ARchitect: Building Interactive Virtual Experiences from Physical Affordances by Bringing Human-in-the-Loop

Chuan-en Lin*, **Ta Ying Cheng***, **Xiaojuan Ma**
Hong Kong University of Science and Technology
{clinaf, ttchengab}@ust.hk, mxj@cse.ust.hk

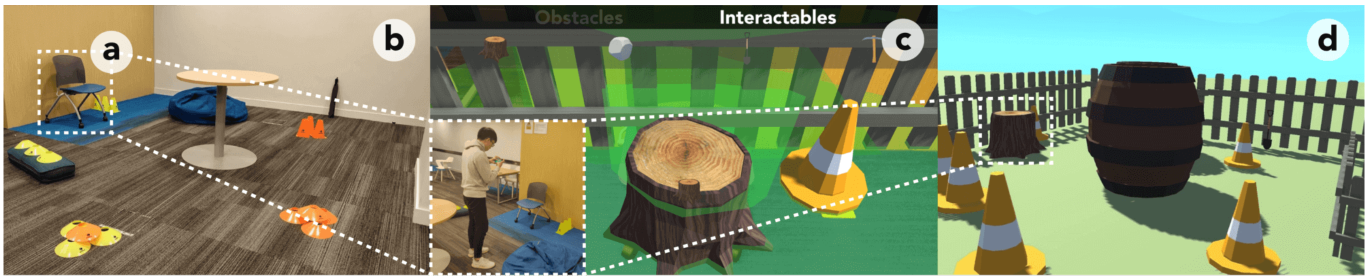


Figure 1. (a) ARchitect allows physical objects to be mapped to virtual proxies offering matching affordances (e.g. both "chair" and "tree stump" afford a "sitting" interaction). A physical scene (b) may be translated using ARchitect's user interface (c) to an interactive virtual experience (d).

ABSTRACT
Automatic generation of Virtual Reality (VR) worlds which adapt to physical environments have been proposed to enable safe walking in VR. However, such techniques mainly focus on the *avoidance* of physical objects as obstacles and overlook their interaction affordances as passive haptics. Current VR experiences involving interaction with physical objects in surroundings still require verbal instruction from an assisting partner. We present *ARchitect*, a proof-of-concept prototype that allows flexible customization of a VR experience with human-in-the-loop. *ARchitect* brings in an assistant to map physical objects to virtual proxies of matching affordances using Augmented Reality (AR). In a within-subjects study (9 user pairs) comparing *ARchitect* to a baseline condition, assistants and players experienced decreased workload and players showed increased VR presence and trust in the assistant. Finally, we defined design guidelines of *ARchitect* for future designers and implemented three demonstrative experiences.

Author Keywords
ARchitect; virtual reality; affordance; passive haptics; asymmetric

CCS Concepts
•**Human-centered computing** → **Virtual reality;** Graphical user interfaces;

*Denotes equal contribution

CHI'20, April 25–30, 2020, Honolulu, HI, USA

ISBN 978-1-4503-6708-0/20/04...$15.00
DOI: https://doi.org/10.1145/3313831.3376614

INTRODUCTION
Modern commercial Virtual Reality (VR) systems (e.g. Oculus Rift and HTC Vive) commit to achieving complete immersion in the virtual world yet come at the cost of losing spatial awareness of physical surroundings. Since virtual and physical realities intrinsically overlap in space, movement and actions of users in the physical environment directly affect their experiences in the virtual world. For example, bumping into physical obstacles may be dangerous and contacting objects in the physical world that are not rendered in VR may lead to Breaks in Presence (BIP) [19] from confusion.

Recent work has attempted to address this dilemma by deploying 3D reconstruction techniques to construct a virtual world that corresponds to the physical environment. This is typically done through scanning the physical environment using an RGB-D camera either a priori [41] or on-the-fly [8] to produce a depth map and procedurally generating a virtual scene with similar geometry. However, depth maps are often incomplete (filled with gaps) and lack semantics (unable to differentiate between interactable objects such as chairs and non-interactable objects such as walls and obstacles). Hence, such approaches have mainly been used for mapping coarse geometries not intended for interaction, e.g. horizontal planes as walkable areas, and overlook the blending of physical objects as passive haptics in the virtual space. Nonetheless, prior research has found that receiving haptic feedback from touching physical objects registered with virtual proxies increases presence in the virtual world [26]. Therefore, there have also been efforts for the automatic mapping of physical objects to virtual counterparts from an existing database of 3D models. Unfortunately, state-of-the-art methods barely achieve satisfactory accuracy [4], are too time-consuming and computationally expensive to run on embedded devices [40], and do not generalize well to an immense number of unannotated object classes in-the-wild [11]. More importantly, current mapping methods primarily concern 1-to-1 mapping using models of the same object class to the physical object (such as mapping a virtual sofa CAD model to a physical sofa). However, we argue for the merits that physical props should be able to be repurposed for a new functionality in the virtual world. For example, a physical umbrella may be repurposed as a shovel in the virtual experience. In addition, if the mapped 3D model does not complement the theme of the virtual world, the user may be constantly reminded of their physical reality and subsequently lose focus of the alternative virtual reality, resulting in BIP.

Evidently, current systems that intend to transfer knowledge of the physical world to VR through an automated approach are still filled with challenges. Hence, current VR experiences involving interactions with objects in physical surroundings still require verbal instruction from an assisting partner [1]. This gives rise to an interesting, untapped area for exploration – to bring together the best of both worlds (assistant's physical world and player's virtual world) through a human-in-the-loop, multi-user system.

We propose ARchitect, a proof-of-concept prototype that explores the construction of a physically aware virtual experience through exploiting the asymmetric dynamic between two users, one as the "Assistant" in AR and the other as the "Player" in immersive VR. We capitalize the distinct perspectives of the shared experience in the following manner: As the Assistant walks through the environment, the mobile device scans the scene and coarse geometries such as horizontal planes are highlighted to aid the Assistant. The Assistant, who sees the physical world, then superimposes physical objects with virtual proxies in consideration of affordances [14] and theme consistency to design a virtual world. The Player may then experience the virtual world, built by the Assistant using the physical world as a template, with physical interactions and real walking. In a user study, we compared ARchitect to a baseline condition (Assistant guiding Player through verbal instructions only), demonstrating that Assistants showed decreased workload and Players experienced decreased workload, increased VR presence, and increased trust in the Assistant. We further explored the design space of ARchitect by defining design guidelines for future designers and implementing three demonstrative experiences.

In summary, our contributions are three-fold:

- A proof-of-concept prototype exploring an asymmetric dynamic as a novel approach to building a virtual experience. Our method considers physical world geometries and interaction affordances with human-in-the-loop.
- Insights from a user study exploring user dynamics with ARchitect and its effects on workload for the Assistant and workload, VR presence, and trust in the Assistant for the Player, and demonstrating its advantages compared to a baseline.
- Design guidelines for future designers of ARchitect and implementation of three example experiences to illustrate the design space of ARchitect.

RELATED WORK

Our work spans three areas of research: real walking in virtual reality, passive haptics, and asymmetric collaboration.

Real Walking in Virtual Reality

Prior work has shown that real walking in VR strengthens presence [45][39]. Due to the difficulty of accurate translation between physical and virtual space, many commercial VR experiences simply develop a physical space identical to the virtual one. For example, The VOID [3] provides a pre-configured physical room for players to walk in.

More recent literature has focused on identifying a "walkable area" so that the user may safely move around a physical environment while avoiding collisions with obstacles. RealitySkins [36] pre-scans the physical environment to generate a matching virtual floor plan with similar geometry. Hirt et al. [21] extracts the wall outlines of an environment to define a walking area. Scenograph [32] divides the large virtual scene into smaller virtual scenes to facilitate real walking in a finite tracking volume. RealityCheck [18] blends physical world objects and geometries into the VR rendering pipeline. Oasis [41] constructs a complete model of the physical environment from 3D reconstruction to procedurally generate a virtual scene with safe walkable areas. VRoamer [8] detects obstacles and extracts walkable areas on-the-fly to instantiate pre-authored virtual rooms that align with walkable areas or generate doors for undiscovered physical areas. However, prior work mainly focus on the avoidance of obstacles by restricting the user to a walkable area and potential utilization of passive haptics for interaction in the VR experience is neglected.

Passive Haptics

A major factor contributing to BIP [19] in a virtual experience is the phenomenon where a user may pass directly through a virtual object with the absence of haptic feedback. Passive haptics, a method of augmenting a high-fidelity visual virtual environment with low-fidelity physical objects, solves this pitfall [26][30][22]. Low et al. [31] adopts styrofoam walls to construct geometrically simplified scenes with AR projections. FlatWorld [35] integrates modular panels which may be rearranged between experiences to match different virtual scenes. Real Virtuality [5] conducts full-body tracking to allow interaction with physical objects registered with markers. Annexing Reality [20] opportunistically overlays virtual proxies on to physical objects based on their salient primitive shape and size. Substitutional Reality [37] pairs physical items to virtual counterparts and explores how discrepancies affect believability. iTurk [6], TurkDeck [9], and Mutual Human Interaction [7] explore the idea of orchestrating human workers to facilitate passive props. In our work, we adopt the benefit of interactions with passive haptics through human-in-the-loop pairing of physical objects to virtual proxies.

Asymmetric Collaboration

Asymmetric collaboration has been applied in VR research where an asymmetric setup, e.g. Head-Mounted Display (HMD) vs. non-HMD, is exploited to create a shared experience from two perspectives. In many cases, such collaborative efforts inherently create a power dynamic where a user in one setup predominantly receives guidance from a user in another setup. For example, Stafford et al. [42] introduces the concept of "god-like interactions" where one user in first-person AR is guided by another user with a tabletop surface which contains a virtual representation of the AR user's world. Dollhouse VR [25] presents a similar, asymmetric setup where the non-HMD user acts as a "designer" with a bird's-eye view, and the HMD user as an "occupant" of the virtual room, to collaborate on an interior designing process. Oda et al. [34] establishes an asymmetric collaboration where a remote expert manipulates virtual replicas of physical objects in a local environment to guide a local user. ShareVR [15] visualizes the virtual world to non-HMD users through floor projection and external mobile displays for collaboration on game tasks between HMD and non-HMD users. While ARchitect does not emphasize "collaboration", we borrow from the concept of asymmetric collaboration to a new domain where we utilize the relationship of asymmetric views between users for the task of translating physical world geometries and affordances.

DESIGN AND IMPLEMENTATION

Design Rationale

We designed ARchitect with three central objectives in mind. Our first design objective is to allow flexible customization of a virtual experience that reflects the physical environment. More specifically, we aim to ensure both the Player's physical safety and sense of presence in the virtual world through an understanding of interaction affordances of objects present in the physical world. To achieve this, we utilize a second participant which we name as the Assistant (human-in-the-loop). The Assistant sees the world in AR and may overlay virtual proxies on top of physical objects to construct the virtual world. The benefit of this approach is that the Assistant is capable of considering various factors when translating the physical environment to virtual space of which generative techniques do not offer. For example, the Assistant may consider affordances of physical objects to repurpose object functionality, theme consistency from a qualitative perspective, and material texture for passive haptics.

The introduction of the Assistant gives rise to our second design objective which is to ensure that the process of building the virtual experience is straightforward and requires minimal effort to achieve simple operations. Hence, while we chose not to adopt generative techniques for building our virtual environment, as largely adopted in previous literature (presented in subsection 2.1), we nonetheless implemented various automated techniques for scene understanding. These implementations do not replace the human worker but more act as an aid to eliminate the tedious mechanical work for completing certain desired operations such as aligning a virtual proxy to the same level of the physical floor. Furthermore, another design consideration is whether to frame the partnership between the Player and the Assistant to be synchronous or asynchronous. We resolved to model our system as asynchronous – the Assistant, who is the experience designer, first builds the virtual world in AR and the Player, who may be a VR layperson, then enters the experience following the Assistant's completion. The major considerations leading to this decision were that assuming a synchronous model: (1) it would be dangerous for

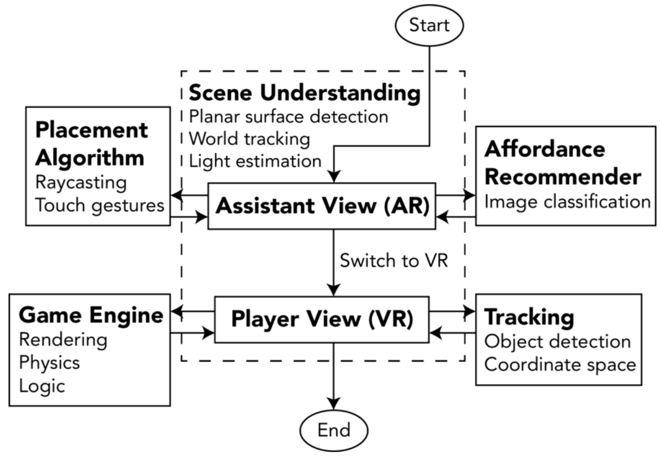


Figure 2. Overview of the ARchitect system

the Player to explore a virtual world with holes still yet to be mapped by the Assistant and (2) having new virtual objects appearing spontaneously may also lead to BIP for the Player.

Finally, our third design objective is to promote easy adoption of our system for the general public. Previous generative techniques for translating physical environments to virtual scenes are either done offline [41] or require extensive hardware support [8]. This is due to the substantial computation required for a complete pipeline of scene understanding and world generation. In our setup, we replaced the world generation component with the Assistant and managed to strip down the scene understanding component so that it may run in real-time on a modern mobile device while preserving competent accuracy for planar detection, tracking, and so on. Therefore, our design has low barriers to entry. For example, our design can accommodate a large classroom scenario where an instructor is the Assistant and students are the Players. It is less feasible for each student to have an expensive HMD. In addition, the majority of the system's scene understanding implementations were built with Unity's ARFoundation which gives cross-platform support to both ARCore for Android devices and ARKit for iOS devices.

According to our design objectives, we have defined the major components of our system as the following: Scene Understanding, the Affordance Recommender, the Placement Algorithm, Tracking, and the Game Engine (Figure 2). The following sections outline the implementation of our system, including (1) hardware and software setup, (2) definition of virtual elements, (3) scene understanding, (4) the affordance recommender and tracking, (5) AR placement and interface for the Assistant, and (6) switching from AR to VR.

Hardware and Software Setup

ARchitect was implemented using the Unity engine in C# and ran on a Google Pixel 3 XL mobile device (Qualcomm Snapdragon 845, 4GB RAM, Dual 8MP + 8MP rear camera, 3,430 mAh battery, 2960 x 1440 display). We used a custom Google Cardboard (cut-out in front for mobile device camera) to view the virtual experience.

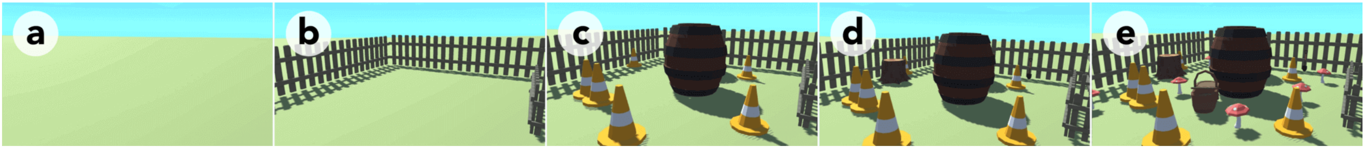


Figure 3. Components of the virtual world being added by the Assistant: (a) scene, (b) barriers, (c) obstacles, (d) interactables, (e) game objects.

Definition of Virtual Elements

We have defined the virtual world to be composed of five types of virtual elements: scene, boundary, obstacle, interactable, and game (Figure 3a-e). Scene elements are static elements (such as sky and terrain) which only contribute to the virtual experience visually. Boundary elements are virtual proxies used by the Assistant to map physical boundaries such as walls of the room. We have picked virtual objects which functionally represent boundaries (wooden fence and brick wall) for the Assistant to use. Obstacle elements are another set of virtual proxies which the Assistant may use to indicate the message of "danger ahead". Again, we picked virtual objects which functionally represent obstacles (traffic cone and wooden barrel). Interactable elements are virtual proxies that suggest possibilities of interaction. We designed two tasks in our user study to experiment with interactable elements: sitting down on a chair and grabbing an umbrella. We have selected virtual objects that afford the interactions of sitting (tree stump and rock) and grabbing (shovel and pickaxe) for the Assistant. Finally, game elements are items, randomly placed by the experimenter, to guide the Player through the experiment storyline (mushroom and mushroom crate).

Scene Understanding

Planar detection is an essential feature for providing an effortless AR experience for the Assistant. We used Unity's ARFoundation to discern distinct feature points and further detect horizontal surfaces. To anchor superimposed virtual objects to its corresponding physical space, we performed both world tracking (device position and orientation) and tracking of reference points (particular points in space). Furthermore, we added light estimation (average brightness and color temperature in physical space) and a vertical directional light source to add a layer of realism by casting shadows and maintaining a consistent color tone for the virtual proxies.

Affordance Recommender and Tracking

The affordance recommender, implemented using image classification, reduces the manual effort for the Assistant. When ARchitect detects an interactable in the physical scene (e.g. chair) in AR mode, the system automatically recommends bundles of virtual proxies with matching affordances (e.g. tree stump, rock) to the Assistant (Figure 4). These recommendations may be accepted or overruled by the Assistant. We adopted Inception V3 [44] with weights pre-trained on ImageNet [12] for image classification.

In VR mode, we also tracked the positions (bounding box) of physical interactables by performing object detection and estimated positional tracking in 3D coordinate space (6DOF) to ensure that interaction tasks are completed by the Player (e.g. must be holding a physical umbrella to hold a virtual shovel

Figure 4. The Affordance Recommender detects predefined interactable classes (e.g. "chair" class corresponds to "sitting on" interactable).

in-game). We adopted the relatively lightweight MobileNet-SDD [24] with weights pre-trained on the COCO dataset [29] for object detection. For both image classification and object detection, we used the TensorFlowSharp (C# implementation of TensorFlow) plugin for Unity (60 FPS). We performed inference once for every 15 frames to reduce processor load.

AR Placement and Interface for Assistant

We implemented a set of intuitive interactions for the Assistant to build the virtual world using an AR interface. The system first prompts the Assistant to scan the floor (Figure 5a). After briefly scanning the floor with the device, detected horizontal planes are visualized as green polygons. We use raycasting to estimate the positions of the detected horizontal planar surfaces in 3D space so that the Assistant may then superimpose virtual proxies over the physical world. To anchor a virtual proxy in the physical world, the Assistant may drag-and-drop an icon representing the virtual object onto a green polygon (Figure 5b). After the icon intersects with the horizontal plane, it is replaced with a 3D model of the virtual object which is initialized with auto-rotation to face the Assistant. Releasing the finger from the screen pins the virtual object to its last corresponding physical location. The virtual object may be further translated by long-press drag (Figure 5c), rotated by two-finger twirl (Figure 5d), and resized by two-finger pinch (Figure 5e) [2]. To remove a virtual object, the Assistant may drag the object out of the green polygon (Figure 5f).

Switching from AR to VR

After the Assistant finishes mapping the virtual world, the mobile device is switched to VR mode, encased in a Google Cardboard, and given to the Player. The screen is split into a two-screen VR display with the fisheye effect corrected with a barrel distortion shader. The Field of View (FOV) is optimized with respect to the mobile device. The AR camera is kept running to track and pin virtual proxies to physical space but the camera feed is visually occluded from the Player by applying a culling mask and replaced with scene and game

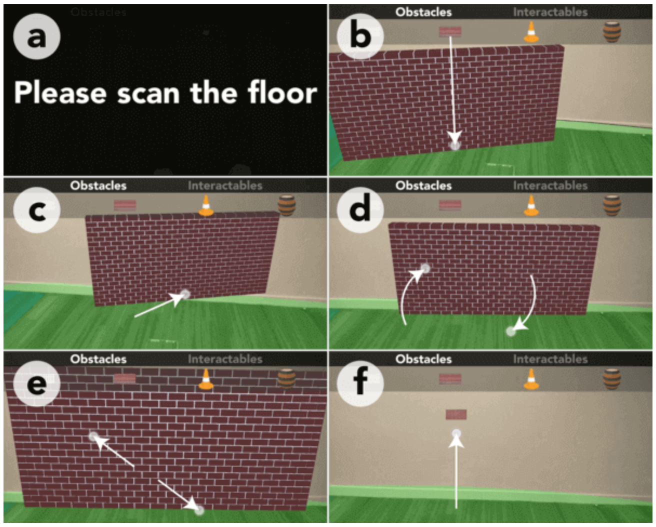


Figure 5. Operations for configuring a virtual proxy over the physical scene using ARchitect: (a) scan the floor, (b) place the virtual proxy, (c) translate the virtual proxy, (d) rotate the virtual proxy, (e) resize the virtual proxy, (f) remove the virtual proxy.

elements in addition to boundary, obstacle, and interactable elements placed by the Assistant. Therefore, the front of the Cardboard has a cut-out for the mobile device's camera.

USER STUDY

We conducted a within-subjects user study to evaluate participants' experience of presence, trust, and workload between our proposed method and a baseline. In the baseline condition, the Player was required to complete the same tasks (presented in subsection 4.4) while immersed in the virtual world, but without virtual proxies embedded through ARchitect in correspondence with physical objects. Therefore, Players saw a virtual world with scene and game objects and without barriers, obstacles, and interactables (Figure 3). Players did not see physical (real) objects. Instead, the Assistant guided the Player through vocal instructions to navigate and interact with the physical environment. This is currently a common method adopted by spectators to ensure the safety of a Player in a typical VR gameplay scenario [1]. We opted for vocal instructions as baseline because we felt it is effective for conveying the affordances of physical objects. Since our work focuses on incorporating interaction affordances, state-of-the-art physical obstacle avoidance systems [8][41] which consider all physical objects as obstacles were not adopted as baseline.

Our main research questions were: *(RQ1) How would the level of presence for the Player be affected in the virtual world constructed with ARchitect? (RQ2) How would the use of ARchitect to map physical objects to virtual proxies affect the Player's trust in the Assistant's ability to ensure safety? (RQ3) How would the levels of perceived workload for both the Assistant and the Player in completing their respective tasks change with the use of ARchitect?*

Study Design

The independent variable of the study was the System (proposed versus baseline). The participants worked in pairs to play a virtual experience (*MushroomHunt*); they were assigned to be either the Player or the Assistant. In the experimental condition, the Assistant built the VR experience using ARchitect first and then switched the platform for the Player to perform the game tasks in VR asynchronously without intervening in the gameplay. In the baseline, the Assistant guided the Player through the VR experience via speech synchronously.

The dependent variables were presence (RQ1) measured using the Slater, Usoh, and Steed's presence questionnaire (SUS) [46], trust (RQ2) measured with the Dependency component of the Trust in Close Relationships questionnaire [23], and workload (RQ3) measured by the NASA TLX questionnaire [17]; all questions were represented in a 7-point Likert scale. More specifically, we measured the presence, trust, and workload of the Player and only the workload of the Assistant after experiencing each system (ARchitect and baseline).

Procedure

We conducted the user study in a quiet laboratory furnished with interactables (chair and umbrella) and obstacles of various sizes from tables to soccer training cones. After getting participants' consent, we first collected information about individual background and measured the trust of the Player in the Assistant using the Predictability component of the Trust in Close Relationships questionnaire. This was to verify that there existed organic trust between the pairs of participants whom were recruited jointly and ensure that the later measured trust (using the Dependency component of Trust in Close Relationships questionnaire) reflected users' feeling with respect to the tested System. For clarification, we measured trust in addition to workload and presence because previous work has shown that trust is a key component for maintaining a sustainable partnership between asymmetric users [16]. We then flipped a coin to decide their respective roles randomly (one participant named the Assistant and the other became the Player). Each pair went through both the proposed method and the baseline method with the roles they were assigned to. We counterbalanced the order of the conditions; that is, half of the pairs experienced the baseline first then ARchitect, and vice versa. The room setup was rearranged between each session to lessen the possibility of spatial memory from the preceding session causing unfair advantage in the later session. Upon finishing each session (ARchitect or baseline), participants were requested to complete the aforementioned questionnaires and were asked some questions regarding their reasoning. We also performed fly-on-the-wall observations and video recordings during each session and a more in-depth post-study interview with the participants. The study lasted for approximately 30 minutes for each pair.

Participants

For this study, we recruited 18 participants (6 female, 12 male) aged from 20 to 30 (mean=22.39, SD=2.79) from our institution. All participants were recruited in pairs (i.e., 9 user pairs) through flyers and word-of-mouth with the requirement that they are friends and have adequate trust in each other under a cooperative scenario. We conducted background surveys with the participants (7-point Likert scale) at the beginning of each study. Overall, participants were moderately familiar with VR (mean=4.83, SD=1.38) and 11 participants who had previous experience with VR used it for entertainment. In addition, there was generally low interest from participants in investing in a new HMD device within the next three months (mean=1.67, SD=0.84) with cost being the primary concern (mean=5.17, SD=1.79).

Task

We implemented an experience called *MushroomHunt* for our user study. This experience consisted of the task of the Assistant guiding the Player to the combined goal of collecting six virtual mushrooms that were placed in a physical room.

Using ARchitect, the Assistant, given an AR interface on a mobile phone, was capable of seeing the physical setting of the room and had the job of overlaying virtual proxies on top of physical objects. A virtual proxy may be either a barrier, an obstacle, or an interactable, depending on the affordances of the physical object. A barrier served the purpose of confining the Player's walkable area, an obstacle signaled the Player to walk around a physical object, and an interactable conveyed the affordances of a physical object to the Player so that it may potentially be utilized to complete in-game tasks through interaction. Particularly in this experience, there were two types of virtual interactables which corresponded to physical interactable objects that afforded grasping (umbrella) and sitting on (chair). The Assistant's mission was to overlay the virtual interactables above the physical objects so that the Player may know the size, position, and orientation of the real-world objects by viewing their virtual counterparts in VR. For each physical interactable object, ARchitect provided at least two designs of virtual proxies offering the same affordances for the Assistant in AR. For example, virtual proxies for sitting interactables included a tree stump and a rock.

After the Assistant was finished with building the virtual world, the mobile device was encased into a Google Cardboard and given to the Player where ARchitect was switched to the VR mode. The Player's tasks were as follows: First, the Player needed to find a virtual shovel (an umbrella in reality), grab it with his/her hands, and use it later to dig up virtual mushrooms for points. Second, the Player was free to physically walk around. The goal was to pick three mushrooms placed randomly in the virtual scene while avoiding obstacles and boundaries (Figure 6). When the Player stuck the shovel out in close proximity to a virtual mushroom, the mushroom was collected (implemented by comparing world coordinates of Player and virtual mushroom). Third, after successfully collecting three mushrooms, the system signaled that the Player was tired and was required to sit down for a rest (Figure 7). While doing so, the Player was to physically sit on the chair which was indicated by a virtual interactable of the same affordance (sitting). Fourth, after restoring energy, the Player then proceeded to fetch the remaining three mushrooms in the scene. Fifth, once done, the Player approached a randomly placed mushroom crate (exit checkpoint) to store the collection and complete the session.

For our baseline condition, we asked the Player to perform the same tasks of grabbing the shovel, collecting three mushrooms, sitting down to rest, fetching the remaining three mushrooms, and storing the collection in the mushroom crate. However, the virtual proxies that represented physical boundaries, obstacles, and interactables were unavailable and the Assistant had to verbally instruct the Player to avoid bumping into boundaries and obstacles and describe how to approach interactables.

Figure 6. The virtual view of the Player in baseline (a) and in ARchitect (b) while bending down to collect virtual mushrooms (c).

Figure 7. The virtual view of the Player in baseline (a) and in ARchitect (b) while sitting down to take a rest (c).

Overall, we designed the task to encourage the Assistant to consider the affordances of physical objects, weaving interaction with passive haptics naturally into a narrative storyline.

EXPERIMENTAL RESULT AND ANALYSIS

Quantitative Results

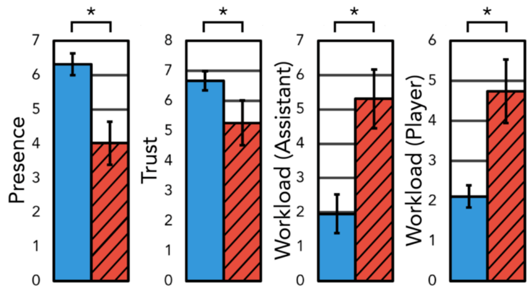
Scores for presence, trust, workload (for Assistant), and workload (for Player) were analyzed through a one-way repeated-measures ANOVA. Figure 8 shows an overview for comparison between ARchitect and the baseline. We further examined the quantitative results with informal interviews conducted after each experience (ARchitect or baseline) which focused on asking participants their rationale behind answering the questionnaires.

Presence

Players reported a significantly higher VR presence (SUS) using ARchitect (mean=6.31, SD=0.32) compared to the baseline (mean=4.02, SD=0.63) ($F(1,8)$=265.10, p<0.001, η^2=0.86). All Players stated that they felt a *"higher sense of presence"* in the virtual environment with ARchitect compared to the baseline. Six Players indicated that they enjoyed the natural interactions with passive haptics from physical objects and expressed that the haptic feedback added more realism to the virtual world. In addition, Players were not distracted by the Assistant (baseline involved verbal instructions from the Assistant which led to BIP). This also led to other interesting findings. For example, Players expressed that they were more confident in their actions despite being completely immersed in the virtual world without immediate supervision from the Assistant. Furthermore, Players stated that they treated all virtual obstacles equally, whether or not they corresponded to actual physical obstacles, and attempted to avoid them in their navigation.

Trust

Players reported a significantly higher trust in the assistant (Dependency component of Trust in Close Relationships) using ARchitect (mean=6.67, SD=0.32) compared to the baseline (mean=5.27, SD=0.75) ($F(1,8)$=60.83, p<0.001, η^2=0.63). Players generally trusted that the placement of virtual proxies



	Mean	SD	Mean	SD	Mean	SD	Mean	SD
ARchitect	6.31	0.32	6.67	0.32	1.96	0.56	2.11	0.28
Baseline	4.02	0.63	5.27	0.75	5.31	0.86	4.74	0.80

* p<0.001, error bars are standard errors

Figure 8. Means and standard deviations of presence, trust, workload (Assistant), and workload (Player) scores comparing ARchitect and the baseline. Error bars are standard errors.

by the Assistant was *"accurate"*. They stated that compared to verbal instructions, visual indications seemed much more trustworthy, especially when involving interactions (grasping, sitting down on) and in obstacle avoidance. Interestingly, Assistants also experienced higher trust in themselves in giving out correct instructions for the partner Player. Assistants stated that with the baseline, they lacked the knowledge on how the virtual scene appeared and were hence uncertain about how to give certain instructions to the Player. On the other hand, Assistants were able to visualize what the Player would see in the virtual world with ARchitect's AR implementation to build the virtual world *"in the shoes"* of the Player.

Workload

Assistants reported a significantly lower workload (NASA TLX) using ARchitect (mean=1.96, SD=0.56) compared to the baseline (mean=5.31, SD=0.86) ($F(1,8)=66.77$, $p<0.001$, $\eta^2=0.86$). The difference averaged from a generally lower workload of mental, physical and temporal demands when using ARchitect. Assistants found scene understanding aids to be *"helpful"* towards reducing manual effort, the interface and interaction controls to be *"easy and intuitive"*, and overall, the entire pipeline of the system to be *"straightforward"*. Compared to having to engage in lengthy and often abstruse descriptions of how to interact with a physical object, Assistants also enjoyed the capability of easily mapping physical objects with virtual proxies offering matching affordances.

Furthermore, Players also reported a significantly lower workload using Architect (mean=2.11, SD=0.28) (averaging mental, physical, temporal) compared to the baseline (mean=4.74, SD=0.80) ($F(1,8)=169.80$, $p<0.001$, $\eta^2=0.85$). Players stated that they enjoyed the capability of making decisions independently and performed navigations and interactions more efficiently and with greater confidence.

Qualitative Feedback

At the end of each user study, we asked participants four open-ended questions, summarized as the following:

"Overall, did you prefer the baseline or our proposed system? Please elaborate." Seven out of nine Assistants and nine out of nine Players preferred our system. Assistants stated that verbal instructions were *"more inaccurate"* in position, *"more involved"* and more *"dependent on the Assistant's ability to articulate verbally"*. They also expressed that they *"had much greater confidence that the Player is safe"* as the Player could see the virtual proxies to determine the size, orientation, and scale of their corresponding physical objects. All Assistants also found the process of building the world to be fun and engaging (*"building a world like Minecraft"*). In addition, one Assistant pointed out that for the baseline, the same physical obstacle needed to be described every time the Player passed it but for ARchitect, the physical obstacle needed only to be anchored once in the virtual world. For the Players, they stated that they felt *"more in control"*, *"independent"*, and that their *"life is no longer in the hands of the Assistant"*. One Player enjoyed a game experience without *"constant interruptions"* by the Assistant warning about obstacles nearby. A Player who regularly plays VR games stated that he *"would want this system"* in his current VR gear so that he *"wouldn't bump into furniture in the house"* and an Assistant who is a secondary school teacher stated that he would like to *"introduce this to his students"* because it was *"really intuitive to use"* and *"engaging"*. Nonetheless, one Assistant preferred the baseline stating that *"speaking was more personal"* and *"greater for bonding"*. Another Assistant suggested a mix of the two by adding live voice communication in ARchitect.

"What did you like or found easy about our system?" All Assistants found the AR UI *"easy to use"* and interaction controls to be *"natural"* (drag and drop and resize, rotate, and translate finger gestures). They also found the assistive scene understanding implementation to be *"very helpful"* (planar detection, edge detection, recommendation with classification, *"shadows were realistic"*). Moreover, Assistants also generally felt that the models for virtual proxies were *"well chosen"* and had *"good representation"*. For the Players, they felt that the storyline was *"fun"* and *"engaging"* and that the virtual environment was *"vivid"* and *"realistic"*. Many Players were particularly excited when talking about the involvement of passive haptics (*"When I sat on the virtual tree stump, I thought I would fall right through but there was actually something physically there"*, *"I loved how I could physically scoop up mushrooms with my shovel"*)

"What did you not like, found difficult or frustrating about our system?" Three Players stated that they *"felt dizzy"* after the VR experience although the dizziness did not affect task performances. Two Assistants mentioned that it was sometimes frustrating to *"perfectly align virtual proxies over physical objects"*, especially in individual cases where many virtual proxies overlap. One Player stated that while the virtual tree stump was *"still believable"* to a degree that it could be sat on, it nonetheless had a *"different feel (material texture) than expected"*.

"How would you improve the system?" Two Players requested adding full body tracking to make the experience *"even more realistic and immersive"*. Some Players said that the virtual proxies for obstacles should be *"even scarier"* (e.g. fire, spikes) and also wanted the addition of animation and sound to virtual objects. Two Assistants expressed that they would like the capability to do *"more incremental adjustments"* in translation, rotation, and resizing and also proposed smart "snapping" capabilities for object alignment.

Discussion

One clear benefit of ARchitect compared to the baseline is its better capability of providing navigational information. In the baseline condition, Assistants generally provided instructions incrementally (e.g. "turn left by 45 degrees" and "walk two steps backward and sit down"). This phenomenon may be analogical to "fog view" where only a small finite distance in a radius around the observer, in this case the Player, is "visible". For example, since the understanding of instructions, such as the length of a step, may differ between the two parties, it would be difficult for the Assistant to provide instructions for long-term actions such as "walk 12 steps forward, turn around, and sit down" accurately. This resulted in the Player being unaware and concerned about the safety of surroundings that were not in close proximity and hence walking extremely slowly and cautiously. The uncertainty of the Player may be further aggravated by the fact that quantitative measures, such as exact walking distance or exact degree of turning, were unavailable to the Assistant, making extrapolated estimations highly prone to errors. This resulted in five out of the nine Players accidentally bumping into obstacles or missing interactables by small offsets in the baseline condition. Conversely, after the Assistant had mapped physical objects to virtual proxies with correct positioning, scaling, and orientation with ARchitect, the Player could easily plan a safe route to walk and also knew how to interact with interactable objects. With ARchitect, Players could independently discern the entire physical surroundings (using virtual counterparts), effectively eliminating the "fog view" phenomenon.

Being able to visualize the physical environment through virtual proxies also significantly increased the enjoyment and confidence of Players in task completion. Although Players were allowed to ask the Assistant questions in the baseline condition, Players walked across the map at a much faster pace, completed tasks quicker with higher degrees of accuracy, and seemed less worried about safety and more engaged in playing through the narrative storyline. The total time spent by Assistants and Players on completing their tasks asynchronously using ARchitect (mean=308.44 seconds, SD=35.10) was generally shorter and more consistent than the total time spent on the baseline condition (mean=365.89 seconds, SD=64.22) ($F(1,8)=10.08$, $p<0.05$, $\eta^2=0.26$). Players were also much less dependent on following the exact instructions given by the Assistants and explored the virtual world more thoroughly.

Players were also generally more positive and less cautious in interacting with passive haptics based on qualitative observations during the studies. In the baseline condition, since interactables were invisible to the Player, the Player could only infer the interactables through descriptions from the Assistant. This resulted in Players often extending one hand cautiously towards the location of the interactable, then performing a swaying motion until the hand collided with the interactable, and finally feeling around the interactable to get a sense of its shape before performing the interaction of grasping or sitting. On the other hand, with ARchitect, Players skipped actions for "testing the waters", performing interactions that corresponded to the affordances of the virtual proxies directly. While there may still be some discrepancies in haptic information (such as in shape, weight, material) between a physical object and a virtual proxy, Players were generally not confused with the mismatch based on post-study interview feedback. For example, many participants stated that they believed they were holding a physical shovel throughout the experience, unaware that it was actually an umbrella. This phenomenon may be supported by prior work done by Warren et al. on visual-proprioceptive interaction which suggests that as long as discrepancies between visual and haptic information are not too large, visual information dominates and the discrepancies do not adversely affect the user's virtual reality experience [47]. Hence, Players seemed to really enjoy the incorporation of passive haptics in their experiences.

DESIGN GUIDELINES AND EXAMPLE EXPERIENCES

Based on the feedback from our user study, we offered several guidelines for future designers of ARchitect and implemented three example experiences for demonstration.

Design Guidelines

Use passive haptics. Since Players generally responded with positive excitement to the implementation of passive haptics in *MushroomHunt*, we encourage future designers to make use of physical props as passive haptics in the virtual world. Some of these props may be interacted in a one-off approach (e.g. sitting on chair) while others may be extended throughout the experience (e.g. carrying shovel to collect mushrooms).

Build imaginary worlds. As long as the affordances of virtual proxies match those of physical objects, there is no limitation on how the virtual world may be perceived visually. Designers are encouraged to repurpose common objects in the physical scene with otherworldly designs that transport the Player to a different space or time.

Map virtual proxies with deliberation. Ensuring that suitable virtual proxies are mapped with appropriate size, orientation, and scale is important for creating a safe and believable virtual experience. Findings from Simeone et al. [37] and Kwon et al [27] also advise Assistants to minimize mismatches in manipulatable parts of interactables (e.g. handles), select proxies in consideration of how shape and texture may affect the Player's expectation of haptics, and take note that reality-based virtual environments have stricter mapping requirements.

Create fake barriers and obstacles. An interesting area for exploration is to exploit the asymmetric relationship between Assistant and Player to create indistinguishable "fake barriers and obstacles" that do not correspond to physical objects as means to manipulate space in the virtual world. From our user study, Players still tended to avoid these objects during their navigation. Nonetheless, Assistants are advised not to create "fake interactables" as the illusion is lost once Player try to interact with them.

Utilize modularity for theme consistency. Virtual experiences designed with ARchitect generally consist of five components:

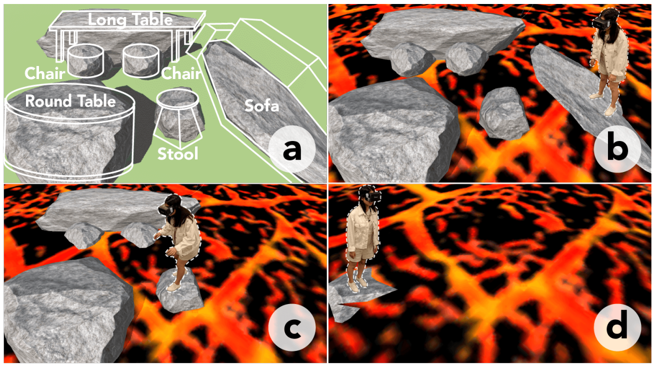


Figure 9. (a) In *LavaEscape*, the Assistant first maps physical furniture to rocks of various shapes and sizes (and heights). The Player steps onto the lowest piece of furniture (b) and continues moving onto higher furniture as the lava plane rises (c) until reaching the higher ground (d).

scene, boundary, obstacle, interactable, and game objects. The modularity of the experience means that each of the components may be designed in groups with consideration of theme consistency. For example, in a car simulation virtual experience, there may be three different types of virtual proxies for the obstacle component: traffic cones, boulders from a landslide, and deer. All of these choices reflect obstacles that may be logically seen on the road.

Design engaging storylines. With the introduction of "interactables" in ARchitect, we encourage future designers to design engaging narratives where Players may perform different interactions to complete various tasks within a storyline, such as the ones designed in *MushroomHunt*. The narratives may also be non-linear to promote replayable experiences.

Example Experiences

We further implemented example experiences to illustrate applications of the design guidelines and demonstrate ARchitect's ability of supporting VR experience design from three directions: (1) repurpose design, (2) level design, and (3) personalization through modular design.

LavaEscape (Repurpose Design)

In *LavaEscape*, we explore the benefit of ARchitect for in situ repurposing of everyday objects into game elements. *LavaEscape* is inspired by the game: The floor is lava [13], in which players pretend that the ground is made of lava and climb onto furniture to move around without coming in contact with the floor. In *LavaEscape*, the Assistant first maps interactables in the room that afford to be stood on or climbed on to virtual rocks (Figure 9a). After the Assistant is finished with the mapping, the Player begins the game from the ground. A virtual lava plane then emerges from ground level and subsequently rises incrementally with time. Thus, the Player has to climb across interactables with increasing heights in order to stay above the level of the lava plane (Figure 9b-d).

LavaEscape was designed to investigate the possibilities of repurposing physical objects in the room into passive haptics, without the need for specialized props to fit a storyline. The experience also examines the Assistant's ability to select appropriate physical objects for mapping (e.g. stable objects to be stood on) as well as the ability to accurately fine-tune the

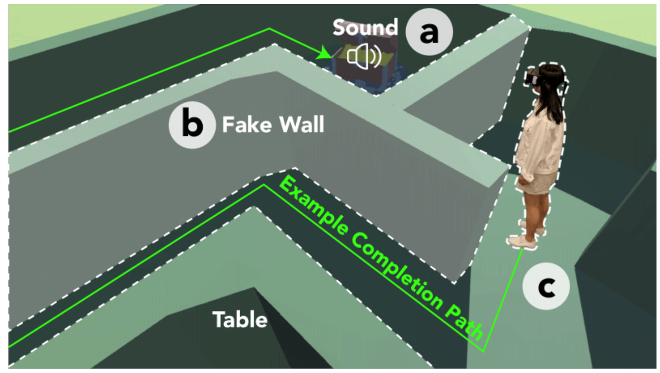


Figure 10. A Player attempting to locate the treasure chest (3D audio source) (a) while playing *Maze*. The virtual maze corresponds to the physical room configuration (walls and table) with the addition of a fake virtual wall (b). (c) shows an example completion path that may be taken by the Player.

size, orientation, and scale of virtual proxies for a reality-based virtual environment with strict mapping requirements.

Maze (Level Design)

In *Maze*, we explore the use of ARchitect for level design. The Assistant first considers the physical room configuration to create a virtual maze by mapping the boundaries of the room with virtual walls and using virtual walls to enclose obstacles (and interactables). The Assistant may also place indistinguishable "fake" virtual walls (e.g. virtual walls that do not correspond to any physical object or barrier) to increase the complexity of the maze (Figure 10b). Hence, the same physical environment may be used for the design of multiple "maze levels" through different configurations of fake virtual walls. Finally, the Assistant is to place a treasure chest in the maze to indicate the final checkpoint. The treasure chest is a 3D spatial audio source implemented with Google Resonance Audio (Figure 10a). The Player's task is then to find a route in the virtual maze (possibility planted with fake walls) to reach the location of the treasure chest (Figure 10c).

Maze was designed to showcase the use of ARchitect for level design while reusing the physical environment. We explore the usage of fake barriers and obstacles to manipulate the concept of space in the virtual world under an asymmetric setting. Hence, with ARchitect, the virtual world may be designed to have greater variety and be far more complex. An interesting area for further exploration may be its usage for designing redirected walking [43] experiences where Players explore a virtual world that is deceivingly larger than the physical configuration offered in reality.

Sandbox (Personalization through Modular Design)

In *Sandbox*, we illustrate the ability to create personalized experiences by exploring the modularity of ARchitect. Rather than being a game with pre-configured virtual proxies, *Sandbox* is a testbed for Assistants to establish their own game theme and configure their own set of virtual proxies for different components of the virtual world. Thus, Assistants define their own scene (sky and floor) as well as sets of virtual proxies to represent boundaries, obstacles, interactables, or game objects, selected from Google Poly, a library of 3D models. In an example, we established the theme to be a futuristic skateboard

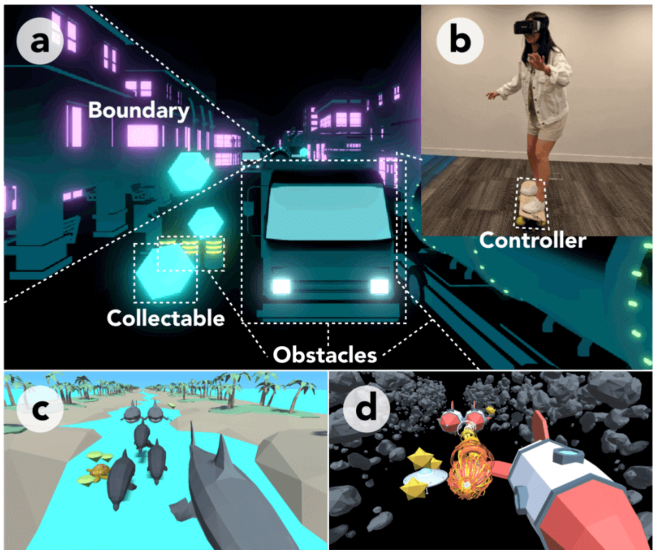


Figure 11. A Player playing with a skateboard simulator configured with *Sandbox* (a) using a physical skateboard controller (b). The experience may be reconfigured to a different theme (e.g. aquatic (c) or space (d)).

simulator with a corresponding scene and defined buildings as boundary elements, vehicles and rising bollards as obstacles, and crystals as collectable game objects (Figure 11a). We also implemented an electric skateboard controller to be the interactable for in-game controls (Figure 11b).

The electric skateboard controller is a skateboard attached with an Arduino microcontroller with a gyroscope on the lower back, a pressure sensor on the upper front, and Turnigy Aerodrive SK3 brushless motors on the wheels for actuation. The Arduino reads data from the gyroscope and pressure sensor to detect the Player's actions, which is classified into three categories: turning left by tilting to the left of the skateboard, turning right by tilting to the right of the skateboard, and going forward by stepping on the pressure sensor.

Sandbox was designed to highlight the benefits of modularity for the personalization of game design using ARchitect. As seen from the given example, it is relatively straightforward for the Assistant to decide a central game theme and then to define virtual proxies in component groups. For example, the Assistant may also easily change the theme to be aquatic or space and define the virtual proxies accordingly (Figure 11c-d). In *Sandbox*, we also explore the possibility of incorporating a physical controller into the virtual experience, perhaps even extending to haptic actuators to blend active haptics [28][10].

LIMITATIONS AND FURTHER WORK

While ARchitect was positively received in our user study, there are still several challenges that we plan to address for future work.

First, even with scene understanding as an aid, overlaying virtual proxies on top of physical objects still required the Assistant to perform careful fine-tuning for alignment. A possible improvement may be reproducing the implementation of Nuernberger et al. [33] by extracting 3D edge and planar constraints for "snapping". Second, we implemented the Affordance Recommender in ARchitect through pre-defined, direct mapping of object classes to affordances, which is fairly limited. For example, we manually mapped a detection of the "chair" object class to detecting a "sitting on" interactable. However, a chair may also afford interactions other than sitting. We hope to further explore the affordance taxonomy to improve the flexibility of the Affordance Recommender. Third, we currently adopt a static visualization of interactables in VR. For example, a static virtual shovel appears in the corner of the Player's view when ARchitect detects a physical umbrella in the camera feed. We expect to extend this into a dynamic visualization by automatically adjusting the position and size of the interactable visualization according to the bounding box coordinates retrieved from object detection. Fourth, Players have suggested areas for improving the general virtual experience. These include adding full-body tracking, animation, and sound to increase realism and improving the frame rate to reduce dizziness in VR. Fifth, we anticipate a more rigorous technical evaluation on the Scene Understanding and Affordance Recommender components to verify a precise and generalizable system. For Scene Understanding, we plan to extensively evaluate tracking accuracy in various scenes and lighting conditions and assess drifting error. For Affordance Recommender, we intend to evaluate top-5 classification accuracy for various indoor objects (e.g., furniture). Finally, ARchitect currently facilitates a dynamic that encourages the Assistant to lead the Player through the asymmetry of AR and VR and in an asynchronous manner. However, it would be interesting to extend ARchitect to applications where the roles may be balanced or even reversed (Player leading Assistant). Moreover, we also plan to improve ARchitect so that partnership may be done in a synchronous manner while preserving safety and presence for the Player. We anticipate an approach by learning the saliency of the Player based on work from Sitzmann et al. [38].

CONCLUSION

In this work, we presented ARchitect, a proof-of-concept prototype which facilitates flexible customization of a virtual experience based on the physical environment. By bringing human-in-the-loop, ARchitect provides the Assistant an AR interface with scene understanding aid to translate affordances of physical objects to interactions with virtual proxies. We conducted a within-subjects study (9 user pairs) comparing ARchitect to a baseline condition (Assistant guiding Player through verbal instruction), showing a decreased perceived workload for Assistants and Players and increased VR presence and trust in the Assistant for Players. We further provided guidelines for future designers of ARchitect and implemented three example experiences for demonstration.

ACKNOWLEDGMENTS

We would like to thank all the participants who volunteered for the user study, mentors and peers in our research group for providing valuable feedback on iterations of our system and paper write-up, and the anonymous reviewers for their comments. This work is partially supported by the Research Grants Council of the Hong Kong Special Administrative Region, China under Grant No. T44-707/16-N.

REFERENCES

[1] 2019. Keep Talking and Nobody Explodes. (2019). Retrieved June 15, 2019 from http://www.keeptalkinggame.com/presskit/sheet.php?p=keep_talking_and_nobody_explodes

[2] 2019. Touch Gesture Reference Guide. (2019). Retrieved July 6, 2019 from https://www.lukew.com/ff/entry.asp?1071

[3] 2019. The VOID | A Virtual Reality Experience. (2019). Retrieved June 15, 2019 from https://www.thevoid.com/

[4] Armen Avetisyan, Manuel Dahnert, Angela Dai, Manolis Savva, Angel X. Chang, and Matthias Niessner. 2019. Scan2CAD: Learning CAD Model Alignment in RGB-D Scans. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[5] Sylvain Chagué and Caecilia Charbonnier. 2016. Real Virtuality: A Multi-user Immersive Platform Connecting Real and Virtual Worlds. In *Proceedings of the 2016 Virtual Reality International Conference (VRIC '16)*. ACM, New York, NY, USA, Article 4, 3 pages. DOI: http://dx.doi.org/10.1145/2927929.2927945

[6] Lung-Pan Cheng, Li Chang, Sebastian Marwecki, and Patrick Baudisch. 2018. iTurk: Turning Passive Haptics into Active Haptics by Making Users Reconfigure Props in Virtual Reality. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. ACM, New York, NY, USA, Article 89, 10 pages. DOI:http://dx.doi.org/10.1145/3173574.3173663

[7] Lung-Pan Cheng, Sebastian Marwecki, and Patrick Baudisch. 2017. Mutual Human Actuation. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology (UIST '17)*. ACM, New York, NY, USA, 797–805. DOI: http://dx.doi.org/10.1145/3126594.3126667

[8] Lung-Pan Cheng, E. Ofek, C. Holz, and A. 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)*. 359–366. DOI: http://dx.doi.org/10.1109/VR.2019.8798074

[9] Lung-Pan Cheng, Thijs Roumen, Hannes Rantzsch, Sven Köhler, Patrick Schmidt, Robert Kovacs, Johannes Jasper, Jonas Kemper, and Patrick Baudisch. 2015. TurkDeck: Physical Virtual Reality Based on People. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology (UIST '15)*. ACM, New York, NY, USA, 417–426. DOI: http://dx.doi.org/10.1145/2807442.2807463

[10] Inrak Choi, Eyal Ofek, Hrvoje Benko, Mike Sinclair, and Christian Holz. 2018. CLAW: A Multifunctional Handheld Haptic Controller for Grasping, Touching, and Triggering in Virtual Reality. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. ACM, New York, NY, USA, Article 654, 13 pages. DOI: http://dx.doi.org/10.1145/3173574.3174228

[11] Angela Dai and Matthias Nießner. 2018. 3DMV: Joint 3D-Multi-View Prediction for 3D Semantic Scene Segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)*.

[12] J. Deng, W. Dong, R. Socher, L. Li, Kai Li, and Li Fei-Fei. 2009. ImageNet: A large-scale hierarchical image database. (June 2009). DOI: http://dx.doi.org/10.1109/CVPR.2009.5206848

[13] Deven Fosberry and Pratik Lakhotia. 2014. Playing 'The Floor is Lava' in Real Life. *Journal of Interdisciplinary Science* 3 (03 2014).

[14] James Jerry Gibson. 2013. James J. Gibson From: the Ecological Approach to Visual Perception Chapter 8 the Theory of Affordances.

[15] Jan Gugenheimer, Evgeny Stemasov, Julian Frommel, and Enrico Rukzio. 2017. ShareVR: Enabling Co-Located Experiences for Virtual Reality Between HMD and Non-HMD Users. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. ACM, New York, NY, USA, 4021–4033. DOI: http://dx.doi.org/10.1145/3025453.3025683

[16] Jan Gugenheimer, Evgeny Stemasov, Harpreet Sareen, and Enrico Rukzio. 2018. FaceDisplay: Towards Asymmetric Multi-User Interaction for Nomadic Virtual Reality. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. ACM, New York, NY, USA, Article 54, 13 pages. DOI: http://dx.doi.org/10.1145/3173574.3173628

[17] Sandra G. Hart and Lowell E. Staveland. 1988. Development of NASA-TLX (Task Load Index): Results of Empirical and Theoretical Research. In *Human Mental Workload*, Peter A. Hancock and Najmedin Meshkati (Eds.). Advances in Psychology, Vol. 52. North-Holland, 139 – 183. DOI: http://dx.doi.org/https://doi.org/10.1016/S0166-4115(08)62386-9

[18] 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 (CHI '19)*. ACM, New York, NY, USA, Article 347, 12 pages. DOI: http://dx.doi.org/10.1145/3290605.3300577

[19] Zhenyi He, Fengyuan Zhu, Ken Perlin, and Xiaojuan Ma. 2018. Manifest the Invisible: Design for Situational Awareness of Physical Environments in Virtual Reality. *arXiv e-prints* (Sep 2018), arXiv:1809.05837.

[20] Anuruddha Hettiarachchi and Daniel Wigdor. 2016. Annexing Reality: Enabling Opportunistic Use of Everyday Objects As Tangible Proxies in Augmented Reality. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. ACM, New York, NY, USA, 1957–1967. DOI: http://dx.doi.org/10.1145/2858036.2858134

[21] Christian Hirt, Markus Zank, and Andreas Kunz. 2017. Real-time Wall Outline Extraction for Redirected Walking. In *Proceedings of the 23rd ACM Symposium on Virtual Reality Software and Technology (VRST '17)*. ACM, New York, NY, USA, Article 72, 2 pages. DOI: http://dx.doi.org/10.1145/3139131.3143416

[22] H. G. Hoffman. 1998. Physically touching virtual objects using tactile augmentation enhances the realism of virtual environments. In *Proceedings. IEEE 1998 Virtual Reality Annual International Symposium (Cat. No.98CB36180)*. 59–63. DOI: http://dx.doi.org/10.1109/VRAIS.1998.658423

[23] John Holmes and John Rempel. 1985. Trust in Close Relationships. *Journal of Personality and Social Psychology* 49 (07 1985). DOI: http://dx.doi.org/10.1037//0022-3514.49.1.95

[24] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. 2017. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. *arXiv e-prints* (Apr 2017), arXiv:1704.04861.

[25] Hikaru Ibayashi, Yuta Sugiura, Daisuke Sakamoto, Natsuki Miyata, Mitsunori Tada, Takashi Okuma, Takeshi Kurata, Masaaki Mochimaru, and Takeo Igarashi. 2015. Dollhouse VR: A Multi-view, Multi-user Collaborative Design Workspace with VR Technology. In *SIGGRAPH Asia 2015 Emerging Technologies (SA '15)*. ACM, New York, NY, USA, Article 8, 2 pages. DOI:http://dx.doi.org/10.1145/2818466.2818480

[26] Brent Edward Insko. 2001. *Passive Haptics Significantly Enhances Virtual Environments*. Ph.D. Dissertation. Advisor(s) Brooks,Jr., Frederick P. AAI3007820.

[27] Eun Kwon, Gerard J. Kim, and Sangyoon Lee. 2009. Effects of Sizes and Shapes of Props in Tangible Augmented Reality. In *Proceedings of the 2009 8th IEEE International Symposium on Mixed and Augmented Reality (ISMAR '09)*. IEEE Computer Society, Washington, DC, USA, 201–202. DOI: http://dx.doi.org/10.1109/ISMAR.2009.5336463

[28] Jaeyeon Lee, Mike Sinclair, Mar Gonzalez-Franco, Eyal Ofek, and Christian Holz. 2019. TORC: A Virtual Reality Controller for In-Hand High-Dexterity Finger Interaction. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI '19)*. ACM, New York, NY, USA, Article 71, 13 pages. DOI: http://dx.doi.org/10.1145/3290605.3300301

[29] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. 2014. Microsoft COCO: Common Objects in Context. *arXiv e-prints* (May 2014), arXiv:1405.0312.

[30] R. W. Lindeman, J. L. Sibert, and J. K. Hahn. 1999. Hand-held windows: towards effective 2D interaction in immersive virtual environments. In *Proceedings IEEE Virtual Reality (Cat. No. 99CB36316)*. 205–212. DOI: http://dx.doi.org/10.1109/VR.1999.756952

[31] Kok-Lim Low, Greg Welch, Anselmo Lastra, and Henry Fuchs. 2001. Life-sized Projector-based Dioramas. In *Proceedings of the ACM Symposium on Virtual Reality Software and Technology (VRST '01)*. ACM, New York, NY, USA, 93–101. DOI: http://dx.doi.org/10.1145/505008.505026

[32] 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 (UIST '18)*. ACM, New York, NY, USA, 511–520. DOI: http://dx.doi.org/10.1145/3242587.3242648

[33] 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 (CHI '16)*. ACM, New York, NY, USA, 1233–1244. DOI: http://dx.doi.org/10.1145/2858036.2858250

[34] Ohan Oda, Carmine Elvezio, Mengu Sukan, Steven Feiner, and Barbara Tversky. 2015. Virtual Replicas for Remote Assistance in Virtual and Augmented Reality. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology (UIST '15)*. ACM, New York, NY, USA, 405–415. DOI: http://dx.doi.org/10.1145/2807442.2807497

[35] J. Pair, U. Neumann, D. Piepol, and B. Swartout. 2003. FlatWorld: combining Hollywood set-design techniques with VR. *IEEE Computer Graphics and Applications* 23, 1 (Jan 2003), 12–15. DOI: http://dx.doi.org/10.1109/MCG.2003.1159607

[36] L. Shapira and D. Freedman. 2016. Reality Skins: Creating Immersive and Tactile Virtual Environments. In *2016 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. 115–124. DOI: http://dx.doi.org/10.1109/ISMAR.2016.23

[37] Adalberto L. Simeone, Eduardo Velloso, and Hans Gellersen. 2015. Substitutional Reality: Using the Physical Environment to Design Virtual Reality Experiences. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI '15)*. ACM, New York, NY, USA, 3307–3316. DOI:http://dx.doi.org/10.1145/2702123.2702389

[38] Vincent Sitzmann, Ana Serrano, Amy Pavel, Maneesh Agrawala, Diego Gutierrez, Belen Masia, and Gordon Wetzstein. 2018. Saliency in VR: How Do People Explore Virtual Environments? *IEEE Transactions on Visualization and Computer Graphics* 24, 4 (April 2018), 1633–1642. DOI: http://dx.doi.org/10.1109/TVCG.2018.2793599

[39] Mel Slater, Martin Usoh, and Anthony Steed. 1995. Taking Steps: The Influence of a Walking Technique on Presence in Virtual Reality. *ACM Trans. Comput.-Hum. Interact.* 2, 3 (Sept. 1995), 201–219. DOI: http://dx.doi.org/10.1145/210079.210084

[40] Shuran Song and Jianxiong Xiao. 2014. Sliding Shapes for 3D Object Detection in Depth Images. In *Proceedings of the European Conference on Computer Vision (ECCV)*.

[41] 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* PP (10 2017), 1–1. DOI: http://dx.doi.org/10.1109/TVCG.2017.2762691

[42] Aaron Stafford, Wayne Piekarski, and Bruce Thomas. 2006. Implementation of God-like Interaction Techniques for Supporting Collaboration Between Outdoor AR and Indoor Tabletop Users. In *Proceedings of the 5th IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR '06)*. IEEE Computer Society, Washington, DC, USA, 165–172. DOI:http://dx.doi.org/10.1109/ISMAR.2006.297809

[43] Qi Sun, Anjul Patney, Li-Yi Wei, Omer Shapira, Jingwan Lu, Paul Asente, Suwen Zhu, Morgan Mcguire, David Luebke, and Arie Kaufman. 2018. Towards Virtual Reality Infinite Walking: Dynamic Saccadic Redirection. *ACM Trans. Graph.* 37, 4, Article 67 (July 2018), 13 pages. DOI: http://dx.doi.org/10.1145/3197517.3201294

[44] C. Szegedy, Wei Liu, Yangqing Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. 2015. Going deeper with convolutions. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 1–9. DOI: http://dx.doi.org/10.1109/CVPR.2015.7298594

[45] Martin Usoh, Kevin Arthur, Mary C. Whitton, Rui Bastos, Anthony Steed, Mel Slater, and Frederick P. Brooks, Jr. 1999. Walking –> Walking-in-place –> Flying, in Virtual Environments. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '99)*. ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 359–364. DOI: http://dx.doi.org/10.1145/311535.311589

[46] Martin Usoh, Ernest Catena, Sima Arman, and Mel Slater. 2000. Using Presence Questionnaires in Reality. *Presence: Teleoper. Virtual Environ.* 9, 5 (Oct. 2000), 497–503. DOI: http://dx.doi.org/10.1162/105474600566989

[47] David H. D. Warren and Wallace T. Cleaves. 1971. Visual-proprioceptive interaction under large amounts of conflict. *Journal of experimental psychology* 90 2 (1971), 206–14.