SpatialProto: Exploring Real-World Motion Captures for Rapid Prototyping of Interactive Mixed Reality

Leon Müller
LMU Munich
Munich, Germany
l.f.mueller@student.tue.nl

Ken Pfeuffer
Bundeswehr University Munich
Munich, Germany
ken.pfeuffer@unibw.de

Jan Gugenheimer
Télécom Paris - LTCI, Institut Polytechnique de Paris
Paris, France
jan.gugenheimer@telecom-paris.fr

Bastian Pfleging
Eindhoven University of Technology
Eindhoven, Netherlands
LMU Munich
Munich, Germany
b.pfleging@tue.nl

Sarah Prange
Bundeswehr University Munich
Munich, Germany
LMU Munich
Munich, Germany
sarah.prange@unibw.de

Florian Alt
Bundeswehr University Munich
Munich, Germany
florian.alt@unibw.de

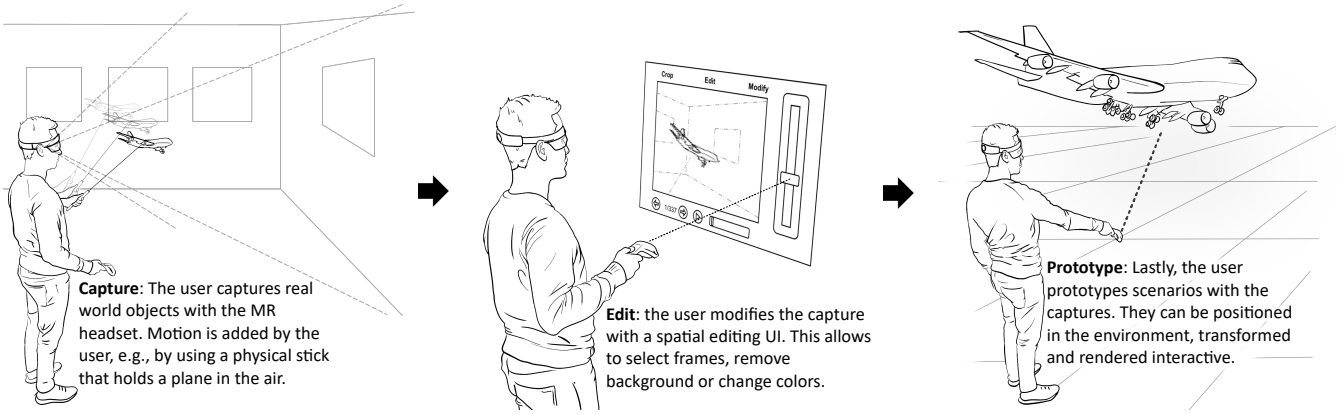

Figure 1: *SpatialProto* enables rapid prototyping of spatial experiences using a depth-sensing capable mixed reality head-mounted display. The headset offers an interface to capture scenes, edit, and prototype operations, whilst an RGB-D camera records the scene the user sees. This affords intuitive recording of real world content, and screening it in the direct environment. Grounding prototyping on how people perceive and know the reality lowers the barrier to creating MR prototypes.

ABSTRACT

Spatial computing devices that blend virtual and real worlds have the potential to soon become ubiquitous. Yet, creating experiences for spatial computing is non-trivial and needs skills in programming and 3D content creation, rendering them inaccessible to a wider group of users. We present *SpatialProto*, an in-situ spatial prototyping system for lowering the barrier to engage in spatial prototyping. With a depth-sensing capable mixed reality headset, *SpatialProto* lets users record animated objects of the real-world environment (e.g. paper, clay, people, or any other prop), extract only the relevant parts, and directly place and transform these recordings in their physical environment. We describe the design and implementation of *SpatialProto*, a user study evaluating the system's prototype with non-expert users ($n = 9$), and demonstrate applications where multiple captures are fused for compelling augmented reality experiences.

CCS CONCEPTS

• **Human-centered computing** → **Systems and tools for interaction design**; *Ubiquitous and mobile computing design and evaluation methods*; Visualization toolkits.

KEYWORDS

Spatial prototyping; mixed reality; interaction; animation

ACM Reference Format:
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 *CHI Conference on Human Factors in Computing Systems (CHI '21), May 8–13, 2021, Yokohama,*

CHI '21, May 8–13, 2021, Yokohama, Japan

ACM ISBN 978-1-4503-8096-6/21/05.
https://doi.org/10.1145/3411764.3445560

Japan. ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/3411764.3445560

1 INTRODUCTION

As Augmented (AR), Virtual (VR), and Mixed Reality (MR) are becoming ever more popular, this comes with a drastic shift in the underlying user interface paradigm. While most prior consumer devices (e.g. tablets, smart phones, monitors) are based on 2D user interfaces, future AR and VR head-worn displays are based on the concept of spatial computing. Instead of arranging all user interface elements on a 2D canvas, spatial computing affords the creation and interaction with 3D models around the user (VR) and/or inside a physical environment (AR).

This novel paradigm creates a set of new challenges for the design and authoring process of spatial environments, in particular when it comes to quickly prototyping animated experiences that blend the virtual and real worlds. One important part of rapid prototyping is to create early versions and mock-ups of a future interactive system quickly and easily. This is often done using paper prototyping, which allows the interaction concept of an idea to be experienced early on with little efforts. However, the complex dynamics of immersive spatial experiences are difficult to prototype, e.g., adding variable 3D artifacts in a scene, their movement and animation, and their interaction with users – tasks usually assigned to domain experts with high technical expertise, but difficult to achieve in early-phase design for non-experts and consumers.

The support of MR early-to-middle stage prototyping is recognised as a significant problem for the wider adoption of the design and development of content. Ashtari et al. recently pointed out that "*[...] devices are becoming easier to access and use, but the barrier to entry for creating AR/VR applications remains high*" [2]. Some tools exist that allow content creators to rapidly prototype AR applications [15, 17, 18] or model animations and dynamic content inside the virtual environment (e.g., Oculus Quill[1]). These are VR-only or provide static artifact design, but there is still a gap toward functionality to rapidly prototype animated and interactive experiences (e.g. future AR or VR games, spatial user interfaces (UIs)) without any programming and/or 3d modeling skills, and without leaving an MR session. Hence, this work is exploring the following two research questions: (1) How can non-expert users be enabled to design animated interactive MR content? (2) What opportunities and challenges are there to create novel MR content with a tool of a novel trade-off between usability and functionality?

We introduce *SpatialProto*, a MR rapid prototyping system focusing on creating and experiencing concepts of animated and interactive MR scenes (e.g., an adventure-style game or spatial UIs). In particular we focus on interactive and animated prototypes which distinguish us from prior work [15, 17, 18]. The system uses a stereoscopic RGB-D camera attached to a VR HMD, enabling a video-see-through MR experience able to display the real world to the user. It can record the physical object inside the field-of-view as a spatial 3D capture from the user's perspective. Regarding the user interface, *SpatialProto* integrates a design pipeline to rapidly prototype spatial computing applications using simple tools (e.g. paper, clay, real-world objects, people) and its functionality. The pipeline consists of three basic steps, all of which can be conducted while being inside MR: 1) Recording, 2) Editing, and 3) Prototyping. The basic interaction concept is illustrated in Figure 1.

To explore how well novice users are able to understand and use *SpatialProto*, we conducted a user evaluation with 9 participants. We found that users without programming expertise were able to design a short animation in less than 20 minutes. Additionally, we present four application scenarios, all using *SpatialProto*, to easily and quickly implement animated spatial computing experiences and let users explore them in MR. Furthermore, we discuss a set of techniques applied during our design process (e.g. stop motion, scaling, chroma keying, interaction chaining) that helped us implement the example applications and informed the design space for MR prototyping systems. We argue that *SpatialProto* is an important step towards making animated and interactive MR prototypes more accessible without requiring any expert skills.

Contribution Statement. In sum, the contribution of our work is threefold: (1) We introduce the concept and implementation of *SpatialProto*, consisting of a MR application and a design pipeline that offers the novel combination of (a) creating animated and lively 3D animations, (b) supports non-expert designers and consumers, and (c) allows immersive prototyping of MR in MR. (2) We report on a user study evaluating the design experience of *SpatialProto*, showing how non-expert users can learn using the tool and creating MR animations within a short amount of time (<20 minutes). (3) We explore spatial prototyping through four applications, created using *SpatialProto* (augmented instructions, spatial task support, interior design, adventure game), demonstrating how combining multiple captures facilitate compelling MR experiences.

2 RELATED WORK

We discuss prior work relating to *prototyping tools*, aimed to provide users support in prototyping/developing MR experiences, and *systems*, which allow novel types of MR experiences mixing real and digital content. The main conceptual properties that make *SpatialProto* novel and distinct to prior work are summarized in Table 1. In the following, we will highlight the distinguishing dimension in-text as *D1-D6*, which also acts as a reference to the corresponding column in the table.

2.1 Prototyping Tools

Expert tools allow to design and implement every little detail towards creating high-fidelity prototypes and products. Frequently used tools for spatial experiences are 3D programming environments such as the Unity 3D or Unreal engine. RoomAlive [11] enables cave-like AR experiences in the user's living room and is build on top of Unity as a toolkit. MARS[2] is a recent extension to Unity that can simulate the real world with proxies for spatially registered content in MR, allowing to more quickly assess and prototype MR scenes. However, the technical barrier in form of required programming skills makes it difficult to quickly assess ideas and concepts, and for non-experts and end-users to utilize these tools. Researchers specifically highlighted the importance to explore tools

[1]Quill: Storytelling in VR: https://quill.fb.com/ (last accessed: 2021/01/12)

[2]Unity MARS: https://unity.com/products/unity-mars (last accessed: 2021/01/12)

Table 1: Comparison of several tools for prototyping of AR scenes across six dimensions. The last row outlines the capabilities of *SpatialProto* that we introduce in this paper.

	***D1* Made for prototyping**	***D2* Immersion of prototyping**	***D3* 3D recording type**	***D4* Fidelity support / Expertise needed**	***D5* Motion support**	***D6* System Complexity**
ProtoAR [18]	✓	Medium (PC +phone AR)	Quasi-3D	Medium (AR + Desktop tool)	2D	Medium (phone + PC)
SceneCTRL [26]		High (HMD)	RGB-D	Low (HMD app)	None	Low (HMD)
RoomAlive [11]	✓	Low (PC)	RGB-D	High (Unity)	3D	High (CAVE)
Unity MARS [24]	✓	Low (PC)	-	High (Unity)	3D	Medium (PC +HMD/Phone)
Remixed Reality [14]		High (HMD)	RGB-D	Medium (HMD app+ room setup)	3D	High (external depth cams)
SpatialProto	✓	High (HMD)	RGB-D	Low (HMD app)	3D POV	Low (HMD)

with novel trade-offs between level of graphical *fidelity supported* and the *expertise needed* (***D4***, Tab. 1) for users [2, 16, 19].

For example, Broy et al. presented two tools, Framebox and Mirrorbox, to prototype a 3D interface targeted at stereoscopic 3D virtual experiences [3]. Another strand of work investigated sketching to make editing 3D easier, e.g. SweepCanvas [12] is an interactive tool for modeling on an RGB-D image. Captured RGB-D images can be viewed on a PC screen and drawing on it allows users to integrate 3D sketches on it. An approach that provides a novel balance between both ends has been introduced by DART [15], sharing with us the goal of providing rapid design explorations of spatial experiences. The toolkit allows users to author content on a 2D GUI application based on a timeline. It offers many features to design with entities, such as behaviors, cues, movements, and 3D actors, and coordinate them on a timeline. Lastly, 360proto from Nebeling and colleagues [17] presents a concept where paper prototyping is exploited to create AR/VR prototypes. Users can capture paper mockups and process them for 360° panorama scenes, and view them in AR/VR capable phones and headsets.

2.1.1 Prototyping MR in MR. Several works pursued the goal of enabling prototyping with the same medium as the final product, to allow to directly experience and evaluate the artifact. As Astari et al. note, "*to constantly transition between a VR headset and the console made it especially difficult to debug and properly test applications*" [2]. There are VR applications that offer to prototype content immersed inside a 3D scene, such as the Unreal Engine VR Mode [7], Oculus Quill, Microsoft Maquette[3], Disney's PoseVR[4]. Nebeling et al's XRDirector is combining multiple mediums such as phones, AR, VR and screens [16, 17] with multiple users, to collaboratively create VR content/prototypes. These tools contribute to the immersion of the prototyping process, but none of them has explored the challenge to integrate the AR/MR landscape, affording real world alignment and capture, and exploit it for rapid prototyping.

Closely related is Nebeling and colleagues's ProtoAR [18], a tool aimed for non-expert users that integrates AR views to prototype AR experiences with a desktop PC. The tool allows to record real play-doh or paper prototypes by using a smartphone, and turn them into "quasi" 3D-stills (***D3***, Table 1) or 2D-animated motion recordings. These captures can be processed and viewed in see-through AR of a phone. Our work builds on their work, and extends this to a fully immersive head-worn MR to explore novel prototyping capabilities of 1) directly recording 3D content and animations for lively and vivid prototypes, 2) supporting the capture of real world objects with variable scale (e.g., recording the whole motion of a moving person), and 3) do all this within a singular immersive experience (***D2***, Table 1) where all stages of prototyping (record, edit, prototype) are performed with a head-worn MR device.

2.1.2 Integrating 3D Captures. Integrating 3D capture through RGB-D images from a depth camera improves depth perception and immersion of spatial experiences, as shown in several works. SceneCtrl [26] is a system to interactively edit MR scenes using scene modification, spatial mapping and custom texturing manipulation in AR, allowing users to cut out a real object and relocate its virtual model in AR, e.g., for moving a chair in context of interior design. However, it is based on the static room capture of the Hololens, which precludes dynamic motion capture. Using projector-based AR, C-Space is an interactive prototyping platform emphasizing the importance of early-phase design explorations for 3D environments [22]. It enables collaborative design on a table that integrates pre-made tangibles with projections for AR. These tools are not made for prototyping per se (***D1***, Table 1). Nonetheless, they demonstrate the utility of integration 3D recordings and animation, a key element of *SpatialProto* (***D3***, Table 1). Explicit prototyping tools such as ProtoAR support still "quasi"-3D captures without using RGB-D images, which *SpatialProto* extends with an RGB-D based perspective-dependent 3D recording.

2.1.3 Integrating Animation to MR Design. The support of 3D content and animations renders prototypes lively and vivid, but a critical problem of current tools is that it is "*Difficult to plan and simulate motion*" [2]. ProtoAR as a prototyping tool support animation, however only for 2D picture-based recordings (***D5***, Table 1). GhostAR [4] is an editor for designing animated robot movement. Users can generate and capture 3D movements directly in AR. The HMD user

[3]Microsoft Maquette: https://www.maquette.ms/ (last accessed: 2021/01/12)
[4]Disney PoseVR: https://www.technology.disneyanimation.com/projects/PoseVR (last accessed: 2021/01/12)

mimicks animations and can then map it to robot motion. In MagicalHands [1], freehand gestures are explored to design animations in VR. A set of gestures was investigated to interact with a variety of design modes. Similarly, 3D puppetry [9] allows creating 3D animations on the screen. What sets our work apart is the ability to integrate any animated capture of the user's real environment and directly feed it back into the reality – all in the same MR session.

2.2 MR Systems

Our work is also inspired by past MR systems.

2.2.1 Stationary Systems. Remixed Reality allows users to copy and drag virtually-tracked physical 3D objects, or play and pause the room's full 3D recording similar to an animation [14]. Their system uses multiple depth cameras attached to the room's walls to enable outside-in full 3D room tracking. Similar combinations of external depth tracking and head mounted devices are exploited by Substitutional Reality [21], Holoportation [20], and KinectFusion systems [10]. The full 3D capture allows users to go around the scene and view objects from multiple perspectives. However, such a system limits the use cases to rooms with expensive hardware and high *system complexity* (***D6***, Table 1). Our focus was on creating a prototyping concept around an affordable hardware prototype so that designers and practitioners can recreate the system in their labs and institutions – an off the shelf VR HMD with a depth camera able to produce points clouds of the environment.

2.2.2 Mobile Systems. For mobile, in-situ usage of MR, the system can be based on one sensor on the HMD to capture depth images. This limits 3D captures to the user's perspective, yet has the advantage to be usable beyond the lab, in broader contexts of mobile interactions. Yang et al.'s Dreamwalker [25] uses such a setup. They use a VR device with two depth cameras to estimate the environment in 3D, and replace it with a virtual experience. This provides a relatively coarse virtual reality, but one that aligns closely to the reality. We employ a similar system to allow users control over recording 3D images and utilize them for spatial prototyping.

2.3 Summary

In sum, we aim to build a prototyping tool like ProtoAR, supporting capture, edit, and replay of scenes. However, we aim for it to be fully immersive as a stand-alone MR application and supporting 3D data through RGB-D captures as non-prototyping tools like SceneCTRL or Remixed Reality do – while keeping the system simple with a single HMD and without prior room scans. This means that the high fidelity of for instance Unity IDE is out of scope, but the overall lower technical and user requirements render MR prototyping more accessible to a wider user group.

3 SPATIALPROTO – A SPATIAL PROTOTYPING SYSTEM

SpatialProto is a system for rapid, spatial prototyping using physical reality captures to prototype MR experiences. It is unique in its combination of the following three aspects of rapid prototyping:

- Enabling animations, i.e. dynamic 3D content with motion. This is essential to prototype compelling MR experiences and creates opportunities for designing interactive and vivid experiences. This is usually a difficult task for non-experts.
- Supporting non-expert designers and consumers, i.e., no programming or complex setups are needed to reach a wide range of potential users.
- Prototype MR within MR, by incorporating the immediate physical 3D environment into the design process through direct capture, segmentation, and manipulation techniques and naturally perceiving object properties such as depth and scale in all design stages.

In the remainder of the paper, we first describe the system and a walkthrough of one example. We then explore the system through a user study and application examples of compelling MR experiences.

4 SYSTEM

SpatialProto implements a three-phase design pipeline: **1) Recording**, to record and store spatial captures of the physical environment as animations, **2) Editing**, to extract the relevant areas and modify the colors of the captures or apply image effects using an editor, and **3) Prototyping**, to translate, rotate and scale captures freely in the user's space and make them interactive through triggers that automatically initiate playback. Figure 4 provides an overview of these stages. To create a spatial prototype, the user goes through the three stages of the pipeline in the following order: Recording, Editing and Prototyping. The user can also switch between the stages via the user interface at any point. As such, all stages are accessible within the same application and session.

All prototyping processes are based on a depth-sensing capable MR HMD without extensive setup or calibration. We used an HTC Vive Pro with an attached Stereolabs ZED Mini. The system was also tested on a Valve Index and a Windows MR Headset with a backpack computer for outdoor use. We use Unity, combined with SteamVR, the ZED SDK, and the OpenCVForUnity plugin. The code is written in C#. In the following, we describe the pipeline in detail.

4.1 Recording

The area recorded by the spatial captures depends on the current field of view of the depth camera, in our case the ZED Mini. Everything the user can currently see is captured during recording. The user has multiple ways of recording spatial captures: 1) a static object can be captured as a single frame, 2) a stop-motion record can be created by capturing each frame successively, and 3) a video allows to record continuously.

A single spatial capture frame encapsulates an RGB image (Figure 3a), a depth image (Figure 3b), a mask image (Figure 3d), and metadata such as a position and rotation relative to the capture origin. Because we are recording multiple snapshots when creating animations via the stop-motion or video approach, they are grouped together in 'snapshot groups'. It contains the snapshots' original origin (position and rotation in world space), the snapshots, name, time of capture, playback parameters, placement, position and scale of the trigger, as well as other chained triggers.

Recording works with up to 30 FPS, but we used mostly 10 FPS due to the high storage needs once written to disk. A single capture takes around 15–20 MB, resulting mainly from the depth information, which is stored in an uncompressed RGBAFloat texture

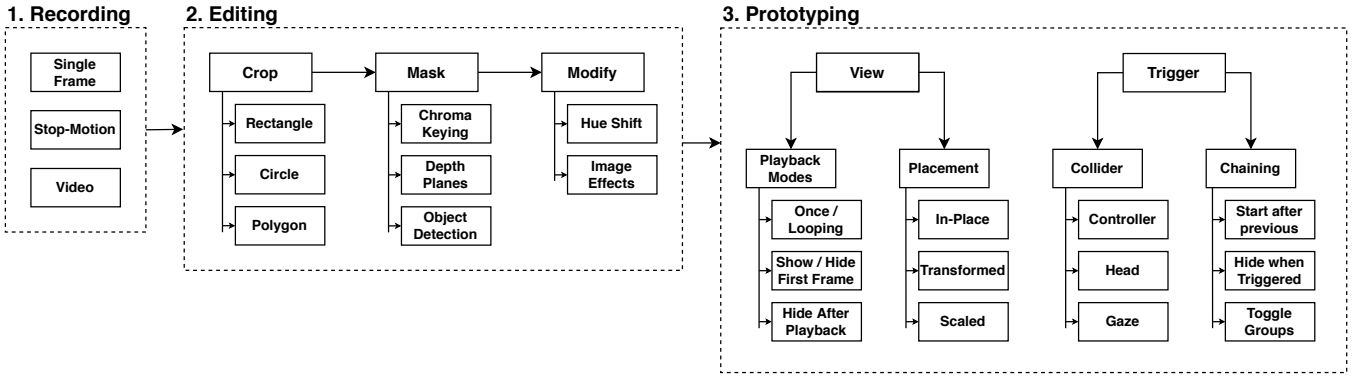

Figure 2: The SpatialProto pipeline has 3 distinct stages: 1. Recording, 2. Editing, and 3. Prototyping. In the recording stage (1), the user can either capture single frames, in a stop-motion approach, or a continuous video. In the editing stage (2), the user has multiple ways of extracting only the relevant area of the capture via cropping and masking. Additionally, the user can also recolor their captures and apply various image effects. Finally, in the prototyping stage (3), the user can define how the captures should be played back, how they should be positioned, and how they should be scaled in the physical world. Users can also create basic interaction with triggers as well as more complex interactions by chaining animations together.

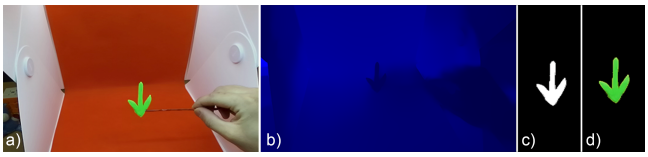

Figure 3: A spatial capture includes an RGB texture to store environmental color (a), a depth texture that provides information on the spatial environment (b) and a mask texture that crops the background (c). The result can be seen in (d).

format as provided and used by the ZED SDK. For example, the arrow animation from the walkthrough takes up around 700 MB.

4.2 Editing

Users can decide which parts of their recordings should be kept or be discarded. We give the user multiple methods of generating masks, to define which part of the spatial capture should be shown and which ones should be discarded. The user can also trim the start and end of the recording, e.g., if they did multiple takes in one recording. Finally, the users can also modify the color of their captures, e.g., to easily create variations, or apply image effects, e.g., a color blindness filter.

4.2.1 Cropping. First, the user can define a rough outline of the area that should be kept, cropping it to only the relevant area. The user can draw a rough outline in the form of a rectangle, circle, or polygon, by using the controller's raypointing and button press. This helps narrowing down the relevant area, making things easier in the next steps.

4.2.2 Chroma Keying. With this method, the user can generate a mask by recording in front of a solid primary color (e.g. green, red, blue as shown in Figure 3a), and subsequently remove this color with a variable threshold from the spatial capture, leaving only areas that are not part of the solid background.

4.2.3 Depth Planes. The user can also generate masks based on the depth information in the spatial capture. If the desired object is clearly separated from the background in the depth dimension, for example, when the user held an object in front of the camera while looking into an vast/empty space, the user can use depth value thresholds to easily extract only that part.

4.2.4 Object Detection. With recent advancements in machine learning, deep neural networks can accurately perform image segmentation, object detection, and mask generation based on pre-trained models and in a short amounts of computing time. With this, the masking process is greatly simplified.

As a first Proof-Of-Concept, we used a version of Mask R-CNN [8] trained on the COCO dataset [13] to generate masks for common objects like furniture, people, animals etc. When generating the mask, the user can select which objects to keep, e.g., people, and masks will be generated only for detected people. To run Mask R-CNN on the RGB images of the snapshots, we use OpenCV's TensorFlow Object Detection API and the OpenCVforUnity plugin. At the time of writing, the OpenCVforUnity plugin did not yet support inference on the GPU, only on the CPU. Because of this, a single frame took about 500 ms to process.

With a rather low threshold, mask generation works well. However, sometimes we experienced that a person was detected successfully in almost all frames, but was missing a single frame in the middle of a recording, thus reducing immersion. Lowering the threshold helped, but could increase the likelihood of false positives. The user can adjust the threshold to their individual captures.

The current implementation is a proof-of-concept. For more specific use cases, Mask R-CNN or similar models could also be trained with specialized images and labels common for prototyping (e.g. paper, cardboard, drawn images, clay, etc.), or a different network together with GPU acceleration could be utilized. Future work should investigate more specific models and parameter choices to increase quality and ease-of-use.

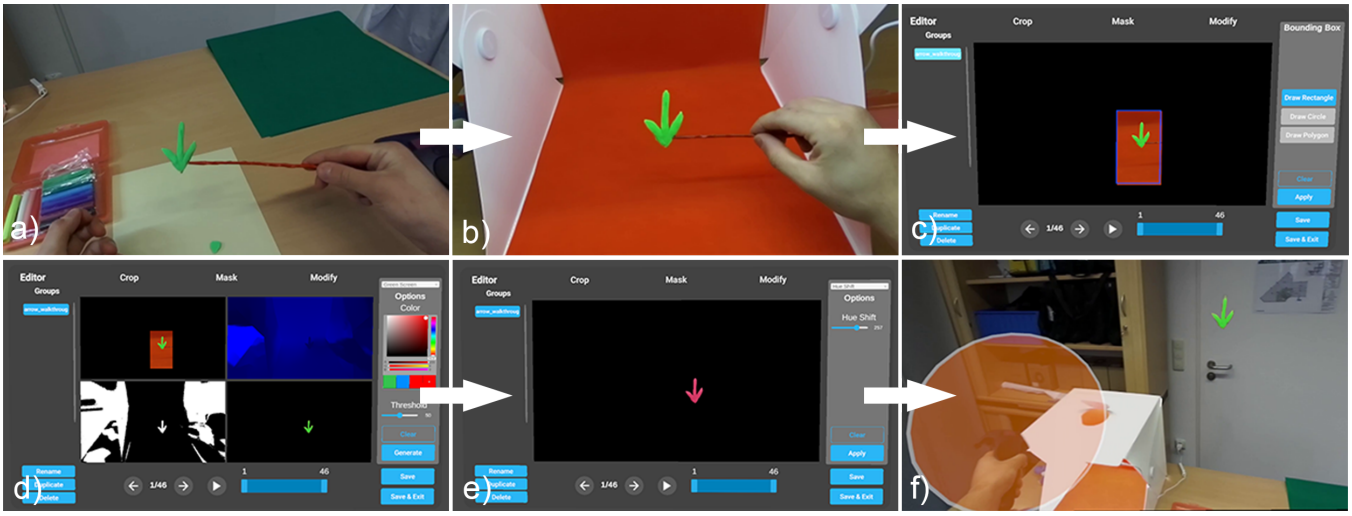

Figure 4: Design Pipeline Walkthrough: (a) The user creates an object using clay, (b) scans it inside the photo-box with an appropriate up-down motion for animation, (c) then uses the editor of SpatialProto to select an area, (d) removes the background, and (e) sets a new color. (f) The last step is to move the object to the desired location in the real world, and also place the trigger sphere to start the animation.

4.2.5 Color and Image Effects. Finally, the user can also modify the source RGB image in two ways. First, a user can shift the hue and the color of a given capture with the Hue Shift option. Hence, the user only needs to build a single version of a specific element and can recolor it in different variants, as seen in Figure 4e. Second, a number of image effects can be applied to the RGB image using shaders. These include a color blindness filter, sharpness and blur filter, and other more artistic filters.

4.3 Prototyping

The final step is the placement, playback mode, and interaction setup in the real world. For playback, the user has different options: Playing once or looping, hiding or showing the first frame, and optionally hiding itself after a single playback. For the placement, the user can either view the recording in its original position or can transform and scale it freely in the real world.

To build interactive experiences, the user can setup triggers in the form of spheres (see Figure 4f) that will trigger the playback of the captures when the head, hands, or gaze (ray) collides with a given trigger sphere. For more complex experiences, the playback of these snapshots can also be chained together.

4.3.1 Rendering. The snapshots are rendered with a shader that takes the RGB, depth and mask texture, a masking color, alpha value as well as a transformation matrix as an input and then renders the masked spatial capture in 3D space in the form of a point cloud. The masking is done by comparing the set masking color (white in our case) with each pixel of the mask in the fragment shader. If the pixel color matches the masking color, the pixel is rendered opaque. If not, that pixel and fragment will get discarded. The transparency (alpha) can also be modified to, e.g., render the snapshot semi-transparent, as is being done in the stop motion approach to help with alignment.

4.3.2 Placement. During the recording of the spatial captures, the world position and rotation of the user and thus the camera are also recorded. With this, spatial captures can be shown in the original position in-place – i.e., the user can experience past captures in place. The user also has the option to freely position, rotate, and scale the capture in the real world. When the placement for a capture is activated, a sphere will appear in front of the capture, and users can grab it with their controller and freely position and rotate it in the real world. As mentioned before, the spacial captures are dependent on perspective. Hence, a second, small sphere with the best viewing angle and rotation is shown for each capture during the placement process. It fades away the closer the user gets to the correct position and rotation. This hint is meant to help users understand the limitations better and place their recordings accordingly. It is hidden once the placement is complete.

4.3.3 Interaction. To make the spatial captures interactive and to better design prototypes that react to input (e.g., pressing a button, entering a room, approaching a window), the user can also place triggers in the form of spheres in the physical space. Triggers will play back a certain capture, if either a hand, the head, or the gaze ray of the user collides with them. For more complex experiences, the user can also define how the spatial captures should be played back and can chain multiple captures together. For playback, the user can choose between playing an animation once or looping it as well as showing or hiding the first frame. In case the animation is played only once, it can also be hidden after playback. Additionally, the user can also chain multiple spatial captures and triggers together to start the playback of a capture once a previous capture has finished playback, hide a collection of captures when a specific capture is triggered, or setup a collection of captures as a toggle group, so that only one can be active at a time.

5 WALKTHROUGH: ANIMATED ARROW

To illustrate the usage of *SpatialProto* we present an example application of an animated arrow floating up and down. This type of visualization could be something an AR designer would implement to explore different variations how to guide the attention of a user towards a certain object (in our case the door knob).

5.1 Building (Figure 4a)

We start by creating a arrow made out of clay and attaching it to a stick. Using a colored stick allows the stick to be removed later in the editing phase.

5.2 Recording (Figure 4b)

We then place the arrow inside a photo-box with a solid red background and extra light. To record, we choose to take a video of our animation. A countdown informs the user to get ready and into position. At the end of the countdown, spatial captures will be taken in very short intervals while we move the arrow up and down. We finish the recording with a button press. To verify that everything was recorded correctly, the recorded captures can be played back in-situ immediately.

5.3 Editing (Figure 4c-e)

In the editor, we first crop to the relevant area by drawing a rectangle around the area where the arrow moves (Figure 4c). The Mask tab allows background removal. As we recorded in front of a solid red background, we choose the chroma-keying option and select the red color to remove the corresponding background (Figure 4d). The optional Modify tab allows shifting the hue (color) of the arrow (see Figure 4e). With this, an arrow can be created in every possible color, without having to record another capture – simply by duplicating the capture and recoloring it. At each step, a live preview is shown to verify and potentially correct the edits. Start/End frames can be modified through a slider.

5.4 Interaction Setup (Figure 4f)

After closing the editor, we open the list of our captures, selecting the new arrow. A large blue sphere will appear in front of the arrow in space, indicating its origin position. We can grab it with a controller to position, rotate, or scale it (with two controllers). To setup a trigger to start the arrow animation, we enable the trigger in the menu and an orange sphere appears at the capture that can be positioned and scaled by the controller. Playback now starts by hitting the orange sphere with either controller.

6 USER STUDY

In this study, we evaluated the design pipeline and process, interaction cycle, prototype implementation, and user experience of SpatialProto. We were interested in how well users with little or no experience in creating MR prototypes could use the system. During the study, we collected primarily qualitative feedback to gain insight to the experience of using the system. The study is meant as early exploration with the goal of (a) showcasing the variety of opportunities our concept creates and (b) demonstrating that its ease-of-use and rapidness make it ideal for low-fi prototyping.

6.1 Tasks and Procedure

After the participants filled out a consent form, they were introduced to the overall study structure that involved three phases.

6.1.1 Learning Phase. In the first phase, we introduced users to the overall idea and concept of the system. A video showcasing the design pipeline and a couple of use-cases was shown. After a brief introduction to VR interaction modalities for users unfamiliar with VR, users could experience a couple of finished animations in MR such as the augmented instructions application example from Section 7.1.

6.1.2 Creation Phase. After the user had a good feeling for the possibilities of the system, we asked the user to create their own animation and to go through the three stages of the design pipeline: *Recording, Editing and Prototyping*. As input (props), we provided a selection of small-size objects such as various toys and models, red and green building blocks, but also more traditional prototyping materials such as modeling clay, paper, tape and pens. We also provided a photo-box with a solid red or green background. To allow the user to better suspend props, we also provided colored strings, wooden sticks, tape and a hot-glue gun.

After the user had selected a prop or multiple props, a plan for capturing the animation was discussed, taking into account the different methods for recording and masking. The time needed to make a single animation was also recorded. After the user had recorded and edited their animations, they were free to position and scale them in the room and could set up the animation trigger.

6.1.3 Feedback Phase. After finishing the tasks, users filled out a usability questionnaire with statements on a 5-point Likert scale about previous experiences with prototyping, familiarity with AR/VR/MR, and questions regarding the experience with the system (Table 2). The study was concluded with a semi-structured interview which went more into detail about the experience, the design pipeline, and usefulness of the system, and also explored what other use-cases and applications users could imagine with this system besides spatial prototyping.

6.2 Participants

We recruited 9 participants. They were between 22 and 42 years old ($M = 26$ years, $SD = 6.08$ years). While five users shared a computer science background, we also had participants from automotive technology, architecture, HCI research, and neuroscience. On a scale from 1(low) to 5 (high), most people had limited experience with VR/MR ($M = 2.37$, $SD = 1.18$) and prototyping ($M = 2.25$, $SD = 1.28$).

6.3 Results

Overall, users found the system "highly interesting" (n=5), "surprisingly easy to use" (n=7), and had fun building an animation. The creation of the first animation including explanations of the UI and capabilities of the system took an average of around 20 minutes.

6.3.1 Prototypes. Example user prototypes are shown in Figure 5. Seven users used stop-motion and chroma-keying for masking, and two users utilized depth-filters and object detection with Mask R-CNN. Four users also used Lego bricks or strings with the same

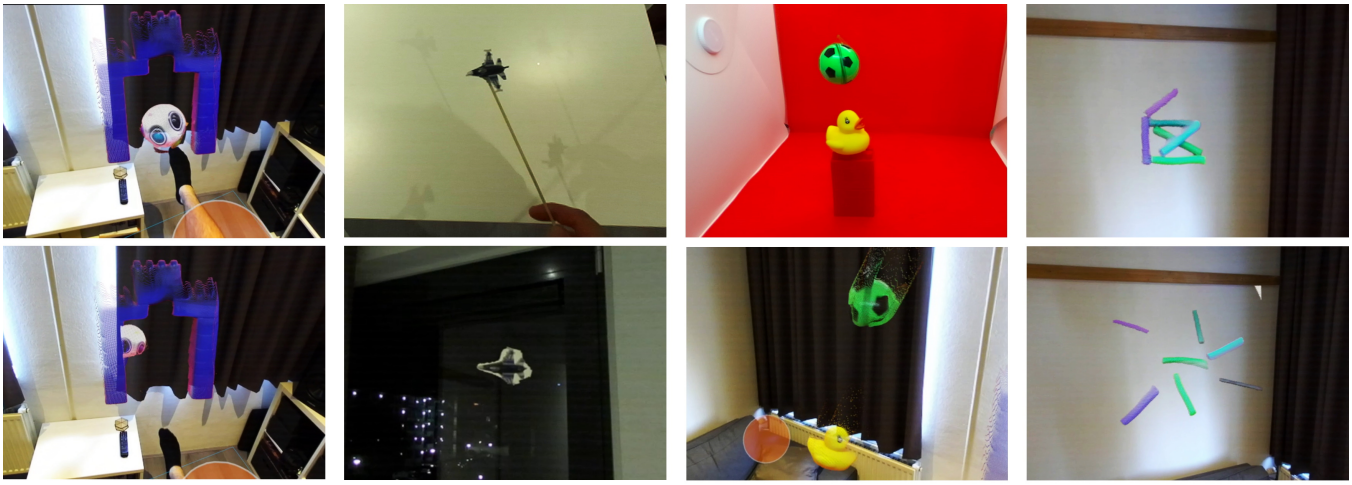

Figure 5: Study Animations: Four examples from the animations created by the study participants. The first animation (a) shows a up-scaled animation of a ball rolling into a goal, giving the impression of kicking a ball into a goal. The second example (b) shows an airplane that was moved on a stick, and placed in the outside world, giving an impression of an airplane flying in the sky. The third example (c) shows an animation of a duck that does a header with a ball. The ball was suspended during recording with a string, and the duck placed on colored bricks which were later removed, to make the duck jump up in the air. The last example shows a house that is slowly built of clay elements and then explodes (d).

Table 2: User feedback on usability questions after the *SpatialProto* sessions with 9 users, rated on a 5-point Likert scale (1 - strongly disagree, 5 - strongly agree).

Statement	M	SD	R
Animating with real props is easier as in software	4.00	0.70	[3,5]
I felt immersed using the tool	4.11	0.92	[2,5]
I found the interaction to be fluent	3.66	1.00	[2,5]
I found the tool easy to use	3.66	1.11	[2,5]
Not having to take off the headset is an advantage	4.22	0.66	[3,5]
For early prototypes, use this over prof. software	4.11	0.92	[2,5]
I would like to use this tool frequently	4.00	0.86	[3,5]
The functions in this tool were well integrated	4.22	0.66	[3,5]
Most people learn to use the tool very quickly	4.44	0.52	[4,5]

color as the background of the photo-box to make props fly or jump. Figure 5a shows a life-size up-scaled animation built out of colorful bricks and a ball that rolls into the goal once the animation is triggered. The animation was recorded in stop-motion and inside a photo-box, and was scaled and placed in such a way that the user could walk up to the goal and kick the ball into the goal with his foot. The next animation, as seen in Figure 5b shows an up-scaled airplane flying through the room, originally recorded as a video by moving a toy airplane mounted to a stick through the room and utilizing Object Detection (ML) to extract only the plane.

In Figure 5c, we can see a rubber duck that is about to jump up, do a header and bounce the ball back into the air. This animation was recorded in stop-motion inside a photo-box, and the ball was suspended and moved by a string. Additionally, to give the impression that the duck jumps up, it was placed on bricks of the same color as the background (red). Other animations include a flying and jumping stuffed animal, a house made of clay building itself up and exploding (see Figure 5d) that could be used as a loading animation, an animated drawing of a car that slowly fills itself with color, a toy air-plane taking off, and an airplane flying around planets.

6.3.2 Learnability. Seven participants mentioned an initial learning curve and need for assistance at the beginning, but once they understood the system and concept, they felt comfortable: "While there's an initial learning curve and some assistance needed at the beginning, it is absolutely worth it once you understand the system" (#7). In the usability questionnaire, all users rated the tool to be fluent, easy to use, and quick to learn for most people (Table 2).

6.3.3 Design Pipeline. Next, we asked participants how they liked the design pipeline and process. The common feedback about the design pipeline was positive, e.g., one user (#6) stated it is "well integrated, easy to follow and use" with similar responses from six other subjects. Seven participants found the concept of using existing props and elements of their reality to be easy and useful and that it provides a "huge advantage, as other ways of prototyping would not have been as easy" (#7). Participants also appreciated not having to take off their headset during the whole process (Table 2).

6.3.4 Comparison with Professional Tools. We asked the participants how the tool compares to professional 3D modeling and animation software. This sparked an interesting discussion, as six users had no experience with 3D modeling software, and three users from automotive design, architecture and industrial design were acquainted with professional 3D modeling and CAD software. Participants with no modeling and programming experience liked that they were able to quickly create actual 3D animations without requiring any 3d modeling or programming skills. Participants with 3D modeling skills mentioned that while SpatialProto would not

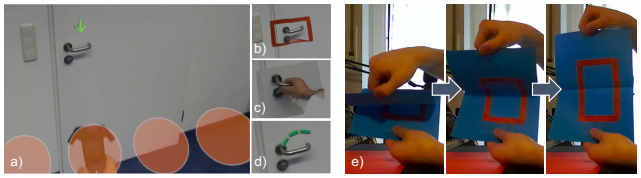

Figure 6: Augmented Instructions: *SpatialProto* can be used to rapidly try out several variations to highlight real world objects, such as a door handle, e.g., (a) by an animated arrow, (b) an animated virtual box, (c) a captured real hand, or (d) a rotating circle. Subfigure (e) shows the creation of the opening box animation created through paper movement for the animated virtual box (b). The orange spheres in subfigure (a) mark the trigger zones for the different animations.

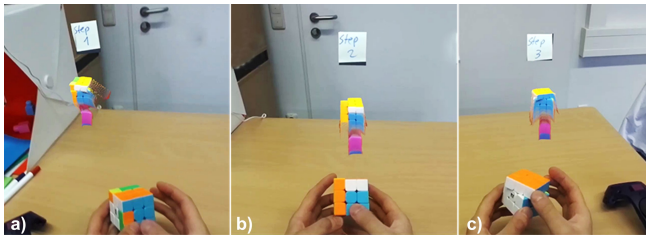

Figure 7: Spatial Task Support: Captures of three different timelines of the same object are shown to aid the user, e.g., to inspect the progress of solving a Rubik's cube.

replace their tools, it could still integrate well into their current design process, and would would allow them to get a feel for their designs and ideas much quicker than with other software, especially in the early stages of a design process where utilizing more professional tools might not be as accessible and more time-consuming. For example, when building small-scale models for architecture or automotive design, participants could seem themselves capturing their designs and scaling them to life-size, to quickly get a better impression and feeling for their design. They also mentioned that the tool could be used to capture their models in multiple stages, and showcase and present the design process. Finally, one participant talked about a complex form-shifting design that would have taken significant resources if built at a real scale, but could have been done much more quickly at a small scale, and by capturing and scaling it up to life-size with SpatialProto, a proper feel and impression of the system could have been evaluated just as well.

6.3.5 Further Use Cases. Other use cases were proposed, e.g., to capture parts of their everyday life or when traveling, such as animals, famous sights and landmarks. Users could then place these captured fragments back in their home, decorating their environment with souvenirs from their travels or exotic animals captured in a zoo. Participants also thought about using SpatialProto as a way to record memories that they could play back at a later time to experience important events of their life again. Another idea was to record processes and step-by-step instructions that could be shared, such as cooking, workouts, DIY tutorials, and furniture disassembly during moving. The latter could also be played in reverse to serve as an instruction how to re-assemble furniture after a move.

7 APPLICATIONS

With our study we were able to understand how good users are able to interact with *SpatialProto*. In this section we will showcase applications that are possible to create using *SpatialProto*. Our main target audience for *SpatialProto* are novice (interaction) designers wanting to explore design iterations for any experience in MR. Since these users have no programming background we grounded our approach in traditional rapid prototyping techniques known by most users with a design background. For a *systematic exploration*, we categorize applications based on four factors. Each combination of instances of a dimension can be interesting for different use cases. We show four unique combinations. Table 3 and Figure 6 to 9 provide an overview. These samples demonstrate the novelty of *SpatialProto*: these applications cannot easily (or only partially) be realized using existing approaches such as ProtoAR or SceneCtrl.

Capture Origin The origin of the captures can be *self-made* by the designer (e.g., through clay), which includes a work process before capture and, for a viewer, shows new content. Alternatively, things *existing* in the world can be recorded. This can include virtual content captured through a screen.

Capture-Time Relation (C-T) A single object can be recorded and played once or in a loop (*1:1*), or the same object can be recorded multiple times where each recording shows a different timeline (*1:N*), e.g., to allow rapidly revisiting different phases of a progress.

Activation Different spatial locations (triggers) can be used to support playing captures (*spatial multiplexing*), but the captures can also be chained to support a flow of interaction over time (*temporal multiplexing*).

Output Captures can be experienced in the *real environment*, or in a *captured environment* if desired, e.g., to assess a transformed environment, a picture-in-picture experience, or to show animations within a specific past timeline.

7.1 Augmented Instructions

We demonstrate how designers can augment real world objects with instructions and support animations, to aid understanding their usage. Figure 6a shows the variants using the basic example of a door knob. In principle it could be generalized to any complex mechanic, requiring instructions. We designed four variants.

- A box that opens around the object (*self-made*). It was created by attaching red tape to a blue piece of paper folded in the middle. During the recording we slowly unfolded the

Table 3: Categorization of applications.

Application	Capture Origin	C-T	Activation	Outp.
Aug. Instruct.	Self-made, Existing	1:1	Spatial	Real
Spatial Tasks	Existing	1:N	Spatial	Real
Interior Design	Self-made, Existing	1:1	Spatial	Capt.
Adventure Game	Self-made, Existing	1:1	Spatial,Temp.	Real

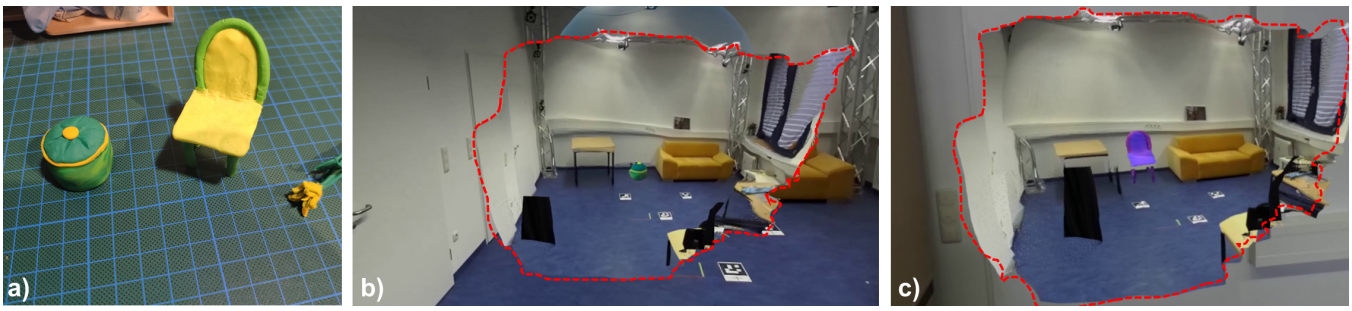

Figure 8: Interior Design: The clay models (a), placed inside a down-scaled version (marked in red) of the same room partially visible in the background of (b). The clay chair is placed in (b), and a different hue-shifted variation is placed in (c).

paper resulting in the red tape to appear and showing a small animation of growing size (Figure 6b).

- An arrow that points to the object (*self-made*). It was created by building an arrow from green modeling clay, attaching it to a wooden stick (masked in red), and then moving and recording it in front of a red background.
- A recording of a real hand using the object in-place (*existing*). It was done by recording the real hand from the user using the object, in this case, moving the hand towards the door handle and pressing it down. A basic mask was generated by drawing a rectangle around the hand and door.
- A circle made of multiple short segments that rotates around the object (*self-made*). It was created in a stop-motion approach, with the segments made from clay laid down on a red paper and moved frame-by-frame by hand. The result shows a rotating motion often seen in loading screens.

The designer can quickly choose between the four variations, as the triggers of the captures are spatially multiplexed. To compare, the user moves the controller into the respective trigger area.

7.2 Spatial Task Support

This example demonstrates a single *spatially multiplexed* capture in the *real environment* – this is useful to offer users the choice to inspect different timelines of the same object (*1:N* Capture-Time Relation). We show this on the example of a manual task, to solve a Rubik's cube, that is supported by *SpatialProto*. However, as it becomes difficult, the user decides to play some of the novel animated instructions in MR. Three steps to solve the Rubik's cube are shown and when the user moves their hand towards the sign of the step, a looping spatial capture is triggered and shows the user the rotations needed to apply to the Rubik's cube to complete it. Animations were recorded in a stop-motion approach inside a photo-box, and thus without need for 3D modeling the cube. In addition, the step-signs above the captures are single-frame captures of posts-its with the different labels written on it.

7.3 Interior Design

This application demonstrates a use case of a *captured environment* that combines *self-made* and *existing* objects. It shows how a scaled environment can be used similar to World in Miniature [23], where users hold a miniature of an environment in their hand and can interact with the inside objects. It allows rapid simulation of new furniture and modifications of them (see Figure 8) to explore interior designs. But instead of moving and placing them in the real room, which requires physical effort, the user uses a scaled down virtual version of the environment. The designer creates small furniture clay models, mimicking the color, rough shape and style they imagine. The user takes a capture of the room to decorate; scales it down to get a better overview, and start placing their creations inside the room to get a feeling for their look and harmony with the other elements and colors in the room. Users can create different color variations using the Hue Shift method, and toggle between them with triggers for each variation.

7.4 Adventure Game

This application prototype shows how multiple input origins can be *spatially* and *temporally multiplexed* for a continuous, longer and spatial MR experience. The use case is an adventure game that progresses linearly and poses three challenges to the player to solve in sequence to finish the game. The *temporal multiplexing* enables this sequence between the three challenges.

The first challenge (Figure 9–top) is to find and open a path to the next level. The path is behind a digitally *real* recorded and placed curtain, blocking the way for the user. The user can open the curtain by placing a Yoshi doll of the right color onto the same colored piece of paper. A *self-made* arrow appears in front of the user pointing at the curtain opening up. This process was implemented by placing an invisible trigger in the correct real world position for the figure, triggering a recording of an arrow and the curtain.

The second challenge begins after the user passes the opened curtain (Figure 9–middle). An opponent appears, trying to approach the user. We recorded a punching motion with an *existing* person using the Mask R-CNN feature. Once the user punches back (detected by a generous trigger area), the opponent falls over (a second *existing* capture) and disappears, completing the second challenge.

The third challenge is to find a way to cross the lava on the floor to the box with a question mark (see Figure 9–bottom). Looking around, the user sees a rotating box with an X' on it. Once the user reaches for the box, the box disappears, and a bridge across the lava appears. Once the lava is crossed, an "You Won" animation is shown, and the box with the question mark can be opened, revealing a coin inside. We implemented this MR game using a combination of

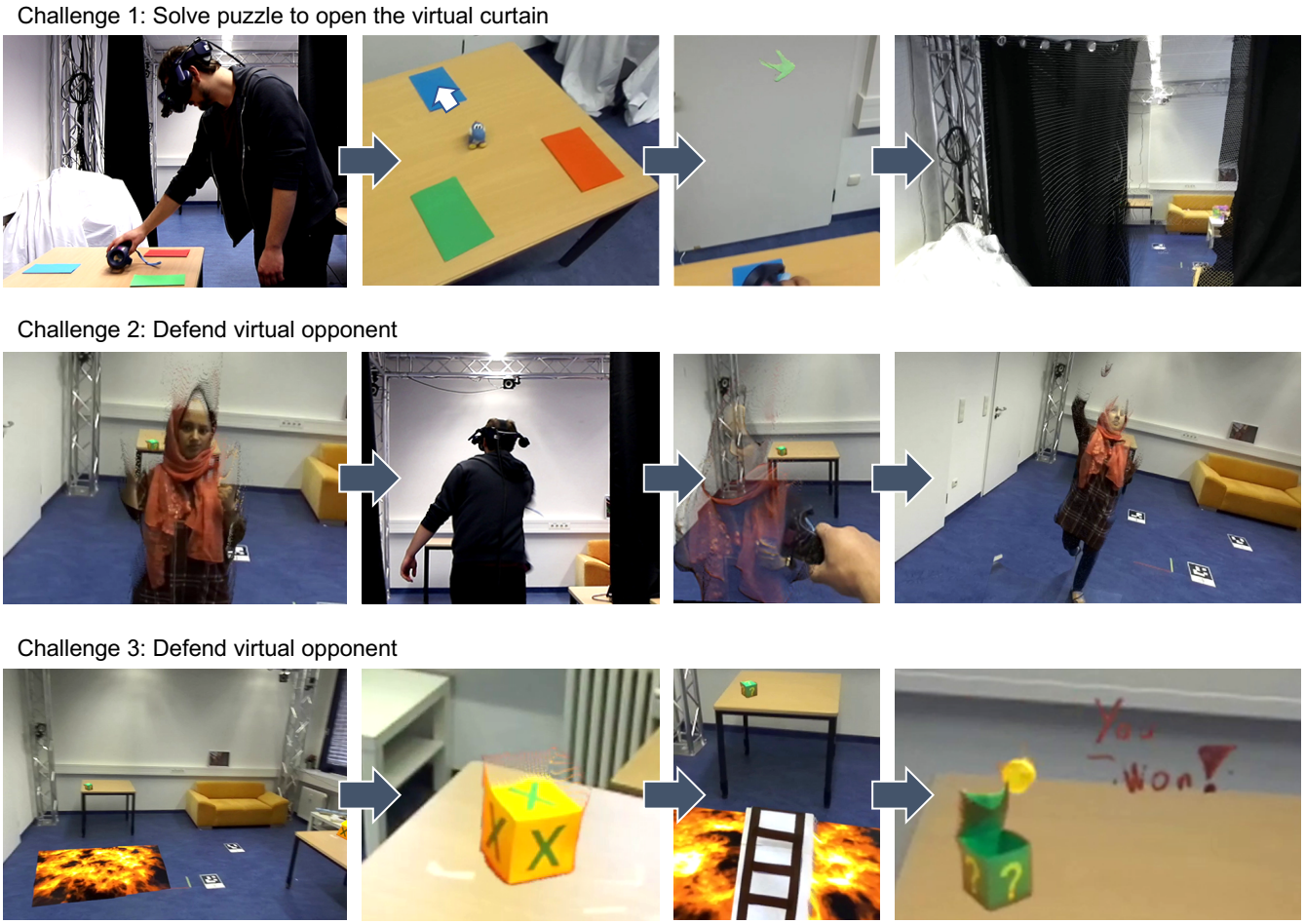

Figure 9: MR Experience Prototype "Adventure Game": This spatial MR experience game comprises three challenges the user needs to solve to reach the goal. 1) Opening a path and virtual curtain by solving a puzzle where the user needs to place the figure onto the right field (top), 2) dodging the move from an animated opponent, overcome by a hit movement (middle), and 3) magically activating a bridge over a patch of lava by reaching for a rotating virtual box, to be able to reach the final destination (bottom). Once the user has reached the goal, a winning text animation is shown, and the user is rewarded with a coin.

the recording techniques (stop-motion, video recording), a clever positioning of triggers and chaining of snapshot groups by *spatial* and *temporal multiplexing*. As the lava animation was not easy to build with clay or paper, and finding real lava was out of scope, we played a lava animation on the smartphone two times. First as an idle animation, and secondly a bridge built out of paper (*self-made*) was slowly placed above the monitor. The animation was then scaled up and placed at the ground.

8 DISCUSSION

In this paper we presented a system specifically for depth-sensing capable MR headsets to exploit the ability to truly blend the virtual with the physical content for spatial prototyping. Recordings can also be accomplished by placing the headset or depth camera anywhere decoupled from the user. As the system is not bound to a lab setup, users can deploy the system at home or in public to record real life situations and activities for integration into MR designs.

8.1 Lessons Learned

8.1.1 Enabling Prototyping for Non-expert Users. Our study showed that non-expert users can understand and use *SpatialProto* to design first MR animations and experiences in a short amount of time. This means that *SpatialProto* satisfies both the need for expressiveness of MR design (creation of animations) as well as a high degree of accessibility because no technical expertise was required. Users envisioned to use the system beyond prototyping, such as capturing everyday events, collecting animated souvenirs and memories, and recording step-by-step instructions and tutorials for cooking, DIY projects, assembly and more.

8.1.2 Rapid Prototyping for 3D Modeling Experts. The study points to the value of the tool for users with 3D modeling or programming

skills. For these users, *SpatialProto* can be complementary to other (high-fidelity) tools. It facilitates rapid prototyping and introduces an opportunity to showcase first design ideas and get a better spatial understanding of design choices.

8.1.3 Expert Use of the System. Our applications provide insight into the upper limits of expert use of SpatialProto. It is interesting for more sophisticated prototypes that include chaining multiple animations in a scene for a sequence of interactions with users, recording the motion of people to create reactive non-player characters, and creating picture-in-picture scenes for design testing. Not possible are fine-grained interactions such as mimicking a virtual keyboard, viewing objects from different perspectives without distortion, and using many animations in parallel.

8.2 Limitations and Future Work

8.2.1 Perspective Dependence and Expressiveness Trade-off. Being able to use off-the-shelf components (a VR headset combined with a depth and pass-through camera), and not having to use special equipment for fully-fledged 3D capture increases the availability of the spatial prototyping capabilities to a broader context of people; not having to leave the MR session increases the efficiency of the prototyping process. However, being free of such constraints comes at a cost: graphical fidelity. Only what is part of the field of view during capture is saved and generates data, meaning an object from the front cannot be viewed from the back, as occluded areas cannot be captured. In future, we could integrate recreating 3D forms from rotation [18], or machine learning models [6] to infer 3D data from single viewpoints. This could be used to approximate missing data, and would allow the spatial captures to be more perspective stable.

8.2.2 Integration of AI-driven Object Segmentation. The current implementation shows a proof-of-concept ML-based implementation to simplify object segmentation. A next step towards a production-ready application would be to systematically investigate models and parameters for ML-based segmentation and to use GPU acceleration for faster inference. Additionally, by integrating approaches to segment moving from static objects as for instance shown around the KinectFusion approach [5, 10], an automatic motion capture tool that complements *SpatialProto* could also be integrated.

8.2.3 System Performance. Using our current prototype, a limited number of captures can be rendered at the same time for an acceptable frame-rate for MR. Our system is capable of rendering around 5-6 captures at the same time at 60 FPS; users building early prototype interfaces and should chain recordings together in a way that unneeded elements are hidden and are only shown when they become relevant. However, in the future this trade-off is likely to shift towards an increased level of detail at decreasing costs. Data reduction methods as also explained for KinectFusion [10], could additional improve the frame rate.

9 CONCLUSION

In this work we presented *SpatialProto*, an in-situ spatial computing prototyping tool implemented using off-the-shelf hardware that allows the user to leverage traditional prototyping tools (e.g. paper, modeling clay, or real-world objects) to build animated and interactive low-fidelity MR prototypes. We further present our design pipeline which allows the user to leverage several recording, editing and prototyping functionalities. In a user-study, we validated that users with little VR/MR/AR experience were able to quickly create 3D animations in MR. Finally, we present a set of example applications, exploring the design space and highlighting the possibilities of expert use. We argue that *SpatialProto* denotes an essential step towards a future where animated and interactive spatial computing prototypes can be designed and implemented by everyone without having 3D modeling or programming knowledge.

Beyond prototyping, *SpatialProto* can be conceived as an app for MR devices for any user to facilitate the ability to record anything you see, replay and reuse it back in actual reality, for example, allowing to capture and playback memories in-place where the original recording was located. This aligns with the vision of beyond-reality capabilities spurred by efforts such as Remixed Reality [14] or Holoportation [20]. *SpatialProto* also affords diminished reality applications, where a previous capture can be superimposed on the live view to realistically diminish objects, albeit posing challenges for correct mapping of light conditions and other environmental cues. These and other potential improvements, such as the object and user detection, are important to increase immersion and seam when experiencing a mix of previously captured and live reality.

ACKNOWLEDGMENTS

This work received funding from DTEC.Bw in the context of the "MuQuaNet" project. We thank Christian Gessner for the support on figure design.

REFERENCES

[1] Rahul Arora, Rubaiat Habib Kazi, Danny M. Kaufman, Wilmot Li, and Karan Singh. 2019. MagicalHands: Mid-Air Hand Gestures for Animating in VR. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology* (New Orleans, LA, USA) *(UIST '19)*. Association for Computing Machinery, New York, NY, USA, 463–477. https://doi.org/10.1145/3332165.3347942

[2] Narges Ashtari, Andrea Bunt, Joanna McGrenere, Michael Nebeling, and Parmit K. Chilana. 2020. Creating Augmented and Virtual Reality Applications: Current Practices, Challenges, and Opportunities. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI '20)*. Association for Computing Machinery, New York, NY, USA, 1–13. https://doi.org/10.1145/3313831.3376722

[3] Nora Broy, Stefan Schneegass, Florian Alt, and Albrecht Schmidt. 2014. FrameBox and MirrorBox: Tools and Guidelines to Support Designers in Prototyping Interfaces for 3D Displays. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Toronto, Ontario, Canada) *(CHI '14)*. Association for Computing Machinery, New York, NY, USA, 2037–2046. https://doi.org/10.1145/2556288.2557183

[4] Yuanzhi Cao, Tianyi Wang, Xun Qian, Pawan S. Rao, Manav Wadhawan, Ke Huo, and Karthik Ramani. 2019. GhostAR: A Time-Space Editor for Embodied Authoring of Human-Robot Collaborative Task with Augmented Reality. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology* (New Orleans, LA, USA) *(UIST '19)*. Association for Computing Machinery, New York, NY, USA, 521–534. https://doi.org/10.1145/3332165.3347902

[5] Jiawen Chen, Dennis Bautembach, and Shahram Izadi. 2013. Scalable Real-Time Volumetric Surface Reconstruction. *ACM Trans. Graph.* 32, 4, Article 113 (July 2013), 16 pages. https://doi.org/10.1145/2461912.2461940

[6] Wenzheng Chen, Jun Gao, Huan Ling, Edward Smith, Jaakko Lehtinen, Alec Jacobson, and Sanja Fidler. 2019. Learning to Predict 3D Objects with an Interpolation-based Differentiable Renderer. In *Advances In Neural Information Processing Systems*. 9609–9619. arXiv:1908.01210 [cs.CV]

[7] Epic Games. 2020. Unreal Engine VR Mode, accessed 5 May, 2020. (2020). https://docs.unrealengine.com/en-US/Engine/Editor/VR/index.html

[8] Kaiming He, Georgia Gkioxari, Piotr Dollar, and Ross Girshick. 2017. Mask R-CNN. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. IEEE Computer Society, 2980–2988. https://doi.org/10.1109/ICCV.2017.322

[9] Robert Held, Ankit Gupta, Brian Curless, and Maneesh Agrawala. 2012. 3D Puppetry: A Kinect-Based Interface for 3D Animation. In *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology* (Cambridge,

Massachusetts, USA) *(UIST '12)*. Association for Computing Machinery, New York, NY, USA, 423–434. https://doi.org/10.1145/2380116.2380170

[10] Shahram Izadi, David Kim, Otmar Hilliges, David Molyneaux, Richard Newcombe, Pushmeet Kohli, Jamie Shotton, Steve Hodges, Dustin Freeman, Andrew Davison, and Andrew Fitzgibbon. 2011. KinectFusion: Real-Time 3D Reconstruction and Interaction Using a Moving Depth Camera. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology* (Santa Barbara, California, USA) *(UIST '11)*. Association for Computing Machinery, New York, NY, USA, 559–568. https://doi.org/10.1145/2047196.2047270

[11] Brett Jones, Rajinder Sodhi, Michael Murdock, Ravish Mehra, Hrvoje Benko, Andrew Wilson, Eyal Ofek, Blair MacIntyre, Nikunj Raghuvanshi, and Lior Shapira. 2014. RoomAlive: Magical Experiences Enabled by Scalable, Adaptive Projector-Camera Units. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology* (Honolulu, Hawaii, USA) *(UIST '14)*. Association for Computing Machinery, New York, NY, USA, 637–644. https://doi.org/10.1145/2642918.2647383

[12] 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

[13] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. 2014. Microsoft COCO: Common Objects in Context. In *Computer Vision – ECCV 2014*, David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars (Eds.). Springer International Publishing, Cham, Germany, 740–755. https://doi.org/10.1007/978-3-319-10602-1_48

[14] 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, Article 129, 13 pages. https://doi.org/10.1145/3173574.3173703

[15] Blair MacIntyre, Maribeth Gandy, Steven Dow, and Jay David Bolter. 2004. DART: A Toolkit for Rapid Design Exploration of Augmented Reality Experiences. In *Proceedings of the 17th Annual ACM Symposium on User Interface Software and Technology* (Santa Fe, NM, USA) *(UIST '04)*. Association for Computing Machinery, New York, NY, USA, 197–206. https://doi.org/10.1145/1029632.1029669

[16] Michael Nebeling, Katy Lewis, Yu-Cheng Chang, Lihan Zhu, Michelle Chung, Piaoyang Wang, and Janet Nebeling. 2020. XRDirector: A Role-Based Collaborative Immersive Authoring System. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI '20)*. Association for Computing Machinery, New York, NY, USA, 1–12. https://doi.org/10.1145/3313831.3376637

[17] 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* (Glasgow, Scotland Uk) *(CHI '19)*. Association for Computing Machinery, New York, NY, USA, Article 596, 13 pages. https://doi.org/10.1145/3290605.3300826

[18] 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* (Montreal QC, Canada) *(CHI '18)*. Association for Computing Machinery, New York, NY, USA, Article 353, 12 pages. https://doi.org/10.1145/3173574.3173927

[19] M. Nebeling and M. Speicher. 2018. The Trouble with Augmented Reality/Virtual Reality Authoring Tools. In *2018 IEEE International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct)*. IEEE Computer Society, 333–337. https://doi.org/10.1109/ISMAR-Adjunct.2018.00098

[20] Sergio Orts-Escolano, Christoph Rhemann, Sean Fanello, Wayne Chang, Adarsh Kowdle, Yury Degtyarev, David Kim, Philip L. Davidson, Sameh Khamis, Mingsong Dou, Vladimir Tankovich, Charles Loop, Qin Cai, Philip A. Chou, Sarah Mennicken, Julien Valentin, Vivek Pradeep, Shenlong Wang, Sing Bing Kang, Pushmeet Kohli, Yuliya Lutchyn, Cem Keskin, and Shahram Izadi. 2016. Holoportation: Virtual 3D Teleportation in Real-Time. 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, 741–754. https://doi.org/10.1145/2984511.2984517

[21] 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* (Seoul, Republic of Korea) *(CHI '15)*. Association for Computing Machinery, New York, NY, USA, 3307–3316. https://doi.org/10.1145/2702123.2702389

[22] Kihoon Son, Hwiwon Chun, Sojin Park, and Kyung Hoon Hyun. 2020. C-Space: An Interactive Prototyping Platform for Collaborative Spatial Design Exploration. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI '20)*. Association for Computing Machinery, New York, NY, USA, 1–13. https://doi.org/10.1145/3313831.3376452

[23] Richard Stoakley, Matthew J. Conway, and Randy Pausch. 1995. Virtual Reality on a WIM: Interactive Worlds in Miniature. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Denver, Colorado, USA) *(CHI '95)*. ACM Press/Addison-Wesley Publishing Co., USA, 265–272. https://doi.org/10.1145/223904.223938

[24] Unity Technologies. last accessed: 12/01/20. *Unity MARS*. https://unity.com/products/unity-mars

[25] Jackie (Junrui) Yang, Christian Holz, Eyal Ofek, and Andrew D. Wilson. 2019. DreamWalker: Substituting Real-World Walking Experiences with a Virtual Reality. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology* (New Orleans, LA, USA) *(UIST '19)*. Association for Computing Machinery, New York, NY, USA, 1093–1107. https://doi.org/10.1145/3332165.3347875

[26] 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