Teachable Reality: Prototyping Tangible Augmented Reality with Everyday Objects by Leveraging Interactive Machine Teaching

Kyzyl Monteiro
Weave Lab, IIIT-Delhi
New Delhi, India
University of Calgary
Calgary, Canada
kyzyl17296@iiitd.ac.in

Ritik Vatsal
Weave Lab, IIIT-Delhi
New Delhi, India
University of Calgary
Calgary, Canada
ritik19321@iiitd.ac.in

Neil Chulpongsatorn
University of Calgary
Calgary, Canada
thobthai.chulpongsat@ucalgary.ca

Aman Parnami
Weave Lab, IIIT-Delhi
New Delhi, India
aman@iiitd.ac.in

Ryo Suzuki
University of Calgary
Calgary, Canada
ryo.suzuki@ucalgary.ca

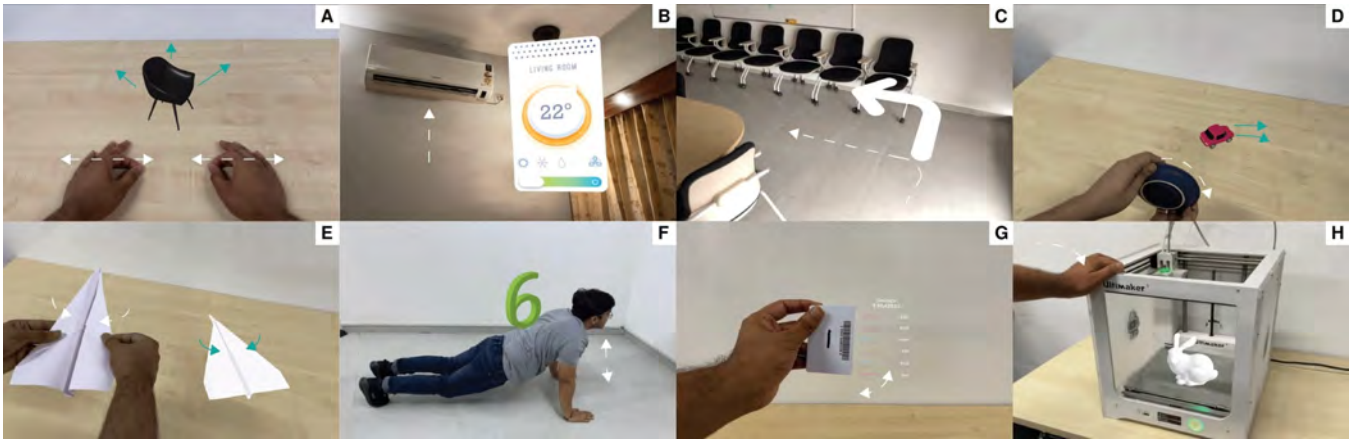

Figure 1: Teachable Reality is an augmented reality prototyping tool to create interactive tangible AR applications that can use arbitrary everyday objects as user inputs. Some prototypes that can be created using Teachable Reality include: (A) An in-situ tangible UI that shows that a pinching gesture can control the scale of the virtual content. (B) A smart home AR application that displays a control panel when you look at the device. (C) A navigation system displays the next arrow showing the direction based on the user's current view. (D) An opportunistic AR controller - using a plate to steer and drive a virtual car. (E) An AR interface that displays 3D origami instructions as a step is completed. (F) An AR Assistant interface that counts the number of push-ups. (G) An intelligent Tangible AR interface that allows the rotation of a card to trigger a different layout. (H) An AR 3D printing interface that enables previewing the print when the user places their hand on the 3D printer.

ABSTRACT

This paper introduces Teachable Reality, an augmented reality (AR) prototyping tool for creating interactive tangible AR applications with arbitrary everyday objects. Teachable Reality leverages vision-based ***interactive machine teaching*** (e.g., Teachable Machine), which captures real-world interactions for AR prototyping. It identifies the user-defined tangible and gestural interactions using an on-demand computer vision model. Based on this, the user can easily create functional AR prototypes without programming, enabled by a trigger-action authoring interface. Therefore, our approach allows the flexibility, customizability, and generalizability of tangible AR applications that can address the limitation of current marker-based approaches. We explore the design space and demonstrate various AR prototypes, which include tangible and deformable interfaces, context-aware assistants, and body-driven AR applications. The results of our user study and expert interviews confirm that our approach can lower the barrier to creating functional AR prototypes while also allowing flexible and general-purpose prototyping experiences.

CHI '23, April 23–28, 2023, Hamburg, Germany

ACM ISBN 978-1-4503-9421-5/23/04…$15.00
https://doi.org/10.1145/3544548.3581449

CCS CONCEPTS

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

KEYWORDS

Augmented Reality; Mixed Reality; Prototyping Tools; Tangible Interactions; Everyday Objects; Interactive Machine Teaching; Human-Centered Machine Learning;

ACM Reference Format:
Kyzyl Monteiro, Ritik Vatsal, Neil Chulpongsatorn, Aman Parnami, and Ryo Suzuki. 2023. Teachable Reality: Prototyping Tangible Augmented Reality with Everyday Objects by Leveraging Interactive Machine Teaching. 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/10.1145/3544548.3581449

1 INTRODUCTION

Today, prototyping AR applications has become easier than ever before, thanks to various commercial prototyping tools (e.g., *A-Frame* [3], *RealityComposer* [38], *Adobe Aero* [36]) and research projects (e.g., *Pronto* [59], *ProtoAR* [65], *360Proto* [64]). However, creating *"functional tangible AR"* applications remains difficult as they need to capture and integrate with ***real-world tangible interactions***. Currently, the common practice for such real-world integration is mainly based on either 1) marker-based tracking [23, 51] or 2) a custom machine-learning pipeline (e.g., *OpenCV* [70], *MediaPipe* [27], etc.). However, marker-based tracking has limited flexibility due to the nature of printed markers (e.g., cannot be used to detect object deformation or body motion [97], the marker always needs to be visible [19], etc.). On the other hand, the custom machine learning (ML) approach allows great flexibility and customizability without the limitation of marker-based tracking (e.g., can incorporate physical motion [83], gesture [89], and body-based interaction [90]), but it requires a significant amount of time and expertise to program such AR experiences.

This paper introduces Teachable Reality, an alternative approach to prototyping tangible AR applications by leveraging ***interactive machine teaching***. Interactive machine teaching [73] is an emerging machine-learning approach that uses user-guided data for a custom classification pipeline (e.g., *Teachable Machine* [11]). By leveraging this, users can easily define their own in-situ tangible and gestural interactions in real-time, which allows the user to prototype functional AR applications without programming. Therefore, our approach enables *quick* and *easy* prototyping similar to marker-based approaches, while allowing *flexible*, *customizable*, and *general-purpose* interactions, similar to the machine learning approach. While interactive machine teaching itself is not new, this paper contributes to the first integration of interactive machine teaching into AR authoring. Based on formative interviews, we design our tool as an end-to-end system that allows the user to detect, train, bind, and author physical-virtual interactions entirely within a mobile AR interface without the need of going back and forth between programming on a desktop screen and testing in the real world. In addition, we explore the design space of our proposed approach. We show the potential of our tool by demonstrating various application scenarios, including tangible and deformable interfaces (Figure 1-D, E), context-aware assistant (Figure 1-B, C), augmented and situated display (Figure 1-G, H), and body-driven AR experiences (Figure 1-A, F).

We evaluate our approach through two user studies: 1) a usability study with 13 participants and 2) expert reviews with six tangible AR experts. The study results confirm that our approach can lower the barrier to creating functional AR prototypes while allowing flexible and general-purpose prototyping experiences. We also found that our approach can complement existing practices, such as marker-based or machine-learning approaches, by allowing rapid iteration toward a high-fidelity prototype. We discuss both benefits and limitations of our approach, pointing out the future opportunity for tangible AR prototyping tools.

Finally, this paper contributes:

(1) A new approach to authoring tangible AR prototypes by combining interactive machine teaching and in-situ AR scene authoring.
(2) A design space of our approach, which covers both input and output of a wide range of real-world tangible and gestural interactions for AR prototyping.
(3) The insights from the two user studies which highlight the benefits and limitations of our proposed approach.

2 RELATED WORK

2.1 AR Prototyping Tools

To better contextualize Teachable Reality within the landscape of the existing AR prototyping tools, we situate ourselves with the following dimensions (underlined category is our focus).

1) Fidelity of Prototypes: *Low-fi vs. Medium-fi vs. High-fi.* Existing AR prototyping tools can be situated in the spectrum between low-fidelity and high-fidelity prototypes. Low-fi prototyping tools like *InVision* [40], *Sketch* [41], and *Adobe XD* [37] allow for quick initial exploration, whereas tools like *A-Frame* [3], *Unity* [33], and *Unreal* [43] are complex but enable high-fidelity interactive AR experiences by providing full-fledged AR development features. Nebeling et al. [66] argue that there is a significant gap between low-fi and high-fi prototyping tools to create interactive AR applications. We aim to fill this gap by providing a medium-fi prototyping tool. This allows the users to create more realistic AR experiences than low-fi prototyping tools, while does not require complex programming like high-fi prototyping tools.

2) Goals of Prototyping: Interaction Design (Interactive) vs. Content Creation (Static) AR prototyping workflow often employs two steps: 1) creating and placing virtual 3D content and 2) defining the interactions between users and virtual content. Many existing tools focus on the first category (e.g., *HoloBuilder* [1], *GravitySketch* [39], *SketchUp* [42], *Lift-Off* [46], *RealityComposer* [38], *Adobe Aero* [36], *SceneCtrl* [96], *Window-Shaping* [34], *DistanciAR* [91]). On the other hand, Teachable Reality focuses on ***interaction design***, similar to systems like *DART* [63] and *ProtoAR* [65], by assuming the user can reuse existing 3D models.

3) Deployment of Prototype: Functional vs. Mock-up When prototyping *interactive* AR experiences, the system needs to detect, track, and understand real-world interactions. Many tools avoid this problem through mock-up prototyping (e.g., video-prototyping

like *Pronto* [59], *Montage* [57] or Wizard-of-Oz prototyping like *ProtoAR* [65], *360proto* [64], *360theater* [81], *WozARd* [5]). In contrast, Teachable Reality, like *Rapido* [58], aims to create a ***functional*** AR prototype, allowing real-world deployment and live user testing in an everyday environment, which is an important need for current AR designers and prototypers [7].

4) Programming Approaches: *Programming by Demonstration vs. Programming by Specification.* Existing approaches to creating functional AR prototypes often rely on simple textual or visual programming. For example, many marker-based AR prototyping tools use block-based or node-based programming, such as *ARCadia* [52], *iaTAR* [56], *ComposAR* [79], *RealityEditor* [33], and *StoryMakeAR* [25]. Alternatively, trigger-action authoring, which is often used with simplified visual programming, allows users to create interactive behaviors by binding a trigger event with a corresponding action, as seen in *ProGesAR* [95], *Situated Game-Level Editing* [67], *MR-CAT* [92], and *Aero* [36]. In either case, most of these tools require the user to *explicitly specify* the desired trigger and action, which can be difficult to work with real-world interactions due to complexity and ambiguity. In contrast, our tool, while leveraging trigger-action authoring, allows the creation of interactive behaviors through ***physical user demonstration***, similar to *Rapido* [58], *CAPturAR* [90], and *GesturAR* [89]. Compared to these tools, however, our tool can support more *flexible* and *open-ended* demonstrations by leveraging interactive machine teaching. For example, while *Rapido* [58] focuses on screen-based interactions, our tool allows the user to demonstrate tangible and physical interactions. This approach not only allows gesture [89] or body-based interaction [90] but also supports a range of user-defined tangible, gestural, and context-driven interactions, such as object deformation, environment detection, and face recognition. While there may be a trade-off in tracking accuracy, our approach can significantly reduce the need for multiple different tools [7, 54] and fill the gap in the fragmented AR prototyping landscape [66].

2.2 Everyday Objects as User Interfaces

Since the birth of tangible and graspable user interfaces [22, 45], HCI researchers have explored ways to use everyday objects and environments as user interfaces. For example, in the context of AR/VR interfaces, researchers use everyday objects as haptic proxies [18, 21], such as *Annexing Reality* [32], *VirtualBricks* [6], and *GripMarks* [98] or blend virtual experiences into surrounding environments [49, 62], such as *WorldKit* [94] and *IllumiRoom* [47]. These prior works augment the tangible paper with fiducial markers (e.g., *Replicate and Reuse* [29], *Paper Trail* [72], *HoloDoc* [60], *Tangible VR Books* [10], *Printed Paper Markers* [97]), or augment surrounding objects and environments with smartphone cameras (e.g., *LightAnchors* [4]), depth-cameras (e.g., *RealFusion* [12], *3D Puppetry* [31]), or embedded invisible tags (e.g., *InfraTag* [19]). Similar to our work, some works also explore in-situ creation of tangible interfaces (e.g., *iCon* [13], *Instant User Interfaces* [16], *Ephemeral Interaction* [88], *Fillables* [17], *Tangible Agile Mapping* [87], *Opportunistic Interfaces for AR* [20]). However, one of the key limitations of these tools is the need for more flexibility and generalizability due to the pre-defined tangible inputs. In contrast, our tool leverages interactive machine teaching, allowing more flexible and customizable user inputs than existing tools.

2.3 Interactive Machine Teaching

Interactive machine teaching is an approach to creating an on-demand machine learning model based on user-guided data [73]. In recent years, systems like *Teachable Machine* [11] have demonstrated the potential by allowing the user to quickly create a classification model through user demonstration. *LookHere* [99] further expands this approach by exploiting users' deictic gestures to create a more accurate model. Since interactive machine teaching is easily accessible for non-technical users, the existing research shows the potential of this approach for tangible storytelling [85], human-robot interaction [93], educational toolkits [14], and programming environments for children [48, 71, 77]. However, to the best of our knowledge, there is no existing work that integrates interactive machine teaching into *AR authoring* or even AR interfaces in general. Since interactive machine teaching itself only supports the creation of the ML model, there is still a significant barrier to incorporating the ML model into AR applications. In contrast, Teachable Reality allows for a no-coding prototyping experience entirely within AR. Thus the user does not need to go back and forth between programming on a desktop and testing in the real world, enabling faster iteration and design exploration, all of which are informed by our formative study.

3 FORMATIVE STUDY AND DESIGN GOALS

To better understand the need for such a system, we conducted formative interviews with six participants (P1-P6) who have experience in prototyping AR applications, tangible UIs, and interactive applications with Teachable Machine. All interviews were recorded and later transcribed with the consent of the participants. During the 30-60 min formative study, we asked about the current practices and challenges of AR prototyping, especially when designing tangible interactions or integrating machine learning for input detection. Two authors conducted a thematic analysis of the transcriptions and identified emerging themes. Another author resolved and compiled them into five themes we describe below.

1) Strong Need of Integrating Real-World Interactions for AR Prototypes. Overall, there is a strong need to integrate tangible objects and interactions for AR applications. Participants shared their previous experiences with tangible AR prototype examples, such as tangible tabletop UI with projection mapping (P4), AR prototypes for sports or exercises (P6), and an AR collaboration tool using physical objects (P3). All participants agreed that blended tangible interaction makes AR applications more unique and interesting.

2) Lack of Flexibility in Marker-based Tracking Techniques. When creating such tangible AR applications, participants often used marker-based tracking (P1, P2, P3, P5). However, they also complained about the limitations of marker-based tracking. For example, the hand occlusion problem diminishes the intended natural interaction (P1, P2). Moreover, participants also point out the lack of flexibility by saying that they need to think of the applications based on what fiducial markers can do rather than what they want (P3). Due to these limitations, the participants sometimes needed

to rely on Arduino and electronic sensors to detect interactions (P1, P3, P6), which could introduce significant overhead (P1). Overall, the participants think that integrating tangible interactions in AR prototypes is "tricky" (P1, P4).

3) Integration of Computer Vision to AR is Not Well-Supported. Participants also used custom computer vision models for gesture detection (P2, P4, P6), but they pointed out that handling raw data to detect a custom gesture was really tedious (P4). Some participants acknowledge that tools like Teachable Machine can lower the barrier, but they also mentioned that integration into AR applications is a challenge (P4, P6). *"P6: I used Teachable Machine, but it is still very time-consuming to integrate it as everything needs to be programmed from scratch"*. In general, participants complained about the lack of available options to integrate real-world tracking into AR. *"P1: I don't believe tools like RealityComposer has any ML support. So if I want to detect custom actions, that's not an option."*

4) Need for Quick Prototyping Without Programming. When asked why they needed integrated tools, they answered that creating a functional prototype is a huge commitment as it can take days to even months (P2, P4, P5). Because of this, most of the participants typically used low-fi prototyping methods such as stop motion (P2), Figma (P1), and Wizard of Oz Powerpoint (P4). *"P1: Within the company, we often use Figma to convey concepts, but it's very hard to actually get a sense of what it's going to feel like."* They agreed that actual functional prototypes allow for a more creative ideation process and easier communication within the team. Moreover, despite their extensive programming experience, they also want to avoid programming as much as possible for quick iteration. Therefore, participants strongly agreed that there is a strong need for an integrated authoring tool without programming.

5) Need for In-Situ Authoring and Live Testing. All participants agreed that the current prototyping workflow using platforms like Unity needs a lot of back and forth. *"P1: It just takes so long to build and push code on AR devices, then sometimes some of the features don't work as expected and so again" "P4: A lot of back and forth between the development devices on the Desktop and the testing devices like mobile phones and the Hololens."* Moreover, they need the virtual assets to be synchronized and directly manipulatable in the AR scene rather than on a separate computer screen (P4, P6). Therefore, it is essential to support in-situ authoring and a live testing environment that leverages both direct manipulation and real-time feedback.

We identified five goals based on the themes that emerged from the formative study analysis, which inform our system design.

(1) ***Real-World Integration:*** should support rich real-world interaction for a blended AR experience.
(2) ***Flexible Tracking:*** should support flexible interaction and tracking for various application scenarios.
(3) ***Integrated AR Authoring:*** should integrate input detection and AR output authoring in the same environment.
(4) ***Direct Manipulation:*** should allow the user to prototype interactive experiences without programming.
(5) ***Live and Real-Time Testing:*** should support the immediate live testing in the real world for quick design iteration.

4 TEACHABLE REALITY

4.1 Overview

Teachable Reality is a mobile AR prototyping tool that combines interactive machine teaching and in-situ AR authoring. Teachable Reality has three key features: 1) *interaction detection:* interaction detection based on an on-demand computer vision classification model, 2) *in-situ AR authoring:* AR authoring environment that lets the user quickly create desired interactive AR behaviors based on user-defined trigger action, and 3) *live deployment and* testing: the user can quickly deploy the AR prototypes for iterative live testing in an everyday environment. Teachable Reality has a simple user interface. The main window shows the live current AR view that provides both authoring and live testing views. The right panel on each screen (Figure 2) presents different options to the user during the two stages of authoring: 1) capture and store the different input states, and 2) save and display each state's corresponding output.

4.2 Authoring Workflow

Step 1: Capturing and Demonstrating the Interaction. The first step is to capture a user's interaction with the tablet's camera. When the user taps the *Add Data* button, the camera starts capturing the scene from the main window so that the user can demonstrate the desired input interaction with an everyday object or environment. The user can add data for multiple states capturing stages of one interaction or multiple interactions according to their need. For example, the user defines four different states based on the position of the black slider handle in a box cutter (Figure 2A). Once the user finishes capturing and taps the *Next* button, the system creates a computer vision classification model based on the provided data and starts automatically detecting each state based on the classifier, similar to *Teachable Machine* [11].

Step 2: Authoring AR Scene for Each Interaction State. After registering each state, the user can author each AR scene with direct manipulation. To do so, the user can tap the + button. Then the user can choose a virtual asset from an asset library (prepared by the user or some default objects) to place into the scene. When tapping the *Save* button located below each state, the user can register the current AR scene as a corresponding AR scene. The placed virtual object can be manipulated with the touch gesture, such as drag-and-drop for position change, pinching gesture to change the scale, and twist gesture to change the orientation. The user can quickly define the interactive behavior by moving the virtual object and saving the scene corresponding to each saved state. The interactive behavior consists of the trigger—the detection of each state and action—storing the corresponding AR scene. For example, Figure 2B illustrates the workflow where the user places a 3D model of a tree on a table, changes the size of the tree with pinching interaction, and then saves it to the corresponding state. When placing the virtual object, the user can also choose different asset types (e.g., 3D object, 2D images, etc) and anchored locations (e.g., surface, object, image, camera, etc), as we discuss in the design space section (Figure 3).

Step 3: Live Testing with Automated Scene Detection. Once the user finishes authoring the AR scene for each state, the prototype is deployed, and the user can start live-testing the prototype.

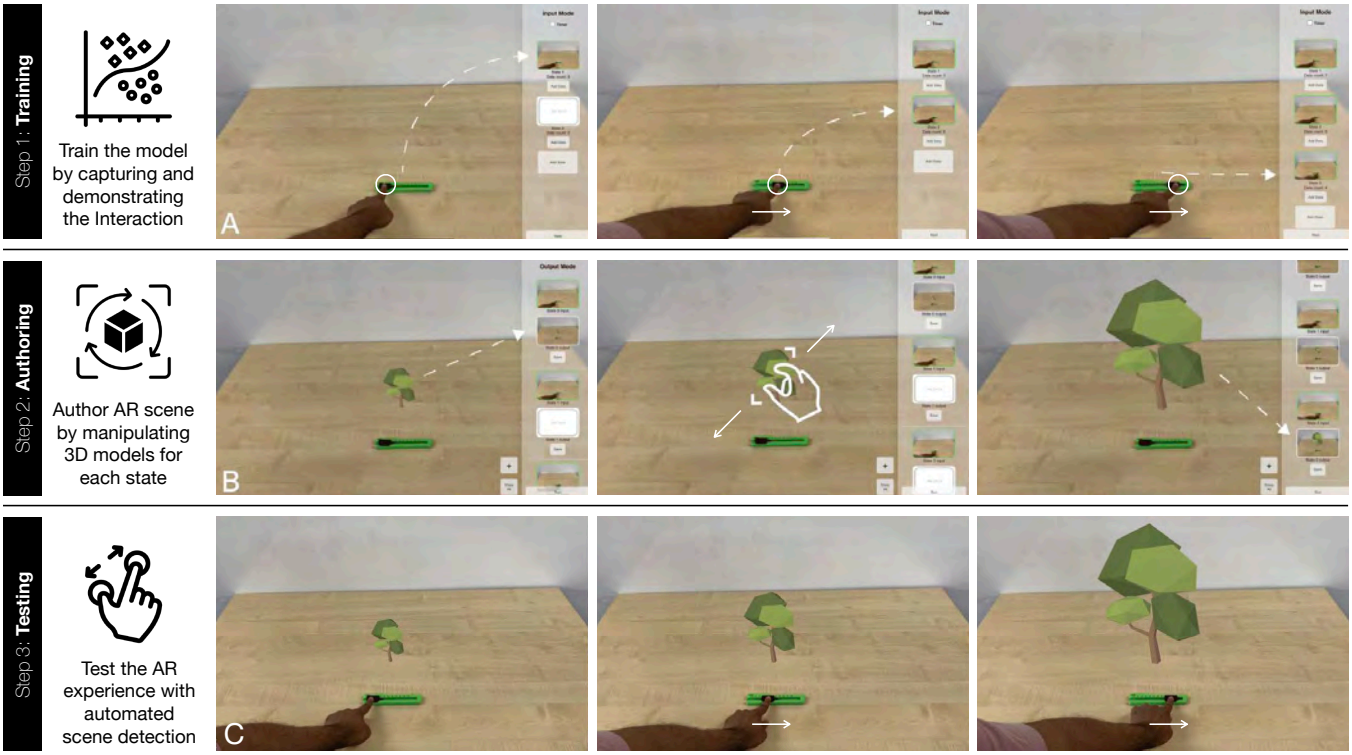

Figure 2: Authoring Workflow of Teachable Reality: Using a paper cutter as a slider to control the scale of a virtual tree: A) User captures three states of the everyday object while demonstrating the tangible interaction. B) User saves the output of the virtual world corresponding to each input state. On importing the virtual asset, the user manipulates the virtual asset according to the desired output and saves it to every corresponding input state. C) User tests the created prototype, which animates between all the outputs that were saved.

In the live preview mode, the system starts automatically detecting the different user-defined states. When transitioning from one state to another, the system automatically animates the virtual object between the corresponding AR scenes, similar to the digital animation technique of auto-tweening. For example, in Figure 2C, the size of the virtual tree changes based on the position of the blade slider of a box cutter, as if the user can use it as a tangible slider. The scale of the virtual tree smoothly due to the automated animation feature, as we described. When transitioning between two stored positions of the slider, the scale of the tree interpolates between the two corresponding scenes the user had demonstrated. For example, in Figure 2C, the size of the virtual tree changes based on the position of the blade slider of a box cutter, as if the user can use it as a tangible slider.

5 IMPLEMENTATION

To democratize tangible AR prototyping experiences, we release our prototype as an open source software [1]. In this section, we describe the implementation detail for each core functionality.

AR Authoring Interface. Teachable Reality is a web-based mobile AR system that runs on any browser that supports the WebXR platform. We tested the system with Google Chrome on Android (Google Pixel 6) and Safari on iOS (iPad Pro 12.9-inch). It is developed using JavaScript, HTML, and CSS and runs entirely on the client side of the browser without needing a web server. The system uses 8th Wall [2], A-Frame [3], and Three.js [84] for the immersive AR authoring system. A-frame enables the placement, manipulation, and animation of virtual assets. While 8th Wall provided us access to spatial understanding, including surface detection and device position tracking.

Image Capturing and Classification. 8th Wall uses the tablet's camera stream to detect the device's position and surface in a real-world environment based on a proprietary SLAM algorithm. We also use the camera stream from 8th Wall for user-defined camera recording as well as the object and human pose detection. For the recorded image classification, we leverage transfer learning using Tensorflow.js, which is the same backend as *Teachable Machine* [11]. The system runs Tensorflow.js on the client side for both the training and inference phase. Since the system needs to train the ML model on-demand, the system trains the model with a separate thread using Web Worker in the background. To reduce the training time, the system leverages the MobileNet model [28] as the base model pretrained on the ImageNet dataset [76]. Training time significantly varies depending on the scenario, thus it is difficult to generalize.

[1] https://github.com/kyzylmonteiro/teachable-reality

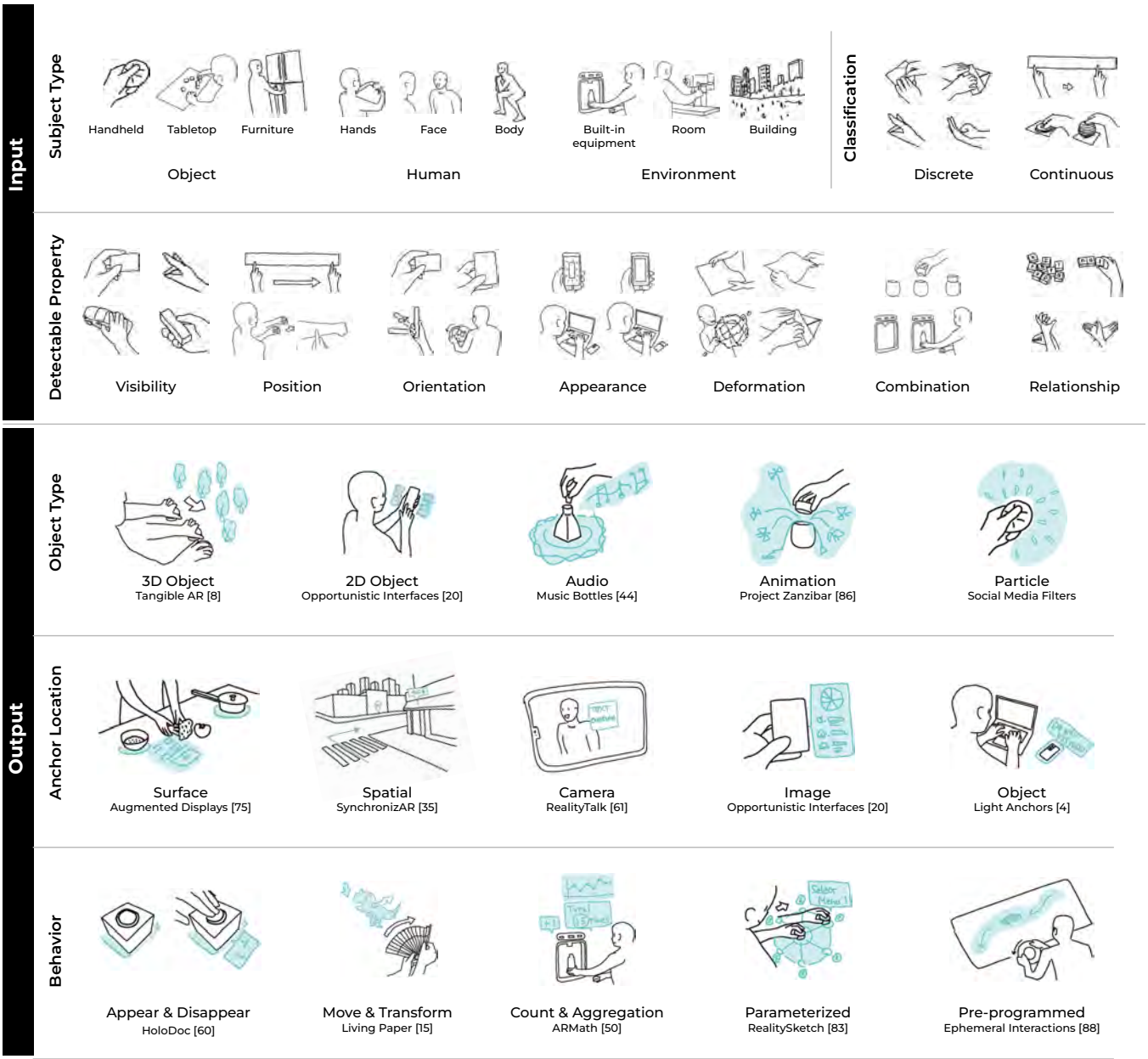

Figure 3: Design space of supported modalities of input and output for tangible AR prototypes.

However, the average training time with 5 states and 100 images for each state took approximately 16.5 sec with 30 trials of the different scenes (min: 8, max: 20.5 sec) with iPad Pro 12.9 inch (M1 chip, 8 Core CPU, 8 Core GPU, and 16GB RAM).

Object Tracking for Location Anchoring. The system also detects and tracks the object for location anchors. For the surface and spatial anchoring, the user uses a detected surface and 3D coordination based on 8th Wall's built-in spatial anchoring features. For image anchoring, the system also uses the 8th Wall's image target feature. For object tracking, the system tracks the object's position based on color tracking. When the user specifies the tracking color by tapping the object on a screen, the system obtains the RGB value of the 2D coordinate, then detects the largest contour of the object with OpenCV.js [70]. Then, the system obtains the 3D position of the tracked color, by raycasting onto the virtual surface. Therefore, the system can only track the object's position on a surface. For human-anchored positions, we use MediaPipe [27] to obtain the human skeleton position data. The system maintains the virtual object location based on the selected anchored origin.

6 DESIGN SPACE

As mentioned, Teachable Reality adopts the *trigger-action* authoring model, which consists of input (trigger) and output (action) for the interactive AR experiences. To better understand what kind of input and output our system can support, we present a design space exploration of Teachable Reality's supported modalities (Figure 3). To explore the design space, we investigated the existing literature on tangible interfaces to identify common elements of input and output for tangible AR applications. To create a generalizable and flexible design space, we first collected examples of tangible AR research, products, and art installations. Then, we abstract the common elements that can be seen in these examples through sketching and categorization. Figure 3 illustrates these abstracted sketches along with a representative example for each category, by providing the name of the project and research paper. While this design space may not be a systematic or exhaustive exploration of all possible tangible AR interfaces, we believe our design space, along with representative examples, provides an overview of what our approach enables and how each element could be used for various applications.

6.1 Input: Types of Subject

At a high level, the system can use any subject as an input as long as it is visible and detectable with the camera. While the design space of detectable subjects is vast, we show three main possible types of subjects (Figure 4).

Figure 4: Input - Types of the subject: The system supports various subjects as inputs, such as objects, humans, and environments.

6.1.1 Object. First, the system can detect a variety of physical objects from handheld- to room-scale objects, such as toys, mugs, papers, books, and furniture, like *HoloDoc* [60] and *3D Puppetry* [31]. The system can detect various tangible interactions with these detected physical objects.

6.1.2 Human. Also, the user can use a human as an input subject, including hand gestures, facial expressions, and body postures, similar to *Interactive Body-Driven Graphics* [78] or *RealityTalk* [61]. Our system itself does not have built-in gesture or posture recognition, but the user can train the model to recognize them in-situ.

6.1.3 Environment. Moreover, while the system itself does not incorporate the device's position or location information, the user could also use a scene and environment as the user input. For example, the user could identify a location or room like a kitchen, bathroom, or living room based on a landmark that is visible with a camera. The user can also quickly create and test an AR navigation experience like *Live View in Google Maps* [26], as seen in Figure 4C.

6.2 Input: Types of Classification

Depending on how the user trains the model, the user can also detect as two different types of inputs.

6.2.1 Discrete. Discrete input means that the detectable states are independent and there is no continuous relationship. By default, Teachable Reality treats all registered states as discrete and independent inputs. For example, the binary state of visibility or different hand gestures are all discrete inputs.

6.2.2 Continuous. But, by registering states in a sequential manner, the user can also define continuous input. For example, the user can use the continuous change of the position, orientation, or deformation of the object as a staggered input parameter. By using this, the user could mimic and treat the input as a numerical and sequential value for a tangible controller (Figure 5).

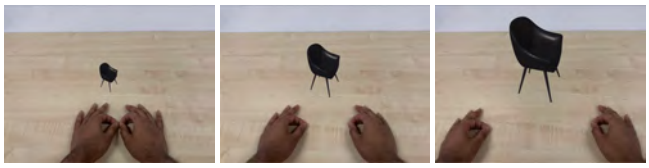

Figure 5: Input - Continuous: Change the size of the virtual chair based on the distance between two hands.

6.3 Input: Types of Detectable Properties

Depending on how the user trains the model, the system can also detect different states of the object for tangible interactions.

6.3.1 Visibility. First, the user can detect the presence or absence of the object based on visibility in the scene. For example, the user can create a binary state to detect whether the user is holding an object in the field of view or not.

6.3.2 Position. Alternatively, the user can detect the different positions of the object in the field of view. For example, the user can use the position of the handle to change the scale of a virtual object just like a tangible slider, similar to *RealitySketch* [83].

6.3.3 Orientation. The user can also use the orientation of the object as an input. For example, the user can quickly create an AR application to show different information about a credit card, such as total balance or detailed monthly expenses, based on the orientation of the card (Figure 6).

Figure 6: Input - Orientation of the object: The user rotates the card to expand on their transaction.

6.3.4 Appearance. The user can also detect the different appearances of the object. Based on the appearance, the user can show different virtual content based on the color of the block, the cover of the book, the application screen of the phone, and the content of a paper.

6.3.5 Deformation. Also, the user can detect deformation of the object, such as bendable paper, expandable Hoberman sphere, origami, and a slinky spring toy (Figure 7). The user can use these deformable objects as input, or alternatively, the user can create an instruction based on the shape of each state (e.g., AR origami instruction).

Figure 7: Input - Deformation of the object: The user can use a deformable object like paper as input to an origami AR instruction animation to show the next step.

6.3.6 Combination. While the above categories are mostly focused on a single object's properties, the user can also detect a combination of objects. For example, by identifying the combination of a hand and an object, the user can detect a simple touch interaction with a physical object.

6.3.7 Relationship. While the combination focuses only on the presence or absence of multiple objects, the user can also use relationships between multiple objects. For example, by detecting the finger's relative position to a paper, the user can mimic multiple touch point detection for the physical paper (Figure 12). Alternatively, the user can also detect the distance between two objects, different grasping gestures for the object, or different arrangements of the multiple objects (Figure 8).

Figure 8: Input - Relationship of the multiple objects: The system detects the different positional relationships between two blocks.

6.4 Output: Types of Virtual Objects

For AR output, the user can place various virtual objects into the AR scene. Here, we describe different types of virtual objects that are supported by our system.

6.4.1 3D Objects. First, the user can place a virtual 3D object in the scene by importing from existing assets (Figure 9). To do so, the user can simply press the *3D Object* button, then the system lets the user select from the available 3D objects. The user can also prepare their own assets to place in the scene. Once placed, the user can change the position, orientation, and scale of the object through touch interaction.

Figure 9: Output - 3D Object: Showing different 3D objects like deer, dog, and elephant, based on the different gestures.

6.4.2 2D Images. The user can also place a 2D image in a scene and create prototypes similar to *Opportunistic Interfaces* [20]. The user first taps the *2D Image* button, then selects the image. 2D image is shown as a texture of the virtual plane in the 3D scene. Therefore, similar to 3D objects, the user can also interactively change the position, orientation, and scale of the 2D objects.

6.4.3 Audio. The system also supports audio output to help create experiences. To do so, the user can simply select an mp3 file, then the system plays the sound when the action is triggered (Figure 10). This way, the user can create a multi-modal output to enrich the AR experience. For example, by detecting the different state of the physical bottle, the user can show a virtual animation and music output when opening the bottle cap, similar to *Music Bottles* [44].

Figure 10: Output - Audio: Playing audio when the lid is lifted.

6.4.4 Other Outputs. The user can also embed various types of pre-programmed assets, such as character animation like in *Project Zanzibar* [86], particle effects, or embedded screens. This enables users to create prototypes of experiences similar to social media filters. Again, the system can load these various types of outputs based on the file import or iFrame. By leveraging the embedded screens, the user can also show other useful outputs like interactive charts or data visualizations.

6.5 Output: Anchored Location

The system also supports different types of anchored locations where the imported virtual object should be placed. When placing a virtual object, the interface lets the user select the anchored location type. By moving the virtual object, the system maintains the position relative to the anchored location.

6.5.1 Surface Anchored. By default, the user can place an object onto a detected surface (Figure 11). Based on the system's built-in surface detection, the user can place a virtual object anchored on a horizontal or vertical surface like a table, floor, or wall like *Augmented Displays* [75].

Figure 11: Output - Surface Anchored: The user can spawn the trees as the position of the token moves.

6.5.2 Spatial Anchored. Similarly, the user can also place a floating virtual object, which stays in a certain spatial position in mid-air. To do so, the user can simply tap the *spatial* option, then the user can start manipulating the object without the bound of the detected surface. Since the mobile AR system can track the spatial position, the spatially anchored object stays in the same position, regardless of the movement of the mobile phone. This allows the users to create prototypes for a system like spatial collaborations like *SynchronizAR* [35].

6.5.3 Camera Anchored. Instead of placing on a spatially-anchored location, the user can also make information always visible by overlaying it in the user's field of view similar to the technique used by *RealityTalk* [61]. When the user taps the *overlay* option, the virtual object is anchored on a screen, so that the user can move the position of the virtual object within the 2D screen.

6.5.4 Image Anchored. The user can also place a virtual object anchored around the image based on the provided target image to create prototypes similar to *Opportunistic Interfaces* [20]. In this case, the virtual object moves along with a paper (Figure 12). To do so, the system leverages a common image target tracking based on the provided image. When the user taps the *image* option, the system lets the user specify the image target based on the selected or uploaded image.

Figure 12: Output - Image Anchored: The user can prototype an augmented display to show related information when reading papers.

6.5.5 Object Anchored. The user can also anchor a virtual object to a physical object or a human (Figure 13). In contrast to image anchored, object anchored can be any physical object or human, which can be useful for object-related information like annotation as seen in *Light Anchors* [4]. When the user taps the *object* option, the user can then tap an object to specify the tracked object. To track an object's position, the system uses simple 2D color tracking and raycasting to obtain the 3D position on a surface, similar to *RealitySketch* [83]. Therefore, the tracking works best with a solid colored object.

Figure 13: Output - Object Anchored: The user can turn off silent mode by flipping the smartphone and hiding notifications.

6.6 Output: Behavior of Virtual Output

As we mentioned, when transitioning from one state to another, the system automatically animates the object. On top of that, the system also supports several additional output behaviors based on the trigger event.

Figure 14: Output - Appear-Disappear: A ball appears when a ball is sketched and Moving & Transformation: The ball moves when the user sketches an arrow.

6.6.1 Appear and Disappear Animation. The most basic output is to appear and disappear a virtual object given the state. By default, the system adds an animation when appearing and disappearing the virtual object by gradually changing its scale. This enables creation of prototypes similar to *TangibleAR* [8], *Holodoc* [60], and *Light Anchors* [4].

6.6.2 Moving and Transforming Animation. Another basic output effect is the movement and transformation of the virtual object. By manipulating the virtual object's position, orientation, and scale for each state, the user can easily create a moving and transformation effect. This can enable the creation of controllers and a tangible user interface, similar to *Living Paper* [15], *Instant UI* [16], *Ephemeral Interactions* [88], and *Bentroller* [80]. By default, the system animates the transition of the virtual object while moving or transforming. For example, the system animates the ball's movement, when transitioning from one location to another (Figure 14).

6.6.3 Counting and Aggregation. The user can also use the detected count for each state. The system automatically counts how many times the specific state is triggered (transitioned from another state) so that the user can also use this value as an output parameter. For example, this can help the user to create a simple counter such as a

count for push-ups or weight-lifting (Figure 15) or an AR prototype of *ARMath* [50]. The user can also integrate this value into HTML to show some aggregated behavior like a graph.

Figure 15: Output - Counting: Counts the number of times a state is achieved

6.6.4 Parameterized Output. Each state is basically discrete from the other, but the user can also define a continuous parameter, as we discussed. For example, the user defines six states based on the position of the tangible object, then the user can use each state as a staggered parameter like [0.0, 0.2, 0.4, 0.6, 0.8, 1.0], given the start (0.0) and end value (1.0). By binding this value to the virtual object's parameter, the user can also create a parameterized output similar to experiences created by *RealitySketch* [83]. For example, the user can associate the parameterized value to the orientation of the virtual object to create a circular slider.

6.6.5 Pre-programmed Control. Finally, the user can also integrate custom scripts for pre-programmed behaviors. For example, the user can change the orientation of the 3D car model based on the detected states of a tangible steering wheel (e.g., left, center, and right, based on the three states) similar to *Ephemeral Interactions* [88], *MarioKart Live* [68], and *Nintendo Labo* [69]. The user can also add a simple script to move the car forward for each time interval as pre-programmed behavior. Then, the user can create a simple virtual radio-controlled car with a tangible steering wheel. The user can also associate custom script behavior to a certain detected state as well (e.g., hiding the steering wheel to stop the car).

Figure 16: Output - Pre-programmed Control: To drive a virtual RC car with a plate as a controller

6.7 Applications

Based on the combination of these numerous modalities, we identify some promising domains and application scenarios.

1) Tangible and Deformable Interfaces: The user can create an in-situ tangible controller with everyday objects similar to *Instant UI* [16], *Ephemeral Interactions* [88], and *RealitySketch* [83], *Music Bottles* [44]. Also users can explore creating deformable user interfaces similar to *FlexPad* [82], *Bendtroller* [80], and *Non-Rigid HCI* [9].

2) Context-Aware Assistant and Instruction: The user can also quickly prototype context-aware assistants, AR tutorials or instructions similar to *Smart Makerspace* [53]

3) Augmented and Situated Displays: The user can replicate situated displays like *HoloDoc* [60], *BISHARE* [100] where the AR scene shows the information related to the object the user is interacting with.

4) Body-Driven AR Experiences: The system can support the quick prototype of body-driven applications, such as exergaming, entertainment, and exercise support. For example, the user can prototype an exercise assistant which identifies correct and incorrect postures of exercises.

7 EVALUATION

We evaluated Teachable Reality in two parts: (1) a usability study and (2) expert interviews. Our study design was based on the "usage evaluation" strategy from Ledo et al.'s HCI toolkit evaluation [55]. Usability studies with end users aim to help verify whether the system is conceptually clear, easy to use, and useful. However, since Teachable Reality is the first to explore rapid prototyping for tangible augmented reality, there is no clear baseline for us to make comparisons with. To address this, we conducted expert interviews to gain insights into current common practices and existing tools. We expect both usability studies and expert interviews will help us identify the benefits and limitations of the current system and gain insights for future iterations.

7.1 Usability Study

7.1.1 Method. We evaluate the usability of our system by asking the user to perform four prototyping tasks. We recruited 13 participants (7 male, 6 female, ages 20-37) from our local community. They all had varying experiences with AR applications on mobile phones, tablets, and head-mounted devices.

7.1.2 Introduction and Set-up. A step-by-step walk-through of the system was given to each participant, using a simple example of spawning virtual objects based on the appearance of a physical object. The participants were given an iPad with a keyboard and a stand to use for the study.

7.1.3 Pre-Defined Tasks. The participants were then asked to perform three pre-defined tasks. For each of the pre-defined tasks, the participants were shown a video demonstrating what had to be created for the task. We chose three simple tasks that enabled the users to explore and use most features of the system. The tasks were as follows:

(1) *Object:* Control a virtual object scale with a physical object.
(2) *Environment:* Decorate the environment with a virtual object.
(3) *Body:* Control the position of a virtual object with body pose.

7.1.4 Open-Ended Task. For the open-ended final task, we asked participants to create prototypes they would like to create using the system. We gave participants inspiration by showing them example prototypes that were created using Teachable Reality. We also provided them with 60 3D objects and 15 physical objects for further inspiration for their creation. We gave minimal assistance to the participants for this task. All tasks were screen recorded to obtain objective measurements (recognition errors, task completion

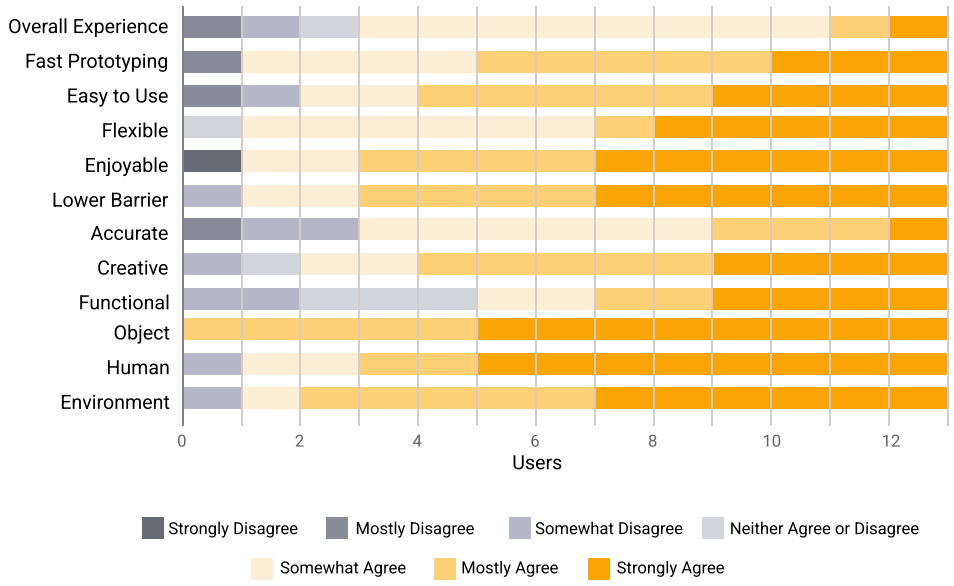

Figure 17: Usability Questionnaire Responses of 13 participants

time, etc). After the four tasks, the participants were asked to fill out a questionnaire evaluating their experience. The study lasted approximately 30-45 minutes per participant, and the participants were compensated with $10 CAD Amazon Gift card.

7.1.5 Results.

1) Overall Experience: Overall, participants responded positively about their prototyping experience. Participants understood how Teachable Reality worked at a high level, found it easy to use, and felt that it was useful. We asked the participants to rate various aspects of Teachable Reality on a 7-point Likert scale followed by some subjective questions. Figure 17 summarizes the 7-point Likert questionnaire response of the usability study.

2) Strengths: Participants found that the workflow was clear (P6, P8), fun (P7), easy to understand (P1, P7, P8), and intuitive (P3, P10). Participants also found the system very enjoyable and rated the system 5.92 on a scale of 1 to 7 for the enjoyability (SD = 1.65). *P1: "I wanted to play with it all day!"*. Participants also stated that they were able to materialize their idea very quickly (P3, P8, M = 5.61, SD = 1.32) for fast prototyping. Participants took 163 seconds (SD = 83s) on average to complete a task (Object-based - 141s, Environment 157s, Human 114s, Open-ended - 238s). Participants agreed the system lowers the barrier (M = 6.07, SD = 1.18) for users from a non-development background. All participants had varying experiences with AR but most participants appreciated Teachable Reality's no coding interface. *P10: "The system is straightforward [and] doesn't use any highly technical language, so it would be a great tool for someone unfamiliar with programming."*. The flexibility of the tool was also recognized by the participants. Especially the capability of creating prototypes with physical objects was seen to be useful (M = 6.61, SD = 0.50). P5: "I definitely think there are lots of opportunities in being able to pick up anything off your desk and being to instrument it." Participants appreciated the authoring of input and output. P7: *"Overall the idea of taking photos of what I want it to respond to was intuitive."* The participants found our approach of integrating interactive machine teaching and AR to enable creativity and that the features add more opportunities for expression (P3). P10: *"It provides lots of opportunities for creativity with minimal effort which I really enjoy."*

3) Areas for Improvements: The participants rated the system's accuracy 4.84 on an average (SD = 1.40), which was a concern among the participants. Recognition errors (like glitches in plane tracking, or delay in state recognition) were seen often (M = 2 times per task), however, these were reported to be minor and did not impair the experience. While most agreed that the interaction of the workflow and use of assets were intuitive, participants gave us suggestions to improve the user experience, which include but are not limited to - the option to delete a virtual asset, an undo-redo button, increasing the size of the buttons, add instructions in the system. Novice participants also pointed out that the tool required them to be creative and that providing some preset animations for the output could be helpful.

7.2 Expert Interview

7.2.1 Method. We also conducted an expert review with six experts to gain in-depth feedback. They all have AR prototyping experiences (Min: 2, Max: 17 years). The experts are faculty in computer science and related areas (E2, E3), PhD or Post-Doc researchers in AR fields (E5, E6), and full-time professional AR prototypers and researchers in large tech companies (E1, E4). We first demonstrated the system with a simple example and showed applications to give a better understanding of the capabilities of the tool. Then, we conducted an in-depth and open-ended discussion about our approach

and use scenarios. The interview lasted approximately one hour for each expert and we provided $20 CAD for their participation.

7.2.2 Insights and Feedback.

1) Overall Impression: All participants were excited by the potential of our tool. Participants found the tool to be quite intuitive (E1, E4), useful (E4, E5), playful (E1, E6), compelling (E2), general-purpose (E1), and versatile (E3). They could see that tool could be easily integrated into their current workflow (E1, E2, E4, E6). All participants agreed that the live testing feature of Teachable Reality could significantly reduce the number of iterations needed during development (E1-E6), by making the prototyping process a lot easier and faster (E1, E2, E4, E6). E2: *"Having this tool available would make my prototyping much faster."* The output authoring interactions like drag-and-drop or pinching were also considered intuitive (E4) and even fun (E1, E6). Besides the general impressions, the participants also commented on specific strengths (listed below) and a few limitations (noted in the next section) of our technical approach.

2) Authoring Workflow and Trigger Action Approach: In terms of the workflow, all participants found the authoring workflow easy to understand. For example, one expert pointed out that the mental model of our workflow is close to the existing creative thinking process (E4), which allows for easy adoption even for non-technical people (E3). E4: *"It's taking a principle of animation and storyboarding. It's basically just turning a storyboard into real life. So it's really easy to understand."*

3) Comparison to Marker-Based Tracking: When compared to marker-based approaches, the participants appreciate the no-setup nature of our approach. Without the need for programming or printing markers, our system significantly reduces the hurdle of prototyping. The participants also appreciated the general-purpose capability, compared to the existing tools (E1). In fact, the participants also found the examples created by the system quite varied (E6), versatile (E3), and impressive (E4), which can broaden the prototyping opportunities that are currently not possible (E2). A participant (E3), who has an extensive experience with the marker-based approach, mentioned that Teachable Reality could be a viable alternative to the current practice.

4) Comparison to Teachable Machine: Since one expert (E3) had used Teachable Machine before, we asked about the difference between the Teachable Machine and our system. E3 pointed out that the no-coding authoring and live testing make for a significant advantage. E3: *"One thing that stands out is the fact that it is deployable immediately. If you use Teachable Machine, it's collecting the data, and then training, training, training, and then checking, which is quite ineffective. One benefit of using your tool is that I can collect the data, manipulate, save virtual content, and deploy the output with a single interface."* In fact, all participants also appreciate Teachable Reality's no-coding approach. E1: *"Even though I know how to program, I like to avoid coding. I always lean towards not coding."* The participants agreed that our tool can lower the barrier for AR allowing designers with visual design backgrounds to create interactive prototypes (E3, E4).

5) Communication Tools as a Potential Use Case: When asked about potential use cases, the experts suggested the strong potential for a *communication tool* (E1, E2, E4). E4: *"I would use it, especially if I'm working in a really collaborative environment if I had an idea, and I wanted to demonstrate it really quickly and have somebody try it out, before I spend a lot of time actually implementing the thing in code."* Currently, they share an idea through a video storyboard, but these storyboards are still just an *approximation* and there is still a communication gap between designers and programmers (E1, E4). Teachable Reality could solve this problem by allowing designers and non-technical people to demonstrate and share their idea quickly between teams. E5 and E6 also mentioned that they would definitely use Teachable Reality for their own current projects if possible.

8 LIMITATIONS AND FUTURE WORK

Addressing Limitations of Computer Vision Based Approaches

While our choice of computer-vision techniques sufficed for novice study participant's prototyping needs and determined suitable by experts for demonstrating a concept (E4), we acknowledge concerns raised generally about its reliability when tested in the wild in uncontrolled environments (E3, E5). In that sense, the participants thought that our tool was not a replacement of existing methods, but rather a complement for early stage yet functional prototypes.

Increase Accuracy of Input Detection. Experts raised concerns regarding the reliability of the accuracy of the system (E3, E5). As they had been using computer vision techniques very frequently they were concerned about how accurate the system was in the wild. On the other hand, they also acknowledged that the accuracy may not be a priority concern as the purpose of this tool is just to demonstrate a concept for oneself or other peers (E4).

We identify the detection accuracy mostly relies on how the user trains the model, and it is difficult to get stable performance. In particular, the inaccurate detection performance for environments or cluttered scenes makes it difficult to test for context-aware applications. To address these problems, future work should leverage user-supported training. For example, *LookHere* [99] uses gestural interactions to specify which region or object the model should focus on. In this way, the model accuracy can greatly improve. Alternatively, instead of an RGB camera, we could also enhance the model training based on the depth camera information like LiDAR input. This allows more accurate tracking and detection.

More Robust Object Tracking. In our current implementation, we use simple object tracking based on selected color matching. We adapt this simple tracking method based on *RealitySketch* [83], which reports that color tracking provides the fastest tracking method for the real-time sketched animation, compared to other sophisticated algorithms *YOLO* [74], *Faster R-CNN* [24], *Mask R-CNN* [30]). However, this tracking method is not robust and generalizable enough for many situations. For example, if the scene has multiple similar colored objects, the tracking may not work well. In the future, we expect the recent advances in computer vision will provide more general, robust, yet fast object tracking methods for our purpose, similar to robust and fast body tracking algorithms (MediaPipe) which we used in our prototype.

Enhancing Usefulness and Usability

Enabling Complex Prototypes. In addition, while the system could create a large number of simple prototypes, experts discussed how future work could look at complex narratives. These would include non-linear narratives, which could have multiple branching of states and more complex logic. E1 suggested a state machine like approach which could look at activating and deactivating states according to the progression of the user's interactions. E5 suggested a block programming approach which would give access to more control over the prototype. We leave these to future work as, though these approaches will be useful, they will be expensive as it would include training multiple models on demand synchronously and activating and deactivating them based on a user's interactions.

Supporting Multi-Modal Input & Output. In this paper, we mostly focus on camera-based input using a computer vision model, but our approach can be generalized to many different inputs. The system can combine multiple inputs for more accurate and expressive interaction. For example, a system that can enable creation of a navigation application that uses the combination of camera and location-based inputs. While our scope for output of prototypes was limited to AR, a potential direction for future work can be to explore other forms of output in addition to AR like haptics. For example, A system that enables creation of a haptic experience which compliments the AR content. This could help create more holistic tangible AR prototypes. With this in mind, future work can showcase a more exhaustive design space and compare the interaction space to present each input and output type's comparison and, as a result, the pros and cons of each interaction type and modality.

Immersive AR Authoring with HMDs. Our current implementation uses screen-based mobile AR, but the integration with head-mounted displays (HMDs) will allow more blended and immersive prototyping experiences like *GesturAR* [89]. In this case, the possible limitation is the lack of computational power, which make the training slow. To avoid this problem, we can leverage cloud-based training for HMDs.

Towards Explainable and Understandable Input Detection. The experts also mentioned that there is a hidden learning curve for how to better train the computer vision model, especially for those without such knowledge. *"E1: You might have to remind that designers don't really have a sense of how these image recognition works. They might be surprised when it doesn't detect a state at a particular angle"*. Experts also pointed out that using ML techniques introduces the "black-box" nature into the input tracking of the system (E2, E3, E4), by saying that if the system trains with some unrelated objects in the background, then it gives incorrect results. They mention that an Explainable AI interface over the tool would be highly beneficial, which we leave to future work.

9 CONCLUSION

This paper presents Teachable Reality, an augmented reality (AR) prototyping tool to create interactive tangible AR applications that can use arbitrary everyday objects as user inputs. When creating functional tangible AR prototypes, current tools often need to rely on markers-based inputs or pre-defined interactions (gesture, location, body posture, electronic devices, etc), which limits the flexibility, customizability, and generalizability of possible interactions. In contrast, Teachable Reality incorporates ***interactive machine teaching*** to immersive AR authoring, which captures the user's demonstrated action as in-situ tangible input, by leveraging on-demand and user-defined compute vision classification. The user can use these classified inputs to create interactive AR applications based on an action-trigger programming model. We showcase various applications examples, which include tangible and deformable interfaces, context-aware assistants, augmented and situated displays and body-driven experiences. The results of our user study confirm the flexibility of our approach to quickly and easily create various tangible AR prototypes.

ACKNOWLEDGMENTS

This research was funded in part by the Natural Sciences and Engineering Research Council of Canada (NSERC RGPIN-2021-02857) and Mitacs Globalink Research Internship. We also thank all of the experts and participants for our user studies.

REFERENCES

[1] 2022. HoloBuilder. https://www.holobuilder.com/
[2] 8th Wall. 2022. Niantic Inc. https://www.8thwall.com/
[3] A-Frame. 2022. A-Frame. https://aframe.io/
[4] Karan Ahuja, Sujeath Pareddy, Robert Xiao, Mayank Goel, and Chris Harrison. 2019. Lightanchors: Appropriating point lights for spatially-anchored augmented reality interfaces. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology*. 189–196.
[5] Günter Alce, Mattias Wallergård, and Klas Hermodsson. 2015. WozARd: a wizard of Oz method for wearable augmented reality interaction—a pilot study. *Advances in human-computer interaction* 2015 (2015).
[6] Jatin Arora, Aryan Saini, Nirmita Mehra, Varnit Jain, Shwetank Shrey, and Aman Parnami. 2019. Virtualbricks: Exploring a scalable, modular toolkit for enabling physical manipulation in vr. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. 1–12.
[7] 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*. 1–13.
[8] Mark Billinghurst, Hirokazu Kato, Ivan Poupyrev, et al. 2008. Tangible augmented reality. *Acm siggraph asia* 7, 2 (2008), 1–10.
[9] Alberto Boem and Giovanni Maria Troiano. 2019. Non-rigid HCI: A review of deformable interfaces and input. In *Proceedings of the 2019 on Designing Interactive Systems Conference*. 885–906.
[10] Jorge CS Cardoso and Jorge M Ribeiro. 2021. Tangible VR book: exploring the design space of marker-based tangible interfaces for virtual reality. *Applied Sciences* 11, 4 (2021), 1367.
[11] Michelle Carney, Barron Webster, Irene Alvarado, Kyle Phillips, Noura Howell, Jordan Griffith, Jonas Jongejan, Amit Pitaru, and Alexander Chen. 2020. Teachable machine: Approachable Web-based tool for exploring machine learning classification. In *Extended abstracts of the 2020 CHI conference on human factors in computing systems*. 1–8.
[12] Vinayak Cecil Piya. 2016. RealFusion: An interactive workflow for repurposing real-world objects towards early-stage creative ideation. In *Graphics interface*.
[13] Kai-Yin Cheng, Rong-Hao Liang, Bing-Yu Chen, Rung-Huei Laing, and Sy-Yen Kuo. 2010. iCon: utilizing everyday objects as additional, auxiliary and instant tabletop controllers. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. 1155–1164.
[14] Jinsung Cho, Geunmo Kim, Hyunmin Go, Sungmin Kim, Jisub Kim, and Bongjae Kim. 2021. DeepBlock: Web-based Deep Learning Education Platform. *The Journal of the Institute of Internet, Broadcasting and Communication* 21, 3 (2021), 43–50.
[15] Stephanie Claudino Daffara, Anna Brewer, Balasaravanan Thoravi Kumaravel, and Bjoern Hartmann. 2020. Living Paper: Authoring AR Narratives Across Digital and Tangible Media. In *Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems*. 1–10.
[16] Christian Corsten, Ignacio Avellino, Max Möllers, and Jan Borchers. 2013. Instant user interfaces: repurposing everyday objects as input devices. In *Proceedings of the 2013 ACM international conference on Interactive tabletops and surfaces*. 71–80.

[17] Christian Corsten, Chat Wacharamanotham, and Jan Borchers. 2013. Fillables: everyday vessels as tangible controllers with adjustable haptics. In *CHI'13 Extended Abstracts on Human Factors in Computing Systems*. 2129–2138.

[18] Florian Daiber, Donald Degraen, André Zenner, Tanja Döring, Frank Steinicke, Oscar Javier Ariza Nunez, and Adalberto L Simeone. 2021. Everyday Proxy Objects for Virtual Reality. In *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems*. 1–6.

[19] Mustafa Doga Dogan, Ahmad Taka, Michael Lu, Yunyi Zhu, Akshat Kumar, Aakar Gupta, and Stefanie Mueller. 2022. InfraredTags: Embedding Invisible AR Markers and Barcodes Using Low-Cost, Infrared-Based 3D Printing and Imaging Tools. In *CHI Conference on Human Factors in Computing Systems*. 1–12.

[20] Ruofei Du, Alex Olwal, Mathieu Le Goc, Shengzhi Wu, Danhang Tang, Yinda Zhang, Jun Zhang, David Joseph Tan, Federico Tombari, and David Kim. 2022. Opportunistic Interfaces for Augmented Reality: Transforming Everyday Objects into Tangible 6DoF Interfaces Using Ad hoc UI. In *CHI Conference on Human Factors in Computing Systems Extended Abstracts*. 1–4.

[21] David Englmeier, Julia Dörner, Andreas Butz, and Tobias Höllerer. 2020. A tangible spherical proxy for object manipulation in augmented reality. In *2020 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*. IEEE, 221–229.

[22] George W Fitzmaurice, Hiroshi Ishii, and William AS Buxton. 1995. Bricks: laying the foundations for graspable user interfaces. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. 442–449.

[23] Sergio Garrido-Jurado, Rafael Muñoz-Salinas, Francisco José Madrid-Cuevas, and Manuel Jesús Marín-Jiménez. 2014. Automatic generation and detection of highly reliable fiducial markers under occlusion. *Pattern Recognition* 47, 6 (2014), 2280–2292.

[24] Ross Girshick. 2015. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*. 1440–1448.

[25] Terrell Glenn, Ananya Ipsita, Caleb Carithers, Kylie Peppler, and Karthik Ramani. 2020. StoryMakAR: Bringing stories to life with an augmented reality & physical prototyping toolkit for youth. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. 1–14.

[26] Google. 2022. Live AR View, Google Maps. https://arvr.google.com/ar/

[27] Google. 2022. MediaPipe. https://mediapipe.dev/

[28] Google. 2022. Mobile Net v3. https://tfhub.dev/google/tfjs-model/imagenet/mobilenet_v3_small_100_224/feature_vector/5/default/1

[29] Aakar Gupta, Bo Rui Lin, Siyi Ji, Arjav Patel, and Daniel Vogel. 2020. Replicate and reuse: Tangible interaction design for digitally-augmented physical media objects. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. 1–12.

[30] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. 2017. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*. 2961–2969.

[31] Robert Held, Ankit Gupta, Brian Curless, and Maneesh Agrawala. 2012. 3D puppetry: a kinect-based interface for 3D animation.. In *UIST*, Vol. 12. Citeseer, 423–434.

[32] Anuruddha Hettiarachchi and Daniel Wigdor. 2016. Annexing reality: Enabling opportunistic use of everyday objects as tangible proxies in augmented reality. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. 1957–1967.

[33] Valentin Heun, James Hobin, and Pattie Maes. 2013. Reality editor: programming smarter objects. In *Proceedings of the 2013 ACM conference on Pervasive and ubiquitous computing adjunct publication*. 307–310.

[34] Ke Huo and Karthik Ramani. 2016. Window-Shaping: 3D Design Ideation in Mixed Reality. In *Proceedings of the 2016 Symposium on Spatial User Interaction*. 189–189.

[35] Ke Huo, Tianyi Wang, Luis Paredes, Ana M Villanueva, Yuanzhi Cao, and Karthik Ramani. 2018. Synchronizar: Instant synchronization for spontaneous and spatial collaborations in augmented reality. In *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology*. 19–30.

[36] Adobe Inc. 2022. Adobe Aero. https://www.adobe.com/in/products/aero.html

[37] Adobe Inc. 2022. Adobe XD. https://www.adobe.com/in/products/xd.html

[38] Apple Inc. 2022. Reality Composer. https://developer.apple.com/augmented-reality/reality-composer/

[39] Gravity Sketch Inc. 2017. Gravity Sketch. https://www.gravitysketch.com/

[40] InVision Inc. 2022. InVision. https://www.invisionapp.com/

[41] Sketch Inc. 2022. Sketch. https://www.sketch.com/

[42] SketchUp Inc. 2022. Sketchup. https://www.sketchup.com/page/homepage

[43] Unreal Inc. 2022. Unreal Engine. https://www.unrealengine.com/en-US

[44] Hiroshi Ishii. 2004. Bottles: A transparent interface as a tribute to mark weiser. *IEICE Transactions on information and systems* 87, 6 (2004), 1299–1311.

[45] Hiroshi Ishii and Brygg Ullmer. 1997. Tangible bits: towards seamless interfaces between people, bits and atoms. In *Proceedings of the ACM SIGCHI Conference on Human factors in computing systems*. 234–241.

[46] Bret Jackson and Daniel F Keefe. 2016. Lift-off: Using reference imagery and freehand sketching to create 3d models in vr. *IEEE transactions on visualization and computer graphics* 22, 4 (2016), 1442–1451.

[47] Brett R Jones, Hrvoje Benko, Eyal Ofek, and Andrew D Wilson. 2013. IllumiRoom: peripheral projected illusions for interactive experiences. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 869–878.

[48] Brian Jordan, Nisha Devasia, Jenna Hong, Randi Williams, and Cynthia Breazeal. 2021. PoseBlocks: A toolkit for creating (and dancing) with AI. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 15551–15559.

[49] Hiroki Kaimoto, Kyzyl Monteiro, Mehrad Faridan, Jiatong Li, Samin Farajian, Yasuaki Kakehi, Ken Nakagaki, and Ryo Suzuki. 2022. Sketched Reality: Sketching Bi-Directional Interactions Between Virtual and Physical Worlds with AR and Actuated Tangible UI. *arXiv preprint arXiv:2208.06341* (2022).

[50] Seokbin Kang, Ekta Shokeen, Virginia L Byrne, Leyla Norooz, Elizabeth Bonsignore, Caro Williams-Pierce, and Jon E Froehlich. 2020. ARMath: augmenting everyday life with math learning. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. 1–15.

[51] Hirokazu Kato and Mark Billinghurst. 1999. Marker tracking and hmd calibration for a video-based augmented reality conferencing system. In *Proceedings 2nd IEEE and ACM International Workshop on Augmented Reality (IWAR'99)*. IEEE, 85–94.

[52] Annie Kelly, R Benjamin Shapiro, Jonathan de Halleux, and Thomas Ball. 2018. ARcadia: A rapid prototyping platform for real-time tangible interfaces. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. 1–8.

[53] Jarrod Knibbe, Tovi Grossman, and George Fitzmaurice. 2015. Smart makerspace: An immersive instructional space for physical tasks. In *Proceedings of the 2015 International Conference on Interactive Tabletops & Surfaces*. 83–92.

[54] Veronika Krauß, Michael Nebeling, Florian Jasche, and Alexander Boden. 2022. Elements of XR Prototyping: Characterizing the Role and Use of Prototypes in Augmented and Virtual Reality Design. In *CHI Conference on Human Factors in Computing Systems*. 1–18.

[55] David Ledo, Steven Houben, Jo Vermeulen, Nicolai Marquardt, Lora Oehlberg, and Saul Greenberg. 2018. Evaluation strategies for HCI toolkit research. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. 1–17.

[56] Gun A Lee, Gerard J Kim, and Mark Billinghurst. 2005. Immersive authoring: What you experience is what you get (wyxiwyg). *Commun. ACM* 48, 7 (2005), 76–81.

[57] Germán Leiva and Michel Beaudouin-Lafon. 2018. Montage: a video prototyping system to reduce re-shooting and increase re-usability. In *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology*. 675–682.

[58] 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.

[59] 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.

[60] Zhen Li, Michelle Annett, Ken Hinckley, Karan Singh, and Daniel Wigdor. 2019. Holodoc: Enabling mixed reality workspaces that harness physical and digital content. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. 1–14.

[61] Jian Liao, Adnan Karim, Shivesh Singh Jadon, Rubaiat Habib Kazi, and Ryo Suzuki. 2022. RealityTalk: Real-Time Speech-Driven Augmented Presentation for AR Live Storytelling. In *Proceedings of the 35th Annual ACM Symposium on User Interface Software and Technology*. 1–12.

[62] David Lindlbauer, Jens Emil Grønbæk, Morten Birk, Kim Halskov, Marc Alexa, and Jörg Müller. 2016. Combining shape-changing interfaces and spatial augmented reality enables extended object appearance. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. 791–802.

[63] Blair MacIntyre, Maribeth Gandy, Steven Dow, and Jay David Bolter. 2004. DART: a toolkit for rapid design exploration of augmented reality experiences. In *Proceedings of the 17th annual ACM symposium on User interface software and technology*. 197–206.

[64] 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.

[65] 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.

[66] Michael Nebeling and Maximilian 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, 333–337.

[67] Gary Ng, Joon Gi Shin, Alexander Plopski, Christian Sandor, and Daniel Saakes. 2018. Situated game level editing in augmented reality. In *Proceedings of the Twelfth International Conference on Tangible, Embedded, and Embodied Interaction*. 409–418.

[68] Nintendo. 2022. Mario Kart Live. https://mklive.nintendo.com/

[69] Nintendo. 2022. Nintendo Labo. https://www.nintendo.co.uk/Nintendo-Labo/Nintendo-Labo-1328637.html

[70] OpenCV. 2022. OpenCV. https://opencv.org/

[71] Youngki Park and Youhyun Shin. 2021. Tooee: A Novel Scratch Extension for K-12 Big Data and Artificial Intelligence Education Using Text-Based Visual Blocks. *IEEE Access* 9 (2021), 149630–149646.

[72] Shwetha Rajaram and Michael Nebeling. 2022. Paper Trail: An Immersive Authoring System for Augmented Reality Instructional Experiences. In *CHI Conference on Human Factors in Computing Systems*. 1–16.

[73] Gonzalo Ramos, Christopher Meek, Patrice Simard, Jina Suh, and Soroush Ghorashi. 2020. Interactive machine teaching: a human-centered approach to building machine-learned models. *Human–Computer Interaction* 35, 5-6 (2020), 413–451.

[74] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. 2016. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 779–788.

[75] Patrick Reipschläger, Severin Engert, and Raimund Dachselt. 2020. Augmented displays: Seamlessly extending interactive surfaces with head-mounted augmented reality. In *Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems*. 1–4.

[76] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. 2015. Imagenet large scale visual recognition challenge. *International journal of computer vision* 115, 3 (2015), 211–252.

[77] Alpay Sabuncuoglu and T Metin Sezgin. 2022. Prototyping Products using Web-based AI Tools: Designing a Tangible Programming Environment with Children. In *6th FabLearn Europe/MakeEd Conference 2022*. 1–6.

[78] Nazmus Saquib, Rubaiat Habib Kazi, Li-Yi Wei, and Wilmot Li. 2019. Interactive body-driven graphics for augmented video performance. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. 1–12.

[79] Hartmut Seichter, Julian Looser, and Mark Billinghurst. 2008. ComposAR: An intuitive tool for authoring AR applications. In *2008 7th IEEE/ACM International Symposium on Mixed and Augmented Reality*. IEEE, 177–178.

[80] Paden Shorey and Audrey Girouard. 2017. Bendtroller: An exploration of in-game action mappings with a deformable game controller. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. 1447–1458.

[81] Maximilian Speicher, Katy Lewis, and Michael Nebeling. 2021. Designers, the stage is yours! medium-fidelity prototyping of augmented & virtual reality interfaces with 360theater. *Proceedings of the ACM on Human-Computer Interaction* 5, EICS (2021), 1–25.

[82] Jürgen Steimle, Andreas Jordt, and Pattie Maes. 2013. Flexpad: highly flexible bending interactions for projected handheld displays. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 237–246.

[83] 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*. 166–181.

[84] Three.js. 2022. Ricardo Cabello. https://threejs.org/

[85] Tiffany Tseng, Yumiko Murai, Natalie Freed, Deanna Gelosi, Tung D Ta, and Yoshihiro Kawahara. 2021. PlushPal: Storytelling with interactive plush toys and machine learning. In *Interaction design and children*. 236–245.

[86] Nicolas Villar, Daniel Cletheroe, Greg Saul, Christian Holz, Tim Regan, Oscar Salandin, Misha Sra, Hui-Shyong Yeo, William Field, and Haiyan Zhang. 2018. Project zanzibar: A portable and flexible tangible interaction platform. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. 1–13.

[87] James A Walsh, G Stewart Von Itzstein, and Bruce H Thomas. 2013. Tangible agile mapping: ad-hoc tangible user interaction definition. In *AUIC*. Citeseer, 3–12.

[88] James A Walsh, Stewart Von Itzstein, and Bruce H Thomas. 2014. Ephemeral interaction using everyday objects. In *Proceedings of the Fifteenth Australasian User Interface Conference-Volume 150*. 29–37.

[89] Tianyi Wang, Xun Qian, Fengming He, Xiyun Hu, Yuanzhi Cao, and Karthik Ramani. 2021. GesturAR: An Authoring System for Creating Freehand Interactive Augmented Reality Applications. In *The 34th Annual ACM Symposium on User Interface Software and Technology*. 552–567.

[90] Tianyi Wang, Xun Qian, Fengming He, Xiyun Hu, Ke Huo, Yuanzhi Cao, and Karthik Ramani. 2020. CAPturAR: An augmented reality tool for authoring human-involved context-aware applications. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology*. 328–341.

[91] Zeyu Wang, Cuong Nguyen, Paul Asente, and Julie Dorsey. 2021. Distanciar: Authoring site-specific augmented reality experiences for remote environments. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. 1–12.

[92] Matt Whitlock, Jake Mitchell, Nick Pfeufer, Brad Arnot, Ryan Craig, Bryce Wilson, Brian Chung, and Danielle Albers Szafir. 2020. MRCAT: In situ prototyping of interactive AR environments. In *International Conference on Human-Computer Interaction*. Springer, 235–255.

[93] Randi Williams, Stephen P Kaputsos, and Cynthia Breazeal. 2021. Teacher Perspectives on How To Train Your Robot: A Middle School AI and Ethics Curriculum. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 15678–15686.

[94] Robert Xiao, Chris Harrison, and Scott E Hudson. 2013. WorldKit: rapid and easy creation of ad-hoc interactive applications on everyday surfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 879–888.

[95] Hui Ye and Hongbo Fu. 2022. ProGesAR: Mobile AR Prototyping for Proxemic and Gestural Interactions with Real-world IoT Enhanced Spaces. In *CHI Conference on Human Factors in Computing Systems*. 1–14.

[96] Ya-Ting Yue, Yong-Liang Yang, Gang Ren, and Wenping Wang. 2017. SceneCtrl: Mixed reality enhancement via efficient scene editing. In *Proceedings of the 30th annual ACM symposium on user interface software and technology*. 427–436.

[97] Clement Zheng, Peter Gyory, and Ellen Yi-Luen Do. 2020. Tangible interfaces with printed paper markers. In *Proceedings of the 2020 ACM designing interactive systems conference*. 909–923.

[98] Qian Zhou, Sarah Sykes, Sidney Fels, and Kenrick Kin. 2020. Gripmarks: Using Hand Grips to Transform In-Hand Objects into Mixed Reality Input. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. 1–11.

[99] Zhongyi Zhou and Koji Yatani. 2022. Gesture-aware Interactive Machine Teaching with In-situ Object Annotations. *arXiv preprint arXiv:2208.01211* (2022).

[100] Fengyuan Zhu and Tovi Grossman. 2020. Bishare: Exploring bidirectional interactions between smartphones and head-mounted augmented reality. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. 1–14.