PointShopAR: Supporting Environmental Design Prototyping Using Point Cloud in Augmented Reality

Zeyu Wang
HKUST(GZ) & HKUST
zeyuwang@ust.hk

Cuong Nguyen
Adobe Research
cunguyen@adobe.com

Paul Asente
Adobe Research
research@asente.com

Julie Dorsey
Yale University
julie.dorsey@yale.edu

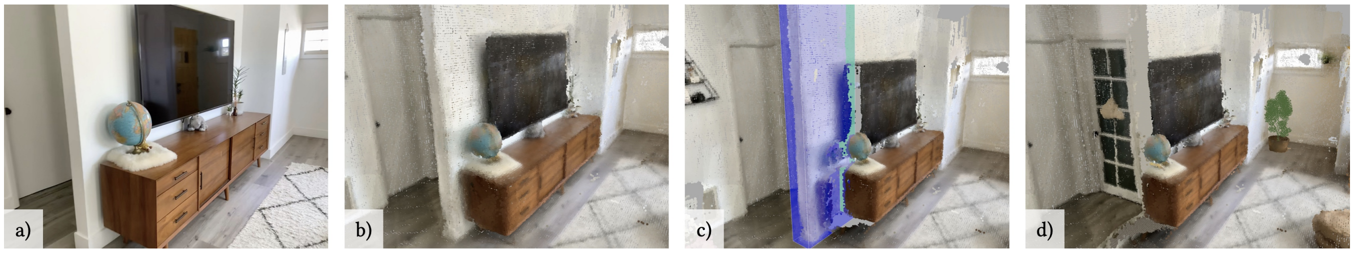

Figure 1: The PointShopAR workflow. a) Designers can quickly scan a physical environment like this living room on a consumer tablet. b) Our system constructs a localized point cloud with color in a few seconds. c) Designers can edit the scan on the same tablet, using selection, transformation, hole filling, and animation. d) The result of the designer removing part of the wall, making it easier for residents to walk from the bedroom to the living room.

ABSTRACT

We present PointShopAR, a novel tablet-based system for AR environmental design using point clouds as the underlying representation. It integrates point cloud capture and editing in a single AR workflow to help users quickly prototype design ideas in their spatial context. We hypothesize that point clouds are well suited for prototyping, as they can be captured more rapidly than textured meshes and then edited immediately in situ on the capturing device. We based the design of PointShopAR on the practical needs of six architects in a formative study. Our system supports a variety of point cloud editing operations in AR, including selection, transformation, hole filling, drawing, morphing, and animation. We evaluate PointShopAR through a remote study on usability and an in-person study on environmental design support. Participants were able to iterate design rapidly, showing the merits of an integrated capture and editing workflow with point clouds in AR environmental design.

CCS CONCEPTS

• **Human-centered computing → Mixed / augmented reality**.

KEYWORDS

AR design, point cloud, capture and editing

ACM Reference Format:
Zeyu Wang, Cuong Nguyen, Paul Asente, and Julie Dorsey. 2023. PointShopAR: Supporting Environmental Design Prototyping Using Point Cloud in Augmented Reality. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems (CHI '23), April 23–28, 2023, Hamburg, Germany*. ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3544548.3580776

1 INTRODUCTION

Environmental design is a discipline that involves externalizing and iterating changes to the physical environment to meet design goals [50]. An architect designing an open-space living area might want to visualize how taking down a wall would affect the existing room arrangement, and a landscape designer might want to show how proposed plantings will look given the site plan of a garden. Consumers also face these issues in everyday life. A homeowner might want to see the effect of changing a regular door to a French door or introducing convertible furniture.

In this work, we focus on supporting users, regardless of their design expertise, in environmental design prototyping. Unlike application prototyping, where users modify virtual assets to explore *digital* design options, we use the term "prototyping" to describe the task of exploring changes to a *physical* environment. Environmental design prototyping is critical to ensure various spaces are functional and efficient, but design prototyping for a physical space can be costly and difficult. Users need to externalize and iterate design ideas quickly, but the size and rigidity of a physical environment pose many challenges. Physically realizing a design can be costly if it involves heavy or irreversible manipulation—a single person might not be able to move a large piece of furniture; a wall that was removed cannot be easily rebuilt. Modifying a digital replica of the physical environment would have been easier, but current practice is tedious and has a steep learning curve. It involves a large collection of hardware and software, such as using a dedicated 3D scanner, processing scan data to create a textured mesh model, editing the model in some desktop 3D software, and exporting the design for mobile viewing on site.

Inspired by Ben Shneiderman's design principles for creativity support tools [45], we seek to develop a novel workflow for environmental design prototyping that 1) is easy for novices to use, 2) reduces frustrating file conversions, and 3) enables users to rapidly generate multiple alternatives.

The first and fundamental requirement for such an environmental design prototyping workflow is that users should be able to capture the physical environment easily. This is made possible by consumer augmented reality (AR) and depth sensing technologies, which are becoming more and more accessible but still underexplored in current environmental design tools. A LiDAR-equipped iPad can capture an entire home in 3D. Although the quality of the resulting scan may not be as good as that created by a professional laser scanner, a rough 3D replica can still provide sufficient spatial context for environmental design prototyping. AR viewing is also supported out of the box. Users can quickly view the scan by walking around and get an approximate sense of the design as if they had been made in the real space [49].

Current practices often include many file conversions, causing great frustration for designers. The raw data from a 3D scanner is usually a point cloud. For visualization purposes, the traditional graphics pipeline needs to transfer the raw data to a desktop machine for further processing, such as extracting a polygon mesh from the point cloud and computing texture mapping. This is notoriously complicated because it requires great technical expertise, involves multiple algorithms and software packages, and takes significant processing time. As a reference, one of the most optimized 3D scanner apps, Polycam [40], takes about three minutes to produce a textured mesh for a bedroom, while a custom program can produce a colored point cloud for the same space in less than ten seconds. The additional processing time required for mesh generation is simply too long for an iterative turnaround cycle, especially if the environment is large or the design requires multiple scans of different areas and objects.

Traditional mesh editors also prevent users from quickly creating multiple design alternatives. Even though carefully reconstructed and texture-mapped meshes can be geometrically and photometrically accurate, and they are widely used in the film and video game industry, editing such meshes takes tremendous effort and is not amenable to rapid environmental design prototyping. Simple Boolean geometric operations, such as adding to a scene by performing a union with a new object, or removing part of a scene by intersecting with another object, are a great challenge that is still being actively investigated by an extensive body of graphics research [35, 54]. Even the best approaches can fail if the underlying mesh has problems like failing to be watertight. Performing direct vertex manipulation on a mesh using current desktop 3D software is straightforward, but the results often do not reflect users' creative intentions effectively, as illustrated by the famous short film *Cubic Tragedy* [10]. Mesh editing has a steep learning curve and can cause frustration, as it is prone to errors in manipulating vertex connectivity and in texture mapping.

Based on our analysis of the three design principles above, this paper mainly focuses on two research questions. First, we explore whether point clouds are suitable for rapid environmental design prototyping, with the goal of providing a more user-centric experience than mesh editing. Raster tools like Adobe Photoshop [3] are more widely used for editing captured images [14], whereas vector tools like Adobe Illustrator [1] are more suitable for rigid and precise editing. Inspired by this, we choose point cloud as the "raster" representation for editing 3D scans because of its freeform structure and its amenability to fast iterative design prototyping. Second, we investigate whether integrating both point cloud capture and editing in a single in-situ AR workflow could help users explore design ideas quickly without having to physically modify an environment. To answer these questions, we develop PointShopAR, a tablet-based system with integrated point cloud capture and editing to support environmental design prototyping.

PointShopAR leverages the LiDAR scanner available on consumer tablets like iPads to allow users to quickly capture their spatial context as colored point clouds. Users can use the same device in an editor mode to perform a variety of point cloud editing operations, inspired by a formative study, such as selection, transformation, hole-filling, drawing, and animation. PointShopAR provides an integrated capture and editing experience in situ, helping users quickly explore spatial design ideas. Our user study shows that point clouds can represent the design context effectively and serve as a versatile medium for a variety of design tasks in AR, including the design of objects, interior environments, buildings, and landscapes.

In summary, this paper makes the following contributions:

- A system that integrates the capture and editing workflows, in which the faster feedback loop enabled by eliminating the 3D model construction allows designers to experience and iterate the design rapidly.
- A demonstration that the point cloud is an effective prototyping primitive, making it easy and intuitive to capture and edit a 3D spatial context.
- A variety of application scenarios for our system showing its versatility and effectiveness, including object design, interior design, architectural design, and landscape design.

2 RELATED WORK

2.1 Interacting With an Environment Capture in VR/AR

An overarching goal in Human-Computer Interaction is to bridge the gap between the physical and virtual worlds. Many solutions have been developed, sitting on a virtual-physical reality continuum defined by Milgram and Kishino [29]. A prominent concept among these solutions is Mediated Reality [26], which proposes that users should be able to modify an environment instead of just augmenting it. Subsequent work has instantiated this concept using different context representations like video, semantics, mesh, and voxel data. Diminished Reality [27] and Altered Reality [23] focus on removing objects from a live video feed in an AR headset. Although this video feed can show a faithful reconstruction of the real environment, real-time techniques for modifying videos are still challenging. Another line of work [41, 43] uses a semantic representation for environment capture, replacing real objects with virtual ones. While this can make editing easier, it lacks realism and fails when semantic information is unclear. Alternatively, researchers explored Mediated Reality concepts by modifying 3D mesh and voxel reconstructions of the environment. In SceneCtrl [53],users can select, cut, move, and delete objects, and the edited scene is rendered back into an AR headset. However, SceneCtrl does not support rapid capture and editing of the environment because it requires substantial preprocessing time. Remixed Reality [25] uses live voxel data to create a virtual space that mimics the real space and is still fully editable. Nonetheless, it requires an elaborate setup of a camera array to capture and update voxels in real time.

Our system is similar to SceneCtrl and Remixed Reality in that it allows users to interact with a 3D reconstruction of an environment and use the 3D reconstruction as a proxy for editing the real environment. However, instead of focusing on constructing a new reality experience, we focus on creating a design medium where users could prototype changes to a physical environment and experience the design via augmented reality. Our key difference is an integrated point cloud capture and editing workflow. Instead of relying on a carefully installed camera array or an expensive head-mounted display (HMD) like a HoloLens, we only require a consumer tablet communicating with a laptop server to let users quickly capture, store, visualize, and edit point clouds. This approach gives users the flexibility and mobility to "remix" arbitrary physical environment around them. Then, inspired by Remixed Reality, and using findings from a formative study with designers, we identified and developed a set of point cloud editing operations that include selection, edit, draw, and keyframe animation. Finally, we integrated capture and editing together into a single workflow. Our user study shows that this integration enables users to easily capture their spatial context and rapidly explore environmental design ideas.

2.2 Design Prototyping Tools in VR/AR

Designing creative tools for immersive authoring is an active research area. Spatially-tracked hardware like HMDs, glasses, or LiDAR tablets carry unique affordances for 3D content creation. They support six-degree-of-freedom direct interaction out of the box. They also render 3D contents on a spatial-tracked display, giving a faster feedback loop than 3D workflows on a desktop. A major theme in recent work in HCI is letting users prototype interactive applications with immersive hardware. Project Pronto [22] , Rapido [21], RealitySketch [46], and ProtoAR [34] use augmented video to help designers quickly create novel AR designs. SpatialProto [31], 360Proto [33], and VRFromX [17] let users capture real-world objects and augment them with virtual elements to prototype interactive virtual and augmented reality experiences. XRDirector [32] enables multiple users to collaborative construct an interactive VR experience, where each user can control a different virtual component such as camera, lighting, characters, or interactive objects.

PointshopAR is also an immersive authoring tool, but our focus is not creating new UX or interactions in VR/AR applications. Instead, we focus on designing for a physical space. We want users to be able to solve physical design problems like adding functionality to existing furniture, creating a new layout for a living room, or testing a stage setup for a theatrical play. Our key insight is to let users manipulate a virtual replica of a physical environment and preview it in AR. This approach lets users visualize spatial changes quickly without doing actual physical work. We chose point cloud as the virtual representation for capture and editing. The freeform structure of point clouds makes the workflow lightweight and iterative. Users can scan, remix, and extend existing design using point clouds from different sources, since they can all be captured and edited in the same authoring device.

2.3 Modeling in Context

In 3D design, context information about the physical environment is often incorporated in the modeling process to help artists create with real-world constraints. This process is termed "modeling-in-context" [20]. To acquire context data, designers use sensors to capture certain information about the environment. For example, in Insitu [38], researchers use GPS sensors attached to stakes that were anchored on a development site to infer terrain information for architectural design. Environmental data is then rendered on top of photographs of the development site, and designers can create on top of these augmented images. Other 3D modeling work has also adopted a similar "modeling with photographs" approach [13, 20, 24]. IKEA recently released a LiDAR-enabled AR app that allows removing existing furniture from a captured image and then showcasing their products [12].

A photograph, however, provides only a single viewpoint and might not be ideal for design tasks performed in a large environment. Researchers have explored reconstructing an environment as a 3D polygonal mesh to aid 3D design. Recent advances in mobile depth sensing and AR technologies let users do 3D scanning with mobile devices instead of expensive and bulky laser scanners. This enables off-the-shell apps that can capture large parts of the surrounding environment [40, 44]. Environment capture with everyday devices creates new opportunities to develop contextual 3D modeling applications. Researchers have explored a growing list of novel design applications, such as remote AR layout design [49], product sketch [18], and in-situ 3D shape creation [15].

However, these projects focus primarily on adding new shapes to a captured context. We are interested in the more complex task of modifying that context to help users rapidly answer environmental design questions. We believe that point clouds are a promising representation for this task. They let us render the environment without having to perform the slow and expensive step of texturing the mesh captured by the iPad scan. Moreover, they offer an unaltered raw form of the context data to work with. They have been used extensively in large-scale design projects where scale and measurements are important, such as site planning [39, 48] and data visualization [52]. We aim to bring the benefits of point cloud capture and editing to everyday users. Non-professionals usually find it relatively easy to edit 2D images by selecting, copying, pasting, and transforming pixels, but most find similar operations on structured vector content challenging. PointShopAR brings the simplicity of image editing to the third dimension through point clouds and AR.

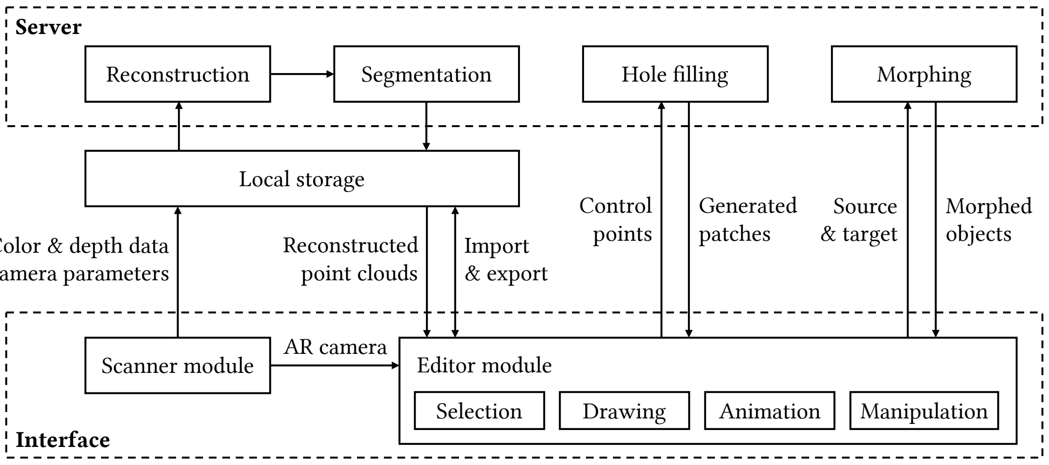

Figure 2: The system pipeline for PointShopAR. The user first uses the scanner module to record color and depth data and tracked camera parameters, which are sent to the server for point cloud reconstruction and semantic instance segmentation. The user can then use the editor module to modify the scan using operations like importing and exporting, selection, manipulation, drawing, animation, hole filling, and morphing. Hole filling and morphing are supported by their corresponding modules on the server.

3 DESIGN CONSIDERATIONS

To inform our system design, we conducted an interview-based formative study to understand how designers use their spatial context for environmental design. We invited six designers (P1–P6) to participate. All held advanced degrees in architecture and four had been working for top architecture firms. We carried out open-ended interviews with two goals: understanding how designers capture and edit spatial context when designing for a physical environment, and identifying challenges in their current workflow and what limitations they are facing with their current tool set. We asked participants to describe recent design projects, how they used spatial context in their design, and how they edit 3D scans to explore environmental design ideas.

The interviews helped us understand that spatial context plays a very important role in design for projects of different scales. For example, building a public school requires an urban plan, building a residential house requires elevation and vegetation maps, and interior design requires a floor plan and furniture dimensions. However, all designers said that it is very difficult to capture spatial context due to the lack of accessible tools. Context representation is also limited—site maps are highly abstract, photos and videos do not support changing to a new viewpoint, and 3D scanning is expensive and time consuming. P1 considers site visits to be important and prefers in-situ design because he no longer has access to the immersive spatial context back in the studio.

Editing the spatial context is also challenging. P2, P4, and P5 rely on heavy-weight desktop software like AutoCAD [9] and Rhino [8] to manipulate 3D data. P1 said that a common challenge with these tools is the mindset switch in design. Using them to manipulate 3D scenes on a desktop computer means learning and finding the right feature for 3D operations. P3 wanted to modify an environment by selecting and manipulating walls and ceilings, but having no streamlined design prototyping workflow made this very difficult. P4–P6 said that they still prefer to sketch on paper to explore and prototype environmental design ideas. All reported familiarity with AR technology and its potential opportunities for architectural design. They mentioned a top use case of AR is to let designers experience their design in AR to get a more realistic sense of how the physical environment was altered.

The formative study showed designers' desire for a faster capture-editing feedback loop, more freeform 3D editing, and being able to experience their design in AR, all of which can let them evaluate and iterate their design choices more rapidly and facilitate environmental design prototyping. Based on these interviews, we distill our design considerations for developing the PointShopAR system as follows:

- Users must be able to capture their spatial context easily and quickly. Our capture representation must be amenable to rapid prototyping.
- Users must be able to edit the 3D capture using intuitive and direct interactions, such as selection, 3D manipulation, hole filling, and drawing.
- Users must be able to experience and iterate their design in situ and animate objects to illustrate new functionalities in the design.

4 THE PointShopAR SYSTEM

We used findings and design considerations from the formative study to motivate the design of the PointShopAR system. PointShopAR uses point clouds as the underlying representation, and integrates the capture and editing processes, as shown in Fig. 2. Our front-end interface was built for an iPad Pro with LiDAR and implemented using Apple's Swift, ARKit [4], RealityKit [5], and SceneKit [6]. The application connects to a back-end server for data processing; we use a consumer laptop with 8-core Intel Xeon E5-1620 3.60GHz and Ubuntu 18.04.6 LTS. The server was implemented using Python, Flask [42], Open3D [37], and scikit-spatial [16]. We chose to use a server because it is more convenient to implement some functions for point cloud processing in Python than in Swift. Current iPad Pro models are comparable in power to a laptop, so one could instead do all the computation on the iPad to avoid network access.

4.1 Point Cloud Capture, Generation, and Rendering

Capturing 3D spatial context is the foundation for users to work on their design. We leverage the LiDAR scanner available on recent iPad Pro tablets to capture the 3D environment in the form of point cloud. We developed a scanner module in PointShopAR capable of capturing common design contexts like objects, interior environments, and outdoor environments. The scanning process constructs a rough wireframe mesh and overlays it on the current camera view in real time, indicating which part of the environment has been captured. The user taps a "Finish Scan" button when the mesh includes all areas that the user wants to capture. To support an iterative design process, we let the user scan multiple times during one design session—they can alternate back and forth between scanning and editing as needed. After scanning is complete, it takes 2 seconds to about 30 seconds to load the point cloud in the editor mode depending on how large the scanned environment is.

Using Apple's ARKit, we record videos from the RGB camera and the depth camera for each scan. We also record estimated camera parameters and a depth-estimation confidence map for each frame of the recording. Once the user finishes the scan, all the recorded data is sent to a web service to generate a point cloud. It then returns the data to the iPad for rendering. Full details of the generation and rendering methods are included in Appendix A. A key implementation feature is storing the points in an octree structure that enables efficient point selection and manipulation.

4.2 Point Cloud Editing

4.2.1 Bounding Volume-Based Object Selection. Fig. 3 shows how a user selects part of the point cloud representing an object. They first tap a button to turn on selection mode and tap on the screen multiple times to select an initial set of 3D points. Often this involves moving around the area to see the object from different viewpoints. The user then taps the selection button again to confirm the selection. Our system automatically creates a bounding volume that covers the object of interest. The user can tap a face of the bounding volume and drag it for fine tuning. Tapping a spot twice selects the back face for adjustment, eliminating the need to move to a spot where the back is visible—and in some cases, such as selecting part of a wall in a building, such a spot might not even be accessible. The user can also use touch-based interactions to move, rotate, and scale the bounding volume as described in Section 4.2.5.

Once the user has finalized the bounding volume, they have three choices: duplicate the points to make a new point cloud object, remove them from the original point cloud object, or separate them from the original point cloud object—effectively duplication combined with removal. The interface lets the user cycle through all point cloud objects in the scene, and they can use a highlight button to make the active point cloud object a uniform orange color to better understand the scene.

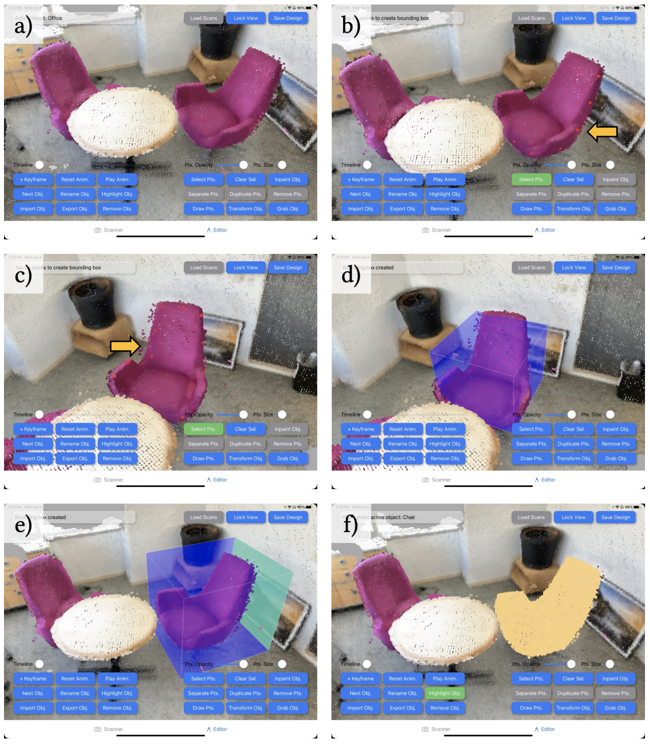

Figure 3: Bounding volume-based object selection. a) Scanned point cloud. b) The user taps on the chair to select 3D points, which the system colors in red. c) The user moves to another view and keeps selecting points. d) A bounding volume is created after the user finishes point selection. e) The user adjusts faces of the bounding volume to include parts of the chair that were missed by the initial volume. f) The user highlights the selected chair in yellow to verify that is has been correctly selected.

4.2.2 Efficient Point Selection. The selection process above and other processes described below rely upon being able to quickly find the 3D point that corresponds to a tap on the screen. We compute a ray from the current camera position through the tap location and see whether it intersects with any 3D point. The projection and view matrices of the current camera let us project the 3D points of the active point cloud object onto the screen. We then find a point with a projected 2D screen coordinate that is close to the tap location. Since the user is more likely to select points that are closer rather than farther away, we do not just pick the point with the smallest 2D distance; instead we find 10 candidate 3D points with screen coordinates closest to the tap location, and pick the one that is closest to the camera.

The octree greatly accelerates this process, and its use is described in Appendix A.3. It takes about half a second for a tap to select a 3D point from a point cloud object containing hundreds of thousands of points.

4.2.3 Import and Export. A benefit of integrating point cloud capture and editing is being able to bring scans of different environments together for rapid prototyping. After a group of points has been separated from a scan, the user can assign it a name and export the group as an individual object in the library. Later, possibly in another physical environment, they can import these point clouds from previously scanned objects. This lets users quickly explore design ideas. For example, one could virtually move an existing couch to a new house to see if it fits in the new space.

4.2.4 Hole Filling / Inpainting. Removing an object from a scan usually leaves a hole in the original point cloud object. Holes may also exist because of incomplete scans. We developed an point cloud inpainting algorithm with an intuitive interface to fill the holes (Fig. 4).

The vast majority of holes are on one plane or are where two or three planes meet. The user first taps an "Inpaint Object" button, which changes its label to "1st Plane." Then they tap the screen multiple times to select 3D points around the hole on one of the bounding planes. They tap the button again, changing its text to "2nd Plane," and select points on a second plane if there is one, and follow this with the third plane. If there is no second or third plane, they just tap the button with no intervening point selection. The system highlights the selected points in different colors depending on their plane.

This results in one to three lists of user-specified control points, depending on the type of the hole. Once the user confirms, the iPad sends the point cloud and the control points to the server. The server runs an inpainting algorithm, described in Appendix A.4, and returns a patch of points that the iPad adds to the scene as a new object.

4.2.5 Touch-Based Object Manipulation. After the user creates a new point cloud object by separating or duplicating points from the original scan, they need to manipulate it in the 3D space. We implemented common touch-based interactions in our system, such as a pan gesture for moving, a rotation gesture for rotation, and a pinch gesture for scaling. We found that sometimes users intended different manipulations using the same gesture, such as having a pinch gesture scale along all three axes, or just along two axes to preserve height. Therefore, we added a gesture control panel that lets the user specify what gestures are active and what transformations pan and pinch perform. The rotation gesture maps to rotation around the vertical axis, the pan gesture can map to horizontal or vertical movement, and the pinch gesture can map to scaling along one, two, or all three axes, depending on the panel settings.

4.2.6 Grabbing-Based Object Manipulation. While touch-based interactions are standard for object manipulation, they fall short in more freeform manipulation. Specifically, specifying an arbitrary axis in 3D to use for rotation is quite challenging. We solve this problem by leveraging the iPad's pose tracking information, and mapping any change in the iPad's pose to a corresponding change in the selected object's pose. We call this function *grabbing* (Fig. 5). The user presses and holds a grab button and moves the iPad. The pose of the point cloud object is updated from iPad's tracked pose in real time until the grab button is released. During the grabbing process, the camera view is fixed so the user can better observe the

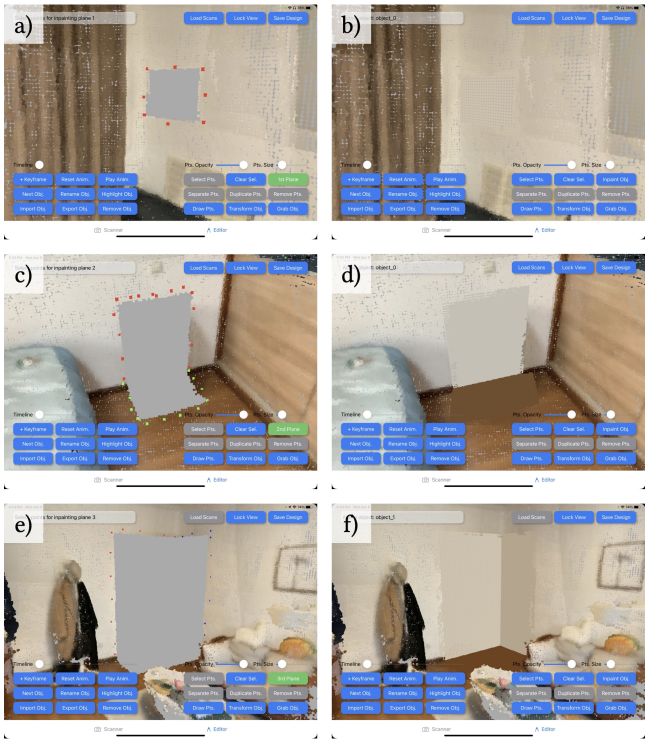

Figure 4: Hole filling examples. a) A hole in a wall. c) A hole where a wall meets the floor. e) A hole at the corner of the room. b, d, f) Inpainting results.

pose change of the point cloud object. Grabbing manipulates the point cloud object in the same physical scale as the iPad, enabling intuitive editing operations such as lifting a lid or opening a door.

Geometrically speaking, we record the original pose $[R_c|t_c]$ and the current pose $[R'_c|t'_c]$ of the AR camera as well as the original pose $[R_o|t_o]$ of the point cloud object. R is a 3×3 rotation matrix and t is a 3×1 translation vector. During grabbing, the pose $[R'_o|t'_o]$ of the point cloud object is computed as follows.

$$R'_o = R'_c \times R_c^{-1} \times R_o, \qquad t'_o = t'_c - t_c + t_o.$$

4.2.7 Midair Point Drawing. During the formative study, we found that some designers wanted to be able to create new 3D structures using freeform drawing. Therefore, we included a feature in PointShopAR that lets the user raw points in mid air using their iPad. To make the system lightweight, we did not require a separate controller or any device other than the iPad. The user can pick a color and start drawing by moving the iPad and pointing it in different directions. Our system creates and renders new points in real time, one meter in front of the camera along the camera front vector in the world coordinate system. Once drawing is complete, the user can manipulate drawn points using gestures and grabbing, in the same way as manipulating scanned points.

4.2.8 Point Cloud Animation. The final feature of PointShopAR is point cloud animation. After the user creates point cloud objects in the scene by scanning, separating, importing, and drawing, they can animate their design. We provide a simple interface that lets the user

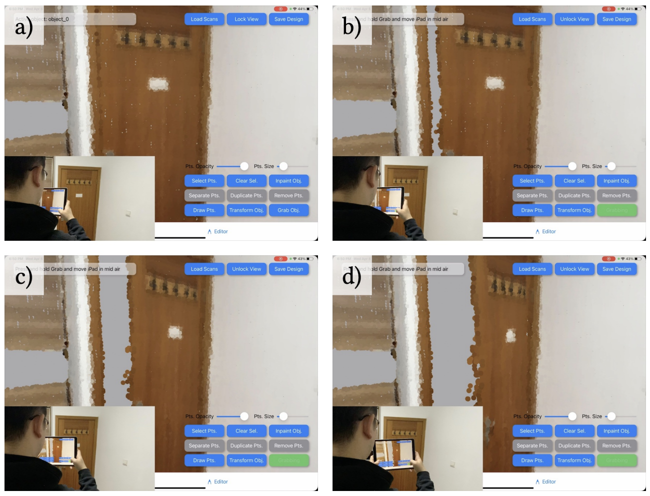

Figure 5: Grabbing-based object manipulation. a–d) The user rotates the tablet (inset) while holding the "Grab" button to simulate the effect of door opening.

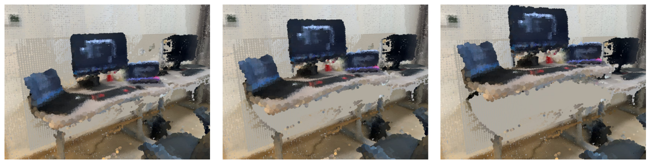

Figure 6: Animation example. The tabletop rises so the user can experience the optimal height for working seated and standing. Object position, orientation, scale, opacity, and point size can be animated.

create keyframes along a fixed-duration timeline. The user positions a slider to a point on the timeline, applies transformations to the objects, and taps a button to add a keyframe. After adding several keyframes, they can play the animation. Five attributes—position, rotation, scale, opacity, and point size—of each point cloud object are animated using linear interpolation. This enables visualizations of processes like opening a door or curtain, making an object appear or grow, and manipulating articulated furniture.

5 STUDY 1: USABILITY OF PointShopAR

With an initial implementation of PointShopAR, we first conducted a remote user study to assess the usability of our system and collect feedback on various system features for point cloud capture and editing for improvements.

5.1 Study Design

The purpose of this study was to evaluate how users could use our system to capture and edit point clouds. We focused specifically on observing how users used the integrated point cloud capture and edit workflow, and how individual features helped designing in a physical environment. Due to COVID-19 restrictions, we conducted this study remotely. Since participants were in different remote environments, we did not instruct them to create any specific design. Instead, we asked them to use PointShopAR to envision a new physical design in their current living space. We encouraged participants to try to use all features of PointShopAR including scanning, separating points, importing objects, hole filling, object manipulation, midair drawing, and animation.

5.1.1 Participants. We invited seven participants (Pa1–Pa7) who had interest but little experience in environmental design. Pa1–Pa4 are graduate students studying computer graphics, Pa5 and Pa7 are user experience designers, and Pa6 is a management consultant.

5.1.2 Procedure. We conducted the study remotely via video conferencing. All participants used an iPad Pro with LiDAR to test our system. First, participants watched a video tutorial showing how to use each feature of the system. Next, we held a 10-minute training session to familiarize them with system functions with a trial task. We then asked them to capture their physical environment and edit the scanned point cloud for up to 30 minutes. Finally, they were asked to complete a survey and interview to give feedback on our system.

5.1.3 Measurement. We recorded the video meetings and asked participants to record their iPad's screen during the study for further analysis. The survey had three sections: 1) personal information, 2) experience rating, and 3) system features rating. In the first rating section, we asked participants to rate how much they agreed with six statements on a scale of 1–7, with 1 being "strongly disagree" and 7 "strongly agree." Fig. 7a) shows the statements (Ra1–Ra6) and the distribution of responses for each.

In the second rating section, we asked participants to rate how useful each feature was in helping them complete the design on a scale of 1–7, with 1 being "not useful at all" and 7 "extremely useful." Fig. 7b) shows the features (Rb1–Rb8) and the distribution of responses for each. We then conducted an interview with each participant to collect feedback on their experience with PointShopAR.

5.2 Results and User Feedback

5.2.1 Overall Usability. Fig. 7a shows an overview of participants' ratings of their experience. Overall, participants were positive about their experience using PointShopAR (Ra1). They were able to capture their environment and edit the scanned point cloud even though we were not there to provide on-site assistance. This finding is significant considering the number of features and the complexity of the design workflow that involves both 3D point cloud scanning and editing.

All participants were satisfied with the design exploration they conducted using PointShopAR (Ra6). Three (Pa1–Pa3) did the study in their work space and the others (Pa4–Pa7) did it at home. Their editing activities include separating objects from the point cloud, swapping two desks, replacing a chair, removing a cabinet, and decorating the room with an animation. On average, participants spent 58 seconds (σ = 17 seconds) on scanning their physical environment and 17 minutes 39 seconds (σ = 3 minutes 5 seconds) on editing the point cloud.

When asked to evaluate using point cloud as a representation for designing an environment, participants liked the ease of capture (Ra2). Participants also liked having a virtual proxy of the physical space to edit, so they did not have to worry about making actual physical changes (Ra4). More importantly, the low-fidelity render of point clouds was deemed sufficient to help participants understand the physical spaces for their design tasks (Ra2).

These results suggest that participants were able to use PointShopAR for an environmental design task. They were able to use our proposed workflow from end to end, including both capturing point clouds as a proxy of the environment that they wanted to design, and using editing tools in PointShopAR to explore some design ideas.

5.2.2 Suggested Improvements. We also asked participants to rate individual features in PointShopAR (Fig. 7b) and asked for their feedback on how to improve the system in a post-study interview. We identified three main areas for improvement.

First, for point selection, even though all participants considered bounding volume object selection useful, some felt that having to manually adjust the boundary of the volume to create a desirable selection could be tedious. P3 suggested performing semantic instance segmentation on the scanned point cloud, so if the user would like to select points on a chair, other points in the bounding volume that belong another object can be easily filtered out.

Second, midair point drawing received most non-positive ratings. Only three participants found this feature useful in their design (Rb7). This is because simply drawing points one meter in front of the iPad created many challenges. P3 thought that it was difficult to draw in mid air because he could not anticipate where points would be placed in 3D. They suggested adding a planar canvas before drawing points, similar to existing work in AR design [7, 19, 22].

Finally, participants pointed out that PointShopAR did not let them modify the shape of an existing point cloud scan, for example, to bend a rectangular dining table into a curved table. Pa1 and Pa6 wanted to move points freely in 3D space to morph and reshape points.

6 SYSTEM IMPROVEMENTS

We improved the PointShopAR system based on participants' feedback from the remote study by adding three new features and several minor enhancements.

6.1 Semantic Instance Segmentation

We added a semantic instance segmentation module using a state-of-the-art deep learning method combining bottom-up grouping and top-down refinement called SoftGroup [47]. Whenever it generates a point cloud, the server assigns each point an instance label, which can help filter instances when the user selects points using a bounding volume, as shown in Fig. 6a) and b). This instance segmentation module requires about 10 more seconds of processing time on the server.

6.2 Drawing on Canvas

We replaced our midair drawing function with drawing on canvas. In Fig. 6d), the user uses the location and orientation of the tablet to place an infinite planar canvas. They can then move back and use touch to draw points on the screen, which are then projected onto the planar canvas, as shown in Fig. 6e) and f).

6.3 Point Cloud Morphing

Participants in our first study expressed a strong desire to reshape point clouds. Reshaping points allows users to create new shapes, for example, creating new vegetation variations for a landscaping project by bending or pulling some points in an existing point cloud scan of a tree or bush. However, a significant number of new editing operations would have to be integrated into PointShopAR to fully support freeform 3D sculpting as done in commercial solutions like ZBrush [28] or Adobe Medium [2]. To help users focus more on iterating ideas and less on sculpting operations, we added a morphing interface that generates an intermediate point cloud between source and target clouds, as shown in Fig. 6c). We begin by normalizing both clouds to fit in a unit cube. For every point in the target cloud, we find the closest point in the source. The position and color of each point in the morph can then be generated using a linear interpolation. The intermediate cloud becomes a new object that can be inserted into a scene. Morphing leverages a key characteristic of point clouds: their freeform structure. It lets designers create new shapes by remixing and combining existing shapes scanned from the physical environment or imported from an online library, such as morphing different plants as shown in Fig. 10f).

6.4 Additional Enhancements

We made other minor changes to the system to improve the overall user experience. Previously, the user had to tap a button to cycle through the scene's objects one by one. We added a popup window listing all objects, making it is easier to select an active object. We changed the highlight color from pure orange to a blend of green and the underlying original color, letting the user see the texture of a highlighted object. We also added a function to preview the selection, which highlights which points will be selected before they are actually separated from the original point cloud.

7 STUDY 2: USING POINTSHOPAR FOR ENVIRONMENTAL DESIGN

We evaluated the improved PointShopAR system with a second in-person user study to understand how our integrated point cloud capture and editing workflow could support users to rapidly prototype environmental design ideas in AR. We also collected subjective feedback on original and new features of PointShopAR, and observe participants' behavior when performing an open-ended design task modifying their physical environment. The research question we tried to answer through this study is whether the integrated scene capture and editing and the point cloud representation are helpful for rapidly prototyping environmental design ideas in AR.

7.1 Study Design

Our first study showed the usability of our system for novice users, but the remote setup did not allow us to observe user behavior in their design process, so our second in-person study aimed to evaluate how PointShopAR supports rapid prototyping of environmental design ideas. We conducted this study in a university building. Participants picked a location, developed an open-ended design concept, and realized it on site using our improved system on a LiDAR-equipped iPad. Similar to the first study, we did not instruct

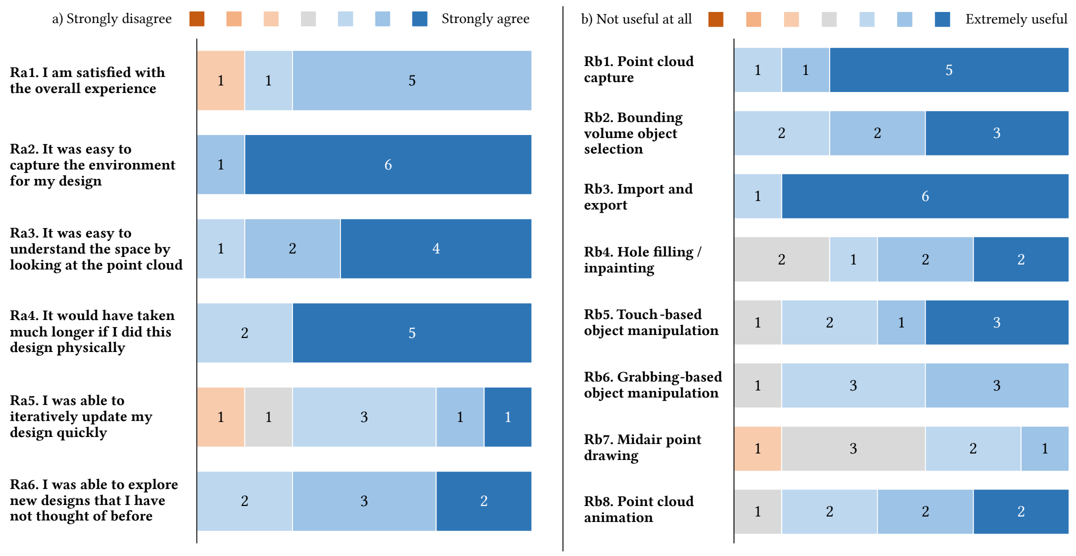

Figure 7: Study 1 results. Ratings of a) the user experience and b) each feature in our remote usability study. The color of a bar represents how much participants agreed with a statement or how useful they found a feature. The number in the bar represents the number of participants who submitted the same rating.

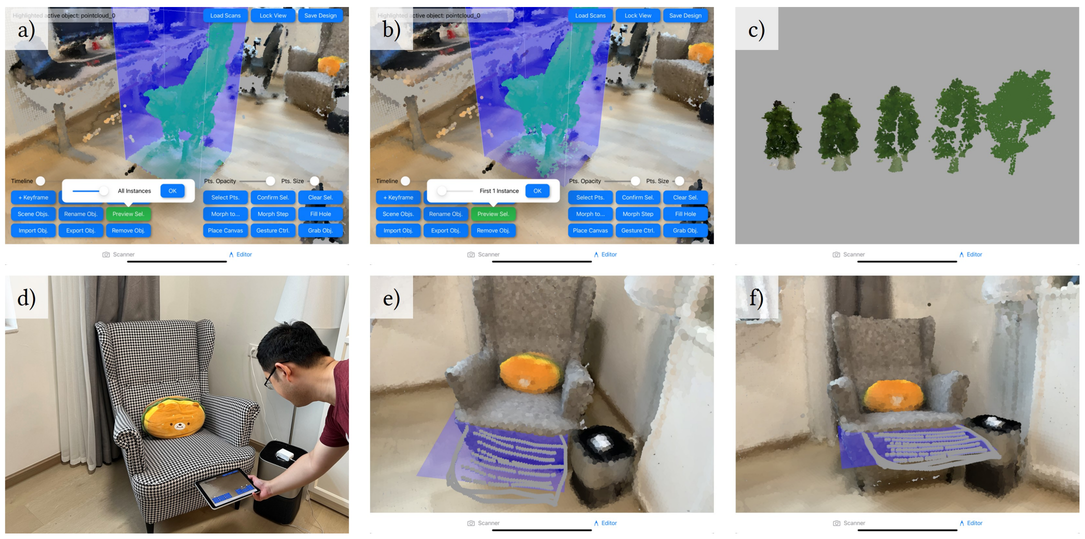

Figure 8: System improvements. a) All points in the bounding volume are selected, highlighted in green. b) Only points of the same instance are selected, leaving out points on the floor when selecting the chair. c) Morphs between a scanned plant and an imported tree. d) Before drawing, the user first uses iPad's physical location to place a planar canvas. e) The user then draws a foot rest appended to the coach. f) The same drawing viewed from another angle.

participants to create any specific design. Participants were encouraged to explore and use all features in PointShopAR, including the improvements that we have made to the system in Section 6.

7.1.1 Participants. To better evaluate how PointShopAR supports environmental design in practice, we recruited ten participants (Pb1–Pb10) with experience in art and design. Eight were graduate students studying computational media and arts and two (Pb8 and Pb10) were university lecturers with extensive experience in art and design.

7.1.2 Procedure. In this in-person study, we first showed each participant how to use each feature of the system, then ran a training session in which they performed a trial task to become familiar with user interactions in PointShopAR. The trial task was importing a scanned trash can, separating its lid, and creating an animation to lift the lid. Then we asked participants to develop a spatial design concept based on their physical environment and carry it out on the iPad. During the design process, we encourage them to think out loud and ask questions, so we could follow their design choices. Finally, we asked them to complete a survey and an interview to evaluate our system.

7.1.3 Measurement. We recorded the conversations and the iPad's screen during the study for further analysis. We used the same survey as before with two more questions rating the new semantic instance segmentation (Rb9) and point cloud morphing (Rb10) features. After they completed all ratings, we again performed an interview to gain more insights into their environmental design prototyping experience using PointShopAR.

7.2 Results and Discussion

Table 1 shows an overview of the types of design carried out by each participant and the time spent in the study. All participants came to a university building in which they could pick an environment of interest and develop their design concept such as rearranging furniture to experiment with a new layout or modifying building structures to repurpose a space. On average, participants spent 48 seconds (σ = 27 seconds) on scanning their physical environment and 21 minutes 26 seconds (σ = 7 minutes 6 seconds) on editing.

7.2.1 Integrated Capture and Editing. Fig. 9 shows an overview of participants' ratings on their experience in the task. Participants were positive of the overall design experience using PointShopAR (10/10 in Ra1).

A prominent benefit of PointShopAR is that it lets users rapidly prototype changes to the physical environment. Most participants agreed that without PointShopAR, it would have taken them much longer to complete the design task physically (9/10 in Ra4). Participants also agreed that they were able to explore new ideas they have not thought of before (10/10 in Ra6). Pb2 explained that "*you are able to move things around in AR that you cannot do physically.*" And Pb3 shared similar remarks, saying that PointShopAR "*is more accessible by saving cost and time and makes me be more creative in this 3D space.*"

The key technical component that enables this novel design experience is the integrated point cloud capture and editing workflow. Participants unanimously agreed that it was easy to capture the environment for their design (10/10 in Ra2). Most participants also rated that they were able to update their design quickly (9/10 in Ra5).

The ease of capture encourages experimentation. Users do not need to worry about making mistakes because they could easily capture something new (Pb5, "*You can easily capture what you want*") or redo the capture (Pb10, "*It is really useful to see the environment immediately after scanning; I can redo the scan if not satisfied.*") More importantly, working with virtual replicas of the environment means that users are no longer restricted by the physical world. They do not need to worry about physical boundaries, weights, or distances that might affect their ability to conceptualize the design. It is faster to experiment with moving and changing large objects. It is also easier to compose, contrast, or make connections between physical objects in different places. Users can just scan and import them into the same AR scene.

In addition, the integrated capture-editing workflow provides users with a faster feedback loop (Pb1, "*it's more intuitive and it's easier to get the feedback after I changed something for design purposes.*") Being able to view the point cloud and make changes directly in AR means users can use their body movement to test out the functionality of the virtual design right after their edits. For example, Pb10 could get a sense of the optimal height of a dining table in their design by sitting next to it and examining its height in the AR view, as shown in Fig. 10f). Other participants experienced their design in similar direct manners: "*walk around and interact with objects*" (Pb1, Pb2, Pb6, Pb10) or "*you can easily move it around with your fingers*" (Pb9).

Finally, participants also shared remarks on how this workflow compares with a traditional desktop experience. Eight participants are familiar with various desktop 3D software. They all agreed that the design experience in PointShopAR feels much better than having to scan, import, and then edit on a desktop computer. Participants listed the following benefits: being able to walk around to understand the point clouds (Pb1, Pb2, Pb5), direct manipulation (Pb2, Pb4, Pb7), more intuitive feedback (Pb1), and quicker to scan and iterate their design (Pb8, Pb10). Pb8 felt that using desktop 3D software for production feels like a big commitment, whereas using PointShopAR encourages him to do more prototyping. He commented, "*you could experiment more in AR without having to do all the setup in Cinema 4D to get a preview.*" As professional artists, both Pb8 and Pb10 said that the rapid capture-editing-experience feedback loop can significantly reduce the gap between designers' eyes and viewers' eyes.

In summary, these findings highlight the key benefits of integrating both capture and editing into a single authoring workflow in PointShopAR. Users can easily and quickly capture their physical environment and immediately prototype changes in context. This approach provides a much faster feedback loop compared with desktop 3D software and encourages designers to explore and experiment with new ideas.

7.2.2 User Strategies. When asked to rate the usefulness of each feature in PointShopAR in the study task, participants ranked the following features highest: point cloud capture (10/10, Rb1), import and export (10/10, Rb3), hole filling (10/10, Rb4), and grabbing (10/10, Rb6). All participants found these features are essential in the design experience using PointShopAR.

These features made our system suitable for rapid prototyping of environment design. In about 20 minutes, participants were able to realize their design ideas even if they were first-time users and were in diverse physical environments. The average number of point cloud objects in their final design was 10 and the maximum is 26. The average area of their environmental design was 10×10 m^2 and maximum is 16×18 m^2. The height ranged from 3 m to 7 m representing one to two floors. This also showed the effectiveness of bounding volume object selection (9/10, Rb2) and semantic instance segmentation (9/10, Rb9), which participants frequently used to create new point cloud objects before manipulating them.

Most participants found touch-based object manipulation useful (8/10, Rb5). They could intuitively move, rotate, and scale objects using standard gestures. Pb9 expressed his desire for object snapping as in mesh-based AR apps. Pb10 suggested using more gestures

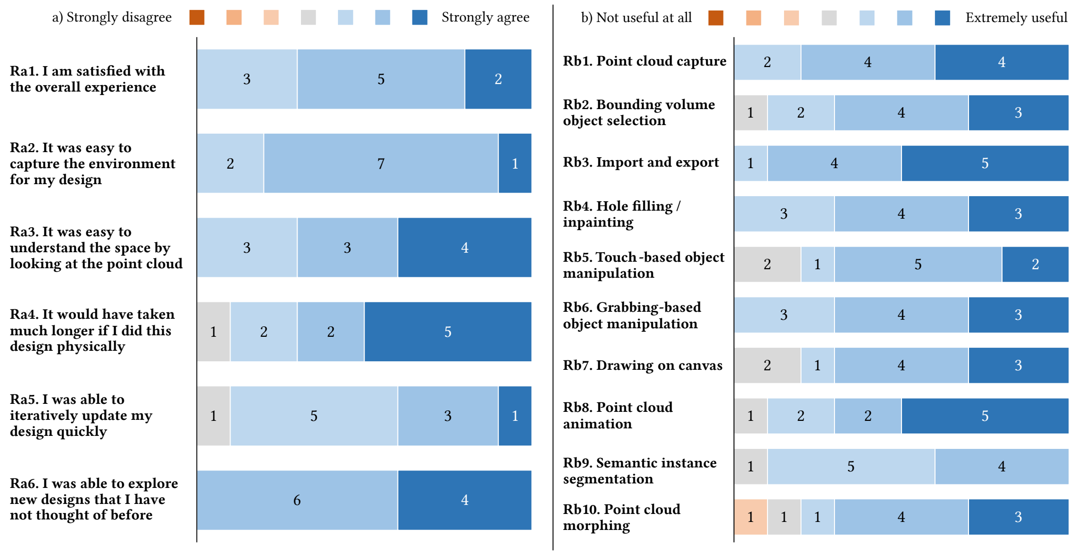

Figure 9: Study 2 results. Ratings of a) the user experience and b) each feature in our in-person user study. The color of a bar represents how much participants agreed with a statement or how useful they found a feature. The number in the bar represents the number of participants who submitted the same rating.

Table 1: A summary of time spent and the design concept developed by each participant in the in-person study.

#	Scanning	Editing	Design concept
Pb1	35 s	21 min 55 s	Shorten a long cabinet in a pantry so it does not block entrance to an office.
Pb2	33 s	18 min 25 s	Elongate a spiral staircase so it takes one circle to go to the next floor.
Pb3	36 s	19 min 38 s	Scan a person to animate in the hallway for conceptualizing an AR game.
Pb4	12 s	16 min 29 s	Repurpose a ladder at the construction site for placing office plants.
Pb5	1 min 6 s	17 min 39 s	Convert a conference room into a party space.
Pb6	24 s	12 min 17 s	Scan a person and create a miniature version on the pantry tabletop.
Pb7	1 min 34 s	23 min 37 s	Remove all furniture from a conference room and add a spiral slide.
Pb8	56 s	24 min 52 s	Add an elevator and a tree house next to a spiral staircase.
Pb9	32 s	20 min 35 s	Break the wall between two offices to make a larger office.
Pb10	1 min 28 s	38 min 48 s	Convert an empty atrium into a place for socializing surrounded by greenery.

instead of a button to switch between moving horizontally and vertically. Participants gave higher ratings to grabbing-based object manipulation (Rb6) compared to the first remote study. The reason could be that in-person instructions helped them better understand how grabbing mapped iPad's motion to a virtual object. Pb5 and Pb8 said that grabbing was well suited for more freeform finetuning of the position and rotation of an object.

Beyond object reorganization, a common operation in interior design tasks, participants also benefited from the support of rapidly removing existing structures and creating new shapes via scanning or drawing. This is useful for exploring environmental design ideas like knocking down a wall (Pb9) or remixing spatial elements with a second scan elsewhere (Pb4, Pb8). It was a significant improvement to replace midair point drawing with drawing on canvas. In Rb7, midair drawing in the first study only received 3 positive ratings out of 7 participants, whereas drawing on canvas in the second study received 8 out of 10. Users could have better control of what they drew on canvas rather than in midair and also a better sense of the physical location of the drawing because the canvas was placed based on the iPad's location. Participants used drawing for objects that would be hard to scan, such as text labels (Pb4, Pb5, Pb6) and symbols like an arrow (Pb1). Drawing was also useful to create specific objects in a more freeform way, such as a first aid kit (Pb7), a tree house (Pb8), and plants (Pb10).

7.2.3 Point Cloud as a Design Medium. The main premise of our work is that the point cloud can be a suitable representation for rapid environmental design prototyping. For this task, being able to understand the space is an important requirement for environmental design. In our study, we found that all participants agreed with the statement "*It was easy to understand the space by looking at the point cloud*" (Ra3). In our system, we let users capture point cloud data using a consumer LiDAR-equipped tablet. The resulting capture may have lower precision than an industry-grade laser scanner. However, the responses from participants show that they were able to understand the space well enough from the point cloud for their design activities. A possible explanation is that most of the design activities in our study were centered around room-scale environments. A very detailed 3D scan is typically not required unless users want to design with smaller objects like a keyboard. This finding resonates with previous work, where a low-quality textured mesh was found to be sufficient to help users carry out a remote AR layout task [49].

When asked to compare point clouds and meshes in representing the spatial context in the design task, most participants thought that point clouds are more amenable to rapid editing, whereas meshes are more suitable for final presentation (Pb1, Pb2, Pb3, Pb5, Pb6, Pb7, Pb8, Pb10). Pb10 described this as a tradeoff, "*meshes can have higher fidelity but you have to spend a lot of time*," whereas "*point clouds are suitable for prototyping because you can get very fast feedback*." Pb6 commented, "*point clouds feel less accurate, but they are interesting to look at and easy to modify*." Pb8 also commented on the technical challenges of mesh editing, "*It would have to generate the texture and might take a long time; editing meshes could be tricky if you'd have to cut many little edges*." Another unique advantage of the point cloud representation is that it supports freeform morphing, which also received positive ratings from participants (8/10, Rb10). Pb10 commented, "*in environmental design it's important to explore a continuum between two shapes, but few existing tools support this*." They extensively used morphing to generate different-looking plants to place around a place for socializing in the atrium.

These comments helped us gain valuable insight into the characteristics of point clouds and meshes from the user's perspective and the different tasks for which these representations are suitable. In contrast to the precise nature of mesh editing, point cloud editing is looser and error-tolerant, contributing to the rapid experience of externalizing and iterating spatial designs that PointShopAR provides. This is analogous to 2D vector and raster editors, which serve different design purposes. Just like paint tools are easier to use than vector editors because the raster representation is quick and easy to use and can directly deal with images scanned from external sources [14], point clouds are more suitable for 3D design prototyping than meshes. The inaccuracy and ambiguity of point cloud scans are acceptable and even can be beneficial in prototyping. We observed new user behaviors as a result of the faster feedback loop and the representation. For example, users remixed and repurposed objects creatively, e.g., bringing an elevator scanned at the other end of the hall (Pb8), turning a ladder into a plant shelf (Pb4), and turning a trash can into a stove (Pa6).

In summary, we found that point clouds are a suitable representation for rapid prototyping of environmental design ideas in AR. Even though a carefully reconstructed mesh with texture has better visual quality, users can sufficiently understand the spatial context from point clouds. Our integrated workflow allows users to easily and quickly capture and edit their physical environment, making point clouds an ideal prototyping medium.

7.3 Example Design Scenarios

Fig. 10 shows some example designs by participants in the second study. PointShopAR was able to support a wide variety of open-ended environmental design tasks described in Table 1. Participants quickly captured the environment of interest and performed a series of interior or functional designs. Their design concepts often involved modifying structures that were hard to change physically such as staircases, walls, and cabinets. PointShopAR supported rapid prototyping of environmental design ideas for spaces up to 16×18 m^2 and two-story high. Although PointShopAR was designed for environmental design, we found Rb3 and Rb6 used our system to scan a person and prototype an interactive AR game, which further broadens the application domain of PointShopAR. In addition, PointShopAR can also be used to design single objects like convertible furniture and outdoor environments like building facades and backyards. These diverse design examples can be found in our supplementary video.

8 LIMITATIONS AND FUTURE WORK

Our system validated using point clouds for rapid prototyping of environmental designs in AR. However, there are limitations and many areas for future work.

The iPad LiDAR sensor and the ARKit camera tracking are optimized for operation in room-sized environments, and so we optimized our scanning and point cloud creation mechanisms for this case. Unfortunately, this means that our system is not very useful for small-scale operations like arranging items on a table or desk, or large-scale operations like designing a building complex. It is likely that improved algorithms, such as combining multiple scans or using machine learning approaches to infer details, could overcome some of the hardware limitations. Editing higher-fidelity representations like NeRF [51] would also be an interesting future direction.

PointShopAR lets users experience their designs in the current environment, and the ability to experience them elsewhere is limited. Approaches explored in DistanciAR [49] could address this. Our system also has all the building blocks needed to support collaborative design. We are interested in seeing how our system can be used across multiple tablets and head-mounted displays.

The current selection method works well in many cases, but it can be difficult to tune the selection, especially in cases where the environment prevents the user from observing the selection from certain angles. It would help to adopt techniques that let the user change the viewpoint without moving [49]. We would also like to use non-bounding-box selection methods for irregular objects and those in crowded environments, such as letting the user draw a curve as in SpaceCast and TraceCast [52].

Object manipulation would be easier if the user could incorporate physical constraints between objects and the device. It is easy to rotate a door around a vertical axis, but modification would be even easier if the user could place the axis at the door's edge. Similarly, placement and movement would be easier if objects could snap to detected planes in the environment. The user interactions would also be more intuitive if the tablet's pose could be linked to different manipulation modes [30], so the user can avoid switching modes using extra buttons.

Our inpainting algorithm fills patches using the average color of the surrounding points, leading to patches that do not integrate well into the scene if the lighting is uneven or the background has multiple objects. A flood-fill algorithm could provide better color

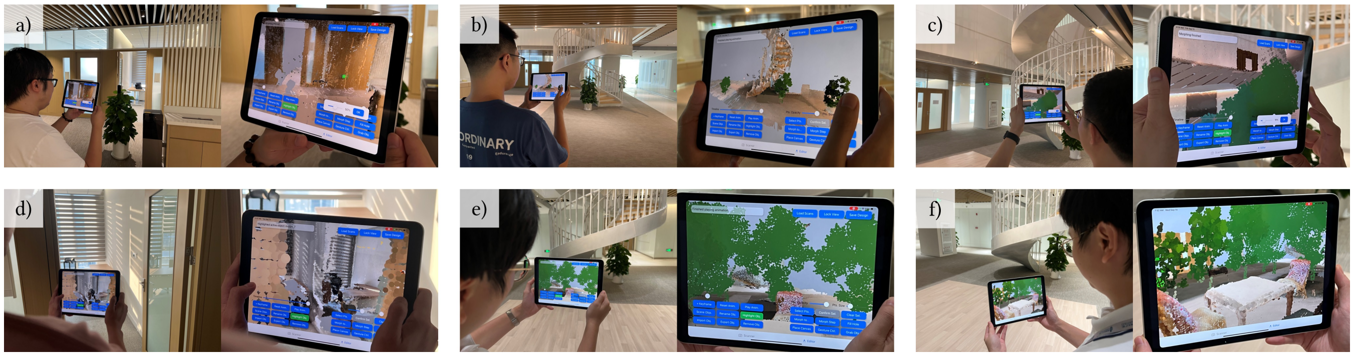

Figure 10: Example designs by participants in the second study. a) Pb1 shortened a long cabinet and removed existing objects so it would be easier to enter an office. b) Pb2 elongated a spiral staircase and rotated it so it takes one circle to go to the next floor. c) Pb8 added a tree and drew a tree house next to a spiral staircase. d) Pb9 broke part of a wall to connect two rooms next door and make a larger office. e) Pb10 moved a spiral staircase to the back and converted an atrium into a place for socializing surrounded by greenery. f) Pb10's design viewed from another angle.

matching. We could also improve the spacing and alignment of points to make patches integrate better with the surroundings.

PointShopAR currently captures a static scene as a point cloud, but the corresponding modules can be readily extended to support recording dynamic scenes to fully explore the potential of the point cloud representation. This would allow freeform animation of spatial elements in combination with other features like morphing. We believe the future incorporation of dynamic point cloud support can enable more creative prototyping for environmental design, such as exploring the relationship between spatial design and moving people. A dynamic point cloud recording can also provide useful information about lighting for the AR renderer to provide a more immersive experience.

9 CONCLUSION

In this paper we introduced PointShopAR, a novel system for prototyping environmental design in AR that integrates point cloud capture and editing on a tablet device. Our user studies helped identify three major contributions.

First, we showed that point clouds are quick and easy to capture, enabling a system that lets users start editing after just a short delay. The turnaround time is sufficiently fast that users can effectively switch back and forth between capturing and editing if their task requires it.

Second, we showed that point clouds support a wide variety of editing operations. The representation is intuitive enough that first-time users were able to make meaningful changes to their environment in about 20 minutes.

Finally, we showed that a tablet-based AR interface is an effective way to edit point clouds and iterate the design. Users could carry out their design on site, move around to view the scene from different angles, and use a combination of touch gestures and device movement to manipulate their physical environment.

We look forward to seeing how future research will be able to expand on our work and take AR point cloud capture and editing in new directions.

ACKNOWLEDGMENTS

This work was supported by a gift from Adobe Research and National Science Foundation award #1942257.

REFERENCES

[1] Adobe. 2022. Illustrator. https://www.adobe.com/products/illustrator.html. Accessed Sep 15, 2022.
[2] Adobe. 2022. Medium. https://www.adobe.com/products/medium.html. Accessed Sep 15, 2022.
[3] Adobe. 2022. Photoshop. https://www.adobe.com/products/photoshop.html. Accessed Sep 15, 2022.
[4] Apple. 2022. ARKit. https://developer.apple.com/documentation/arkit. Accessed Apr 7, 2022.
[5] Apple. 2022. RealityKit. https://developer.apple.com/documentation/realitykit. Accessed Apr 7, 2022.
[6] Apple. 2022. SceneKit. https://developer.apple.com/documentation/scenekit. Accessed Apr 7, 2022.
[7] Rahul Arora, Rubaiat Habib Kazi, Tovi Grossman, George Fitzmaurice, and Karan Singh. 2018. SymbiosisSketch: Combining 2D & 3D Sketching for Designing Detailed 3D Objects in Situ. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (Montreal QC, Canada) *(CHI '18)*. Association for Computing Machinery, New York, NY, USA, 1–15. https://doi.org/10.1145/3173574.3173759
[8] Robert McNeel & Associates. 2022. Rhinoceros 3D. https://www.rhino3d.com/. Accessed Apr 5, 2022.
[9] Autodesk. 2022. AutoCAD Software. https://www.autodesk.com/products/autocad/overview. Accessed Apr 5, 2022.
[10] Ming-Yuan Chuan. 2005. Cubic Tragedy. https://www.youtube.com/watch?v=FOOynE1F4P4. Accessed Apr 5, 2022.
[11] Brian Curless and Marc Levoy. 1996. A Volumetric Method for Building Complex Models from Range Images. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '96)*. Association for Computing Machinery, New York, NY, USA, 303–312. https://doi.org/10.1145/237170.237269
[12] Steve Dent. 2022. IKEA's Latest AR App Can Erase Your Furniture to Showcase Its Own. https://www.engadget.com/ikea-ar-app-lets-you-preview-its-furniture-in-your-own-house-130004284.html. Accessed June 22, 2022.
[13] Julie Dorsey, Songhua Xu, Gabe Smedresman, Holly Rushmeier, and Leonard McMillan. 2007. The Mental Canvas: A Tool for Conceptual Architectural Design and Analysis. In *15th Pacific Conference on Computer Graphics and Applications (PG'07)*. IEEE, 201–210.
[14] Charles M. Eastman. 1990. Vector Versus Raster: A Functional Comparison of Drawing Technologies. *IEEE Computer Graphics and Applications* 10, 5 (1990), 68–80. https://doi.org/10.1109/38.59039
[15] 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.
[16] Andrew Hynes. 2022. scikit-spatial. https://scikit-spatial.readthedocs.io/en/stable/index.html. Accessed Apr 7, 2022.

[17] Ananya Ipsita, Hao Li, Runlin Duan, Yuanzhi Cao, Subramanian Chidambaram, Min Liu, and Karthik Ramani. 2021. VRFromX: From Scanned Reality to Interactive Virtual Experience with Human-in-the-Loop. In *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems*. 1–7.

[18] Yongkwan Kim and Seok-Hyung Bae. 2016. SketchingWithHands: 3D Sketching Handheld Products with First-Person Hand Posture. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology* (Tokyo, Japan) *(UIST '16)*. Association for Computing Machinery, New York, NY, USA, 797–808. https://doi.org/10.1145/2984511.2984567

[19] 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* (Glasgow, Scotland Uk) *(CHI '19)*. Association for Computing Machinery, New York, NY, USA, 1–11. https://doi.org/10.1145/3290605.3300406

[20] Manfred Lau, Greg Saul, Jun Mitani, and Takeo Igarashi. 2010. Modeling-In-Context: User Design of Complementary Objects with a Single Photo. In *Proceedings of the Seventh Sketch-Based Interfaces and Modeling Symposium*. 17–24.

[21] 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.

[22] 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.

[23] Crystian Wendel M. Leão, João Paulo Lima, Veronica Teichrieb, Eduardo S. Albuquerque, and Judith Kelner. 2011. Altered Reality: Augmenting and Diminishing Reality in Real Time. In *2011 IEEE Virtual Reality Conference*. 219–220. https://doi.org/10.1109/VR.2011.5759477

[24] Yuwei Li, Xi Luo, Youyi Zheng, Pengfei Xu, and Hongbo Fu. 2017. SweepCanvas: Sketch-Based 3D Prototyping on an RGB-D Image. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology* (Québec City, QC, Canada) *(UIST '17)*. Association for Computing Machinery, New York, NY, USA, 387–399. https://doi.org/10.1145/3126594.3126611

[25] 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* (Montreal QC, Canada) *(CHI '18)*. Association for Computing Machinery, New York, NY, USA, 1–13. https://doi.org/10.1145/3173574.3173703

[26] Steve Mann. 1994. Mediated Reality. *Technical Report MIT- ML Percom TR-260,* (1994).

[27] Steve Mann and James Fung. 2002. EyeTap Devices for Augmented, Deliberately Diminished, or Otherwise Altered Visual Perception of Rigid Planar Patches of Real-World Scenes. *Presence: Teleoper. Virtual Environ.* 11, 2 (apr 2002), 158–175. https://doi.org/10.1162/1054746021470603

[28] Maxon. 2022. ZBrush. http://pixologic.com/. Accessed Sep 15, 2022.

[29] Paul Milgram and Fumio Kishino. 1994. A Taxonomy of Mixed Reality Visual Displays. *IEICE TRANSACTIONS on Information and Systems* 77, 12 (1994), 1321–1329.

[30] Annette Mossel, Benjamin Venditti, and Hannes Kaufmann. 2013. 3DTouch and HOMER-S: Intuitive Manipulation Techniques for One-Handed Handheld Augmented Reality. In *Proceedings of the Virtual Reality International Conference: Laval Virtual* (Laval, France) *(VRIC '13)*. Association for Computing Machinery, New York, NY, USA, Article 12, 10 pages. https://doi.org/10.1145/2466816.2466829

[31] Leon Müller, Ken Pfeuffer, Jan Gugenheimer, Bastian Pfleging, Sarah Prange, and Florian Alt. 2021. SpatialProto: Exploring Real-World Motion Captures for Rapid Prototyping of Interactive Mixed Reality. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. 1–13.

[32] 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 *(CHI '20)*. Association for Computing Machinery, New York, NY, USA, 1–12. https://doi.org/10.1145/3313831.3376637

[33] Michael Nebeling and Katy Madier. 2019. 360proto: Making Interactive Virtual Reality & Augmented Reality Prototypes from Paper. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. 1–13.

[34] Michael Nebeling, Janet Nebeling, Ao Yu, and Rob Rumble. 2018. ProtoAR: Rapid Physical-Digital Prototyping of Mobile Augmented Reality Applications. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. 1–12.

[35] Julius Nehring-Wirxel, Philip Trettner, and Leif Kobbelt. 2021. Fast Exact Booleans for Iterated CSG using Octree-Embedded BSPs. *Computer-Aided Design* 135 (2021), 103015. https://doi.org/10.1016/j.cad.2021.103015

[36] Richard A. Newcombe, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J. Davison, Pushmeet Kohi, Jamie Shotton, Steve Hodges, and Andrew Fitzgibbon. 2011. KinectFusion: Real-Time Dense Surface Mapping and Tracking. In *2011 10th IEEE International Symposium on Mixed and Augmented Reality*. 127–136. https://doi.org/10.1109/ISMAR.2011.6092378

[37] Open3D. 2022. A Modern Library for 3D Data Processing. http://www.open3d.org/. Accessed Apr 7, 2022.

[38] Patrick Paczkowski, Min H Kim, Yann Morvan, Julie Dorsey, Holly E Rushmeier, and Carol O'Sullivan. 2011. Insitu: Sketching Architectural Designs in Context. *ACM Trans. Graph.* 30, 6 (2011), 182.

[39] Haekyung Park and Dongkun Lee. 2019. Comparison Between Point Cloud and Mesh Models Using Images from an Unmanned Aerial Vehicle. *Measurement* 138 (2019), 461–466.

[40] Polycam. 2022. LiDAR 3D Scanner. https://poly.cam/. Accessed Apr 7, 2022.

[41] 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 *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems* (New Orleans, LA, USA) *(CHI '22)*. Association for Computing Machinery, New York, NY, USA, Article 65, 18 pages. https://doi.org/10.1145/3491102.3517665

[42] Armin Ronacher. 2022. Flask. https://flask.palletsprojects.com/en/2.1.x/. Accessed Apr 7, 2022.

[43] Aditya Sankar and Steve M. Seitz. 2017. Interactive Room Capture on 3D-Aware Mobile Devices. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology* (Québec City, QC, Canada) *(UIST '17)*. Association for Computing Machinery, New York, NY, USA, 415–426. https://doi.org/10.1145/3126594.3126629

[44] Sensopia. 2022. Construction & Floor Plan App for Contractors. https://www.magicplan.app/. Accessed Apr 7, 2022.

[45] Ben Shneiderman. 2007. Creativity Support Tools: Accelerating Discovery and Innovation. *Commun. ACM* 50, 12 (dec 2007), 20–32. https://doi.org/10.1145/1323688.1323689

[46] Ryo Suzuki, Rubaiat Habib Kazi, Li-yi Wei, Stephen DiVerdi, Wilmot Li, and Daniel Leithinger. 2020. RealitySketch: Embedding Responsive Graphics and Visualizations in AR through Dynamic Sketching. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology* (Virtual Event, USA) *(UIST '20)*. Association for Computing Machinery, New York, NY, USA, 166–181. https://doi.org/10.1145/3379337.3415892

[47] Thang Vu, Kookhoi Kim, Tung M. Luu, Xuan Thanh Nguyen, and Chang D. Yoo. 2022. SoftGroup for 3D Instance Segmentation on 3D Point Clouds. In *Proceedings of the IEEE/CVF Computer Vision and Pattern Recognition Conference (CVPR)*.

[48] Michael Wand, Alexander Berner, Martin Bokeloh, Arno Fleck, Mark Hoffmann, Philipp Jenke, Benjamin Maier, Dirk Staneker, and Andreas Schilling. 2007. Interactive Editing of Large Point Clouds. In *PBG@ Eurographics*. 37–45.

[49] 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* (Yokohama, Japan) *(CHI '21)*. Association for Computing Machinery, New York, NY, USA, Article 411, 12 pages. https://doi.org/10.1145/3411764.3445552

[50] Wikipedia. 2022. Environmental Design. https://en.wikipedia.org/wiki/Environmental_design. Accessed Apr 7, 2022.

[51] Bangbang Yang, Yinda Zhang, Yinghao Xu, Yijin Li, Han Zhou, Hujun Bao, Guofeng Zhang, and Zhaopeng Cui. 2021. Learning Object-Compositional Neural Radiance Field for Editable Scene Rendering. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.

[52] Lingyun Yu, Konstantinos Efstathiou, Petra Isenberg, and Tobias Isenberg. 2016. CAST: Effective and Efficient User Interaction for Context-Aware Selection in 3D Particle Clouds. *IEEE Transactions on Visualization and Computer Graphics* 22, 1 (2016), 886–895. https://doi.org/10.1109/TVCG.2015.2467202

[53] 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* (Québec City, QC, Canada) *(UIST '17)*. Association for Computing Machinery, New York, NY, USA, 427–436. https://doi.org/10.1145/3126594.3126601

[54] Qingnan Zhou, Eitan Grinspun, Denis Zorin, and Alec Jacobson. 2016. Mesh Arrangements for Solid Geometry. *ACM Trans. Graph.* 35, 4, Article 39 (Jul 2016), 15 pages. https://doi.org/10.1145/2897824.2925901

A APPENDIX: POINT CLOUD OPERATIONS

This section provides details of various point cloud operations implemented in PointShopAR.

A.1 Generation

After the user finishes the scan, all the recorded data is sent to a web service to generate a point cloud. We first recover the depth maps in meters (range: 15 m) and keep depth values with high confidence. They are used to build a truncated signed distance function (TSDF) volume [11, 36], integrating the RGBD images every 30 frames with corresponding extrinsic and intrinsic camera parameters. The TDSF

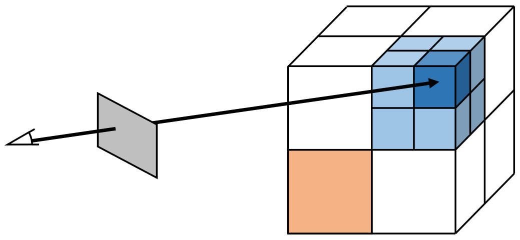

Figure 11: Octree-based point selection. When the user taps on the screen to select a 3D point, the system computes whether the corresponding ray hits any point in the point cloud. The octree makes this significantly faster than a naive sequential approach. Here, the search is able to focus on the light and then dark blue partitions and completely skip nodes in the orange partition.

uses a voxel size of 5/512 meter and 0.05 as the truncation value. We then extract a colored point cloud from the TSDF volume and downsample it with a voxel size of 5/256 meter. The server sends the point cloud to the front-end iPad for rendering and editing.

To interact with points efficiently, we build a three-level octree on the iPad by sorting all points by their XYZ coordinates and computing the size and center of the enclosing bounding box. For each point, we compute a three-digit index of its relative position in an 8×8×8 subdivision of the volume, e.g., 026 to indicate that it is in the 0th eighth of the volume in the x direction, the 2nd eighth in the y direction, and the 6th eighth in the z direction. We first create a fully-populated octree with 256 (= 8^3) leaf nodes, and then traverse all 3D points to update the octree. Each octree node records the number of points in its space partition (all children, grandchildren, etc.) and the xyz octree index (e.g., 026) Each leaf node includes a list of all 3D points in its volume. The octree significantly improves the speed of point-related computations.

A.2 Rendering

Once the point cloud has been processed, we begin rendering it on the iPad. We use Apple's SceneKit for point cloud visualization and editing. For each point, we record its XYZ coordinates, RGB color, and the index of its octree node. A point cloud object is then defined as a list of such points. For each point cloud object, we also maintain its vertex, color, and index buffers to create a `SCNGeometry`. We can visualize the point cloud once the `SCNGeometry` is added to the scene graph. The point cloud is relocalized so the user can view it from the current AR camera in situ. The user can also adjust point opacity and point size using two sliders in the editing interface to make sure they can understand the spatial context represented by the point cloud.

A.3 Point Selection

Fig. 11 shows how we use the octree to enable efficient point selection. Starting at the top level, we project the eight vertices of each octree space partition onto the screen plane and check whether the tapped point falls within the bounding box of the resulting eight points. If not, we ignore all points in that partition. Otherwise we recur down to the next level of the tree. Because the tree often contains large areas with no points, we further optimize this by skipping partitions that had no points assigned to them in the construction. The resulting performance is logarithmic in the number of points. For example, it takes about half a second for a tap to select a 3D point from a point cloud object containing hundreds of thousands of points.

A.4 Inpainting

Our inpainting algorithm has the following steps. First, we use the singular value decomposition (SVD) to find a best-fit plane for each list of control points. We then create one to three rectangular patches that fill the hole but do not extend too far beyond it. For a single-plane hole, we project the control points onto the fitted plane and take their bounding box. For two- and three-plane holes, we find the intersection lines between each pair of planes. Then, for each plane, we project its control points onto its fitted plane and also onto the intersection lines between its plane and the others. We then take the bounding box of these projected points. The result is one rectangle in space, two intersecting rectangles, or three rectangles that intersect at a single point. We sample each rectangle in two dimensions with a spacing of 5/256 meter—the same resolution as our original capture—to generate a grid of points that make a patch. Finally, we assign the average color of the control points on the patch's plane to the patch's points and merge them all to create a point cloud object to send back to the iPad.