Lindlbauer. 2021. SemanticAdapt: Optimization-based Adaptation of Mixed Reality Layouts Leveraging Virtual-Physical Semantic Connections. In *The 34th Annual ACM Symposium on User Interface Software and Technology*. 282–297.
[10] Hung-Lin Chi, Shih-Chung Kang, and Xiangyu Wang. 2013. Research trends and opportunities of augmented reality applications in architecture, engineering, and construction. *Automation in construction* 33 (2013), 116–122.
[11] Subramanian Chidambaram, Hank Huang, Fengming He, Xun Qian, Ana M Villanueva, Thomas S Redick, Wolfgang Stuerzlinger, and Karthik Ramani. 2021. ProcessAR: An Augmented Reality-Based Tool to Create In-Situ Procedural 2D/3D AR Instructions. In *Proceedings of the Designing Interactive Systems Conference*.
[12] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. 2017. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 5828–5839.
[13] Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Sanja Fidler, Antonino Furnari, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, et al. 2018. Scaling egocentric vision: The epic-kitchens dataset. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 720–736.
[14] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*. Ieee, 248–255.
[15] Zhi-Chao Dong, Wenming Wu, Zenghao Xu, Qi Sun, Guanjie Yuan, Ligang Liu, and Xiao-Ming Fu. 2021. Tailored Reality: Perception-aware Scene Restructuring for Adaptive VR Navigation. *ACM Transactions on Graphics (TOG)* 40, 5 (2021), 1–15.
[16] Ruofei Du, Eric Turner, Maksym Dzitsiuk, Luca Prasso, Ivo Duarte, Jason Dourgarian, Joao Afonso, Jose Pascoal, Josh Gladstone, Nuno Cruces, et al. 2020. DepthLab: Real-Time 3D Interaction With Depth Maps for Mobile Augmented Reality. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology*. 829–843.
[17] 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 of the 43rd Graphics Interface Conference*. 156–162.
[18] Barrett Ens, Eyal Ofek, Neil Bruce, and Pourang Irani. 2015. Spatial constancy of surface-embedded layouts across multiple environments. In *Proceedings of the 3rd ACM Symposium on Spatial User Interaction*. 65–68.
[19] Mark Fiala. 2005. ARTag, a fiducial marker system using digital techniques. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, Vol. 2. IEEE, 590–596.
[20] Scott Frees, G Drew Kessler, and Edwin Kay. 2007. PRISM interaction for enhancing control in immersive virtual environments. *ACM Transactions on Computer-Human Interaction (TOCHI)* 14, 1 (2007), 2–es.
[21] Yoav Freund. 2001. An adaptive version of the boost by majority algorithm. *Machine Learning* 43, 3 (2001), 293–318.
[22] Qiang Fu, Xiaowu Chen, Xiaotian Wang, Sijia Wen, Bin Zhou, and Hongbo Fu. 2017. Adaptive synthesis of indoor scenes via activity-associated object relation graphs. *ACM Transactions on Graphics (TOG)* 36, 6 (2017), 1–13.
[23] Yasutaka Furukawa, Brian Curless, Steven M Seitz, and Richard Szeliski. 2009. Manhattan-world stereo. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 1422–1429.
[24] Ran Gal, Lior Shapira, Eyal Ofek, and Pushmeet Kohli. 2014. FLARE: Fast layout for augmented reality applications. In *2014 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. IEEE, 207–212.
[25] Paul Grimm, Michael Haller, Volker Paelke, Silvan Reinhold, Christian Reimann, and R Zauner. 2002. AMIRE-authoring mixed reality. In *The First IEEE International Workshop Agumented Reality Toolkit,*. IEEE, 2–pp.
[26] Saurabh Gupta, Pablo Arbeláez, Ross Girshick, and Jitendra Malik. 2015. Indoor scene understanding with rgb-d images: Bottom-up segmentation, object detection and semantic segmentation. *International Journal of Computer Vision* 112, 2 (2015), 133–149.
[27] Taejin Ha, Woontack Woo, Youngho Lee, Junhun Lee, Jeha Ryu, Hankyun Choi, and Kwanheng Lee. 2010. ARtalet: tangible user interface based immersive augmented reality authoring tool for Digilog book. In *2010 International Symposium on Ubiquitous Virtual Reality*. IEEE, 40–43.
[28] Lei Han, Tian Zheng, Yinheng Zhu, Lan Xu, and Lu Fang. 2020. Live semantic 3d perception for immersive augmented reality. *IEEE Transactions on Visualization and Computer Graphics* 26, 5 (2020), 2012–2022.
[29] Jeremy Hartmann, Christian Holz, Eyal Ofek, and Andrew D Wilson. 2019. Realitycheck: Blending virtual environments with situated physical reality. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. 1–12.
[30] Stefan Hartmann, Björn Krüger, and Reinhard Klein. 2015. Content-aware retargeting of discrete element layouts. (2015).
[31] Devamardeep Hayatpur, Seongkook Heo, Haijun Xia, Wolfgang Stuerzlinger, and Daniel Wigdor. 2019. Plane, ray, and point: Enabling precise spatial manipulations with shape constraints. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology*. 1185–1195.
[32] Alexander Hermans, Georgios Floros, and Bastian Leibe. 2014. Dense 3d semantic mapping of indoor scenes from rgb-d images. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2631–2638.
[33] Dominik Herr, Jan Reinhardt, Guido Reina, Robert Krüger, Rafael V Ferrari, and Thomas Ertl. 2018. Immersive modular factory layout planning using augmented reality. *Procedia CIRP* 72 (2018), 1112–1117.
[34] Horizon Workrooms 2021. Horizon Workrooms for VR Remote Collaboration | Meta. https://about.fb.com/news/2021/08/introducing-horizon-workrooms-remote-collaboration-reimagined/.
[35] Ke Huo, Yuanzhi Cao, Sang Ho Yoon, Zhuangying Xu, Guiming Chen, and Karthik Ramani. 2018. Scenariot: Spatially mapping smart things within augmented reality scenes. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. 1–13.
[36] Ke Huo and Karthik Ramani. 2017. Window-shaping: 3d design ideation by creating on, borrowing from, and looking at the physical world. In *Proceedings of the Eleventh International Conference on Tangible, Embedded, and Embodied*

Interaction. 37–45.
[37] Ahmed Hussein, Mohamed Medhat Gaber, Eyad Elyan, and Chrisina Jayne. 2017. Imitation learning: A survey of learning methods. *ACM Computing Surveys (CSUR)* 50, 2 (2017), 1–35.
[38] Hans-Christian Jetter, Roman Rädle, Tiare Feuchtner, Christoph Anthes, Judith Friedl, and Clemens Nylandsted Klokmose. 2020. " In VR, everything is possible!": Sketching and Simulating Spatially-Aware Interactive Spaces in Virtual Reality. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. 1–16.
[39] Dongsik Jo and Gerard Jounghyun Kim. 2016. ARIoT: scalable augmented reality framework for interacting with Internet of Things appliances everywhere. *IEEE Transactions on Consumer Electronics* 62, 3 (2016), 334–340.
[40] Peter Kán and Hannes Kaufmann. 2017. Automated interior design using a genetic algorithm. In *Proceedings of the 23rd ACM Symposium on Virtual Reality Software and Technology*. 1–10.
[41] Peter Kán and Hannes Kaufmann. 2018. Automatic furniture arrangement using greedy cost minimization. In *2018 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*. IEEE, 491–498.
[42] Max Krichenbauer, Goshiro Yamamoto, Takafumi Taketom, Christian Sandor, and Hirokazu Kato. 2017. Augmented reality versus virtual reality for 3d object manipulation. *IEEE Transactions on Visualization and Computer Graphics* 24, 2 (2017), 1038–1048.
[43] Kin Chung Kwan and Hongbo Fu. 2019. Mobi3dsketch: 3d sketching in mobile ar. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. 1–11.
[44] Yining Lang, Wei Liang, and Lap-Fai Yu. 2019. Virtual agent positioning driven by scene semantics in mixed reality. In *2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*. IEEE, 767–775.
[45] Tobias Langlotz, Stefan Mooslechner, Stefanie Zollmann, Claus Degendorfer, Gerhard Reitmayr, and Dieter Schmalstieg. 2012. Sketching up the world: in situ authoring for mobile augmented reality. *Personal and Ubiquitous Computing* 16, 6 (2012), 623–630.
[46] Florian Ledermann and Dieter Schmalstieg. 2005. APRIL: a high-level framework for creating augmented reality presentations. In *Proceedings of the IEEE Virtual Reality*. IEEE, 187–194.
[47] Gun A Lee, Gerard J Kim, and Mark Billinghurst. 2005. Immersive authoring: What you experience is what you get (wyxiwyg). *Commun. ACM* 48, 7 (2005), 76–81.
[48] Germán Leiva, Jens Emil Grønbæk, Clemens Nylandsted Klokmose, Cuong Nguyen, Rubaiat Habib Kazi, and Paul Asente. 2021. Rapido: Prototyping Interactive AR Experiences through Programming by Demonstration. In *The 34th Annual ACM Symposium on User Interface Software and Technology*. 626–637.
[49] Germán Leiva, Cuong Nguyen, Rubaiat Habib Kazi, and Paul Asente. 2020. Pronto: Rapid augmented reality video prototyping using sketches and enaction. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. 1–13.
[50] Wei Liang, Xinzhe Yu, Rawan Alghofaili, Yining Lang, and Lap-Fai Yu. 2021. Scene-Aware Behavior Synthesis for Virtual Pets in Mixed Reality. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. 1–12.
[51] Chuan-en Lin, Ta Ying Cheng, and Xiaojuan Ma. 2020. ARchitect: Building Interactive Virtual Experiences from Physical Affordances by Bringing Human-in-the-Loop. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. 1–13.
[52] 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.
[53] David Lindlbauer and Andy D Wilson. 2018. Remixed reality: Manipulating space and time in augmented reality. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. 1–13.
[54] Jingyang Liu. 2021. Semantic Mapping: A Semantics-based Approach to Virtual Content Placement for Immersive Environments. In *2021 17th International Conference on Intelligent Environments (IE)*. IEEE, 1–8.
[55] Blair MacIntyre, Maribeth Gandy, Steven Dow, and Jay David Bolter. 2004. DART: a toolkit for rapid design exploration of augmented reality experiences. In *Proceedings of the 17th Annual ACM Symposium on User Interface Software and Technology*. 197–206.
[56] Sebastian Marwecki and Patrick Baudisch. 2018. Scenograph: Fitting real-walking vr experiences into various tracking volumes. In *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology*. 511–520.
[57] Daniel Mendes, Filipe Relvas, Alfredo Ferreira, and Joaquim Jorge. 2016. The benefits of dof separation in mid-air 3d object manipulation. In *Proceedings of the 22nd ACM Conference on Virtual Reality Software and Technology*. 261–268.
[58] Microsoft HoloLens 2021. Microsoft HoloLens | Mixed Reality Technology for Business. https://www.microsoft.com/en-us/hololens.
[59] Mark Mine. 1995. ISAAC: A virtual environment tool for the interactive construction of virtual worlds. (1995).
[60] MRTK Spatial Awareness 2021. Spatial Awareness - Microsoft Mixed Reality Toolkit. hhttps://docs.microsoft.com/en-us/windows/mixed-reality/mrtk-unity/features/spatial-awareness/spatial-awareness-getting-started?view=mrtkunity-2021-05.
[61] 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 *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. 1–12.
[62] Andrew YC Nee, SK Ong, George Chryssolouris, and Dimitris Mourtzis. 2012. Augmented reality applications in design and manufacturing. *CIRP annals* 61, 2 (2012), 657–679.
[63] Gary Ng, Joon Gi Shin, Alexander Plopski, Christian Sandor, and Daniel Saakes. 2018. Situated game level editing in augmented reality. In *Proceedings of the Twelfth International Conference on Tangible, Embedded, and Embodied Interaction*. 409–418.
[64] Yinyu Nie, Xiaoguang Han, Shihui Guo, Yujian Zheng, Jian Chang, and Jian Jun Zhang. 2020. Total3dunderstanding: Joint layout, object pose and mesh reconstruction for indoor scenes from a single image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 55–64.
[65] Benjamin Nuernberger, Eyal Ofek, Hrvoje Benko, and Andrew D Wilson. 2016. Snaptoreality: Aligning augmented reality to the real world. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. 1233–1244.
[66] Oculus Quest 2 2021. Oculus Quest 2: Our Most Advanced New All-in-One VR Headset | Oculus. https://www.oculus.com/quest-2.
[67] Oculus Store 2021. Oculus Store: VR Games, Apps, and More | Oculus. https://www.oculus.com/experiences/quest.
[68] Viet Toan Phan and Seung Yeon Choo. 2010. Interior design in augmented reality environment. *International Journal of Computer Applications* 5, 5 (2010), 16–21.
[69] Wayne Piekarski and Bruce Thomas. 2002. ARQuake: the outdoor augmented reality gaming system. *Commun. ACM* 45, 1 (2002), 36–38.
[70] Arnaud Prouzeau, Yuchen Wang, Barrett Ens, Wesley Willett, and Tim Dwyer. 2020. Corsican twin: Authoring in situ augmented reality visualisations in virtual reality. In *Proceedings of the International Conference on Advanced Visual Interfaces*. 1–9.
[71] Francisco J Romero-Ramirez, Rafael Muñoz-Salinas, and Rafael Medina-Carnicer. 2018. Speeded up detection of squared fiducial markers. *Image and vision Computing* 76 (2018), 38–47.
[72] Rafael Sacks, Amotz Perlman, and Ronen Barak. 2013. Construction safety training using immersive virtual reality. *Construction Management and Economics* 31, 9 (2013), 1005–1017.
[73] Hartmut Seichter, Julian Looser, and Mark Billinghurst. 2008. ComposAR: An intuitive tool for authoring AR applications. In *2008 7th IEEE/ACM International Symposium on Mixed and Augmented Reality*. IEEE, 177–178.
[74] Tobias Sielhorst, Marco Feuerstein, and Nassir Navab. 2008. Advanced medical displays: A literature review of augmented reality. *Journal of Display Technology* 4, 4 (2008), 451–467.
[75] Spatial 2021. Spatial - Metaverse Spaces That Bring Us Together. https://spatial.io.
[76] Misha Sra, Sergio Garrido-Jurado, and Pattie Maes. 2017. Oasis: Procedurally generated social virtual spaces from 3d scanned real spaces. *IEEE Transactions on Visualization and Computer Graphics* 24, 12 (2017), 3174–3187.
[77] Tomu Tahara, Takashi Seno, Gaku Narita, and Tomoya Ishikawa. 2020. Retargetable AR: Context-aware Augmented Reality in Indoor Scenes based on 3D Scene Graph. In *2020 IEEE International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct)*. IEEE, 249–255.
[78] Tilt Brush 2021. Tilt Brush by Google. https://www.tiltbrush.com.
[79] Shubham Tulsiani, Saurabh Gupta, David F Fouhey, Alexei A Efros, and Jitendra Malik. 2018. Factoring shape, pose, and layout from the 2d image of a 3d scene. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 302–310.
[80] Unity3D 2021. Unity3D. https://unity.com.
[81] Unreal 2021. Unreal. https://www.unrealengine.com.
[82] Julien Valentin, Vibhav Vineet, Ming-Ming Cheng, David Kim, Jamie Shotton, Pushmeet Kohli, Matthias Nießner, Antonio Criminisi, Shahram Izadi, and Philip Torr. 2015. Semanticpaint: Interactive 3d labeling and learning at your fingertips. *ACM Transactions on Graphics (TOG)* 34, 5 (2015), 1–17.
[83] Khrystyna Vasylevska, Hannes Kaufmann, Mark Bolas, and Evan A Suma. 2013. Flexible spaces: Dynamic layout generation for infinite walking in virtual environments. In *2013 IEEE Symposium on 3D User Interfaces (3DUI)*. IEEE, 39–42.
[84] Manuel Veit, Antonio Capobianco, and Dominique Bechmann. 2009. Influence of degrees of freedom's manipulation on performances during orientation tasks in virtual reality environments. In *Proceedings of the 16th ACM Symposium on Virtual Reality Software and Technology*. 51–58.
[85] Vuforia 2021. Vuforia. https://www.ptc.com/en/products/vuforia.
[86] Daniel Wagner and Dieter Schmalstieg. 2007. Artoolkitplus for pose tracking on mobile devices. (2007).
[87] Tianyi Wang, Xun Qian, Fengming He, Xiyun Hu, Yuanzhi Cao, and Karthik Ramani. 2021. GesturAR: An Authoring System for Creating Freehand Interactive Augmented Reality Applications. In *The 34th Annual ACM Symposium on User*

Interface Software and Technology. 552–567.

[88] Tianyi Wang, Xun Qian, Fengming He, Xiyun Hu, Ke Huo, Yuanzhi Cao, and Karthik Ramani. 2020. CAPturAR: An Augmented Reality Tool for Authoring Human-Involved Context-Aware Applications. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology*. 328–341.

[89] Zeyu Wang, Cuong Nguyen, Paul Asente, and Julie Dorsey. 2021. DistanciAR: Authoring Site-Specific Augmented Reality Experiences for Remote Environments. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. 1–12.

[90] Hsin-Kai Wu, Silvia Wen-Yu Lee, Hsin-Yi Chang, and Jyh-Chong Liang. 2013. Current status, opportunities and challenges of augmented reality in education. *Computers & education* 62 (2013), 41–49.

[91] Ting-Fan Wu, Chih-Jen Lin, and Ruby C Weng. 2004. Probability estimates for multi-class classification by pairwise coupling. *Journal of Machine Learning Research* 5, Aug (2004), 975–1005.

[92] Haijun Xia, Sebastian Herscher, Ken Perlin, and Daniel Wigdor. 2018. Spacetime: Enabling fluid individual and collaborative editing in virtual reality. In *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology*. 853–866.

[93] Hui Ye, Kin Chung Kwan, Wanchao Su, and Hongbo Fu. 2020. ARAnimator: in-situ character animation in mobile AR with user-defined motion gestures. *ACM Transactions on Graphics (TOG)* 39, 4 (2020), 83–1.

[94] Lap Fai Yu, Sai Kit Yeung, Chi Keung Tang, Demetri Terzopoulos, Tony F Chan, and Stanley J Osher. 2011. Make it home: automatic optimization of furniture arrangement. *ACM Transactions on Graphics (TOG)-Proceedings of ACM SIGGRAPH 2011, v. 30,(4), July 2011, article no. 86* 30, 4 (2011).

[95] Ya-Ting Yue, Yong-Liang Yang, Gang Ren, and Wenping Wang. 2017. SceneCtrl: Mixed reality enhancement via efficient scene editing. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology*. 427–436.

[96] Hui Zhang. 2017. Head-mounted display-based intuitive virtual reality training system for the mining industry. *International Journal of Mining Science and Technology* 27, 4 (2017), 717–722.

[97] 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.

[98] Egui Zhu, Arash Hadadgar, Italo Masiello, and Nabil Zary. 2014. Augmented reality in healthcare education: an integrative review. *PeerJ* 2 (2014), e469.