PrototypeVR: A VR Rapid Prototyping System for Smart Devices

Bethany Lu, Joshua Yang, Frederick Kim, Trista Hu, Xinwei Zhuang, James Smith
UC Berkeley EECS, Berkeley, CA, USA
{bethanylu, joshua.yang, kwk, yue_hu0414, xinwei_zhuang, james.smith}@berkeley.edu

INTRODUCTION

The process of product design involves many phases before it being released to the public, among which prototyping is the most critical step because it allows designers iteratively test their design and measure usability, and allows marketers to collect stakeholder and user feedback.

With the advent of 3D Printing, prototyping has become more efficient and flexible. In fact, a subset of prototyping techniques known as Rapid Prototyping (RP), which involves CAD models being 3D printed to yield short iteration times, has proven to greatly improve prototyping efficiency [18]. However,building prototypes can still be costly and time inefficient, as 3D printing could take overnight, and printing materials also cost a lot of money. Furthermore, when it comes to "smart" devices which involve both hardware and software components, it is hard to integrate into a 3D printed prototype.

Efforts such as Zygote (Karki *et al.*) [7] have been made to improve this situation. Zygote is a prototyping software that easily and with little coding simulates IoT systems without the need for designers and developers to know all aspects of running production-scale IoT systems. Previous studies have also utilized the AR/VR space to overcome some of the physical limitations of prototyping, as researchers Ko *et al.* [10] have designed a virtual prototyping system that utilizes physical controls but puts the user in VR to test out different interface designs. Boletsis *et al.* [2] point out that past human computer interaction research has had limited problem solving capacity for the general public. Hence, our project focuses on the practicality of 3D prototyping in VR with simulated digital screens, which would potentially improve the prototyping process for smart devices. With testing prototypes in VR, it effectively brings costs of iteration down, and provides a simulated environment in which software-hardware integration can be both accurate and highly flexible.

To explore rapid prototyping for smart devices in virtual reality, we introduce PrototypeVR, a system that integrates industry-grade software to build a coherent and user-friendly prototyping system for designers. We propose a pipeline, using 3DS MAX for CAD modelling, Figma for UI design, and integrating 3D model and user interface in Unity for virtual prototyping. PrototypeVR consists of a Figma extractor plugin and a Unity importer plugin that allow users to import Figma files and the transitions between frames in the files into a Unity environment where 2D UI can be easily synced and projected onto 3D models, and the prototyping process can be rapidly altered, play-tested, and iterated upon.

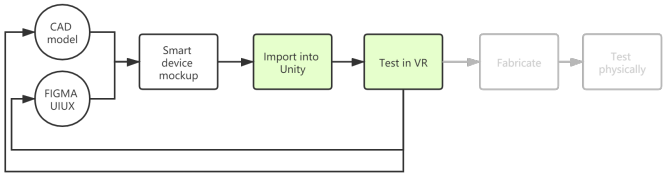

Figure 1. Diagram of our approach using Unity to prototype.

We hypothesize that PrototypeVR can provide cost and time-effective prototyping for smart devices, where designers have more flexibility and ease of access for choosing testing environments and product materials and dimensions. By using VR to simulate software and hardware of devices at a very low cost, our project serves to immensely improve upon previous innovations of rapid prototyping, taking into consideration the rising need for designers to prototype smart devices of the future.

RELATED WORK

Past work encompasses current prototyping methods in both physical and virtual environments, human computer interaction with 2D screens in VR and accessible VR, and existing VR applications that attempt to create a prototyping environment. We will discuss each of the categories.

Prototyping

Prior to virtual prototyping techniques coming in, rapid prototyping is widely adopted for its cost and time effectiveness [18]. Different RP technologies involve materialisation to build prototypes from CAD software. However, confirms that 3D printing prototypes is a pain-point to prototypers and designers. Printing wireframes of objects, such as WirePrint [13], can largely reduce time and material efficiency. Coutts *et al.* [3] compared different prototyping approaches, including physical models, CAD prototype, AR and VR prototypes. The

ISBN 978-1-4503-2138-9.
DOI: 10.1145/1235

authors conclude that VR prototyping excelled in the appearance, scale and cost.

Visual prototyping has been implemented in many industrial fields using AR and VR to visualize objects and do assembly and disassembly testing, structural and ergonomic analysis. Park *et al.* [17] developed a low-cost AR prototyping system that combines AR-based tangible interaction with functional behavior simulation, which is effective in helping users rapidly prototype different digital interfaces and test certain interactions from users while providing tangible interactions in an AR-based environment. Karki *et al.* [7] created Zygote to offer a centralized framework for IoT system prototyping for smart devices. It demonstrates the usability and a low barrier to entry for designers. Ko *et al.* [10] developed a VR prototyping system which can be used to change the pose and material of a virtual product, as well as assembly, disassemble, manipulate. Nebeling and Madier [14] used VR prototyping to evaluate the see-through features of mobile work machines. Their project demonstrated the advantage of VR prototyping that other methods of prototyping cannot achieve, ie., immersive, cost-effective, time-effective and light weighted which can be tested anywhere with basic equipment.

Appliancizer showcases a pipeline to transform digital user interfaces into physical devices by automatically creating PCBs based on HTML elements [5]. This provides an option to quickly create the physical device once virtual prototypes have been successfully tested.

Human Computer Interaction in VR

A key focus of smart device prototyping is to make sure the prototyping and testing process is accessible and usable. There have been past attempts to bring 2D interfaces to 3D virtual spaces in order to compare the usability of the two formats [6]. 2D interfaces are shown to be preferred but the addition of 3D models can potentially change the way users interact with the screen. Recent study investigates using virtual reality as a tool for design review shows that immersive VR are useful simulation tools with excellent cognitive performance [16]. Despite issues with motion sickness due to technical limitations, they show that there is high ergonomic quality with high levels of attention and concentration.

When bringing a 2D screen to a virtual environment, there are multiple ways to incorporate the interface with the functionality of VR. Scalable vector graphics (SVG) provide an avenue to making the user interfaces for 3D prototyping. SVGs has the option to be flexible with backward compatibility with interactivity through the framework of Unity [11, 15]. With each interaction, different states of the user interface can be simulated using dynamically generated SVG [12]. Another option would be to use Unity plugins that allow browser functionality. Plugin technology can allow us to simulate already existing user experience testing software such as Figma [19].

When designing software that will be used to test prototypes, accessible design will be crucial in order to obtain the same data points that a real prototype can offer. Dombrowski *et al.* highlights ways to design more inclusive VR experiences which is crucial to making the prototyping process accessible [4]. The outline provided serves as a guideline to design a "diversity of ways to participate so that everyone has a sense of belonging" (Susan Goltsman), which is crucial to user testing.

Examine on existing VR Apps

The idea of using virtual reality for prototypes has generated a great deal of interest and many apps have been developed for practical use. SketchStudio is a prototyping tool for generating, sharing, and reviewing an animated design scenario involving complex design subjects [9]. It allows designers to define the spatiotemporal experience flow and animate a 2.5D virtual space to visualize the user experience. However, the effectiveness and fidelity in a 3D virtual world is still uncertain. The tool has limitations when it comes to visualizing an idea that needs to be illustrated in 3D form. The result was unnatural due to a mismatch in perspective when changing the viewpoint in a 3D space.

Figure 2. Bimanual asymmetric sketching drawing function in VR prototyping, Benne *et al.* 2012

Daniel *et al.* implemented several studies investigating tools and methodologies for artist-scientist-technologist collaboration in designing multivariate virtual reality visualizations [8]. They concluded that effective artist-accessible design tools must support quick sketch-like visual exploration, sufficient control over form to represent complex subjects in science, and an ability to capture the animated views and interactive scenarios that are critical to VR experiences. Bennes *et al.* [1] also studied the implementation of virtual screens in virtual environments for multidisciplinary design for both mechanical engineering and designers. They found that using VR eased multidisciplinary product design process, and provided clear and efficient design and development guidelines for Virtual Reality 3D user interface design.

SYSTEM DESCRIPTION AND IMPLEMENTATION

PrototypeVR offers a workflow that allows researchers and innovators to quickly create prototypes with virtual screens by allowing users to utilize existing tools that are tailored for UI/UX or 3D modeling. It is composed of two plugins: one to gather information from Figma, and the other to display and format that information in Unity. An alternative plugin can be used to import 3D models from 3DS MAX, though it was not used in this project.

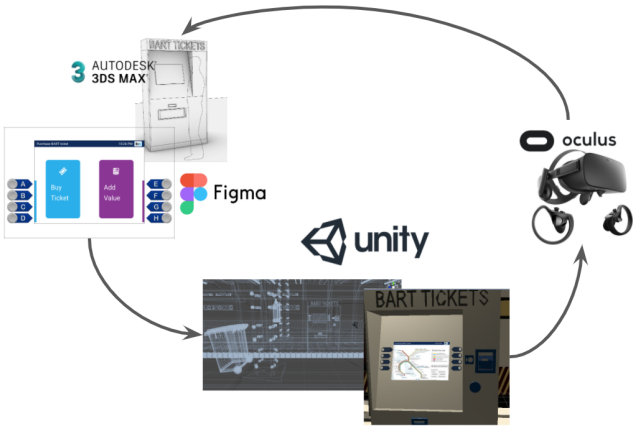

Figure 3. An overview of the various components of PrototypeVR: 3DS MAX/Figma, Unity, and a VR headset.

In this pipeline, users can work in parallel to create the 3D model on modelling software and 2D interface on Figma. The Figma plugin is then used to create a JSON file that holds the necessary information needed for user interaction to work on Unity. To bring in the prototyping information from Figma into Unity, a Unity plugin is used to read the JSON file and map user interactions to buttons on the 3D model in Unity.

Users are then able to run the Unity application in a virtual environment to test the prototype with various interaction types. For example, users can interact with buttons and sensors that have been created on the 3d model to simulate button presses in the Figma prototype. They can also interact with the screen directly as if they are using a touch screen device. The system also supports both types of interactions at once. Since the JSON file is easily exportable using the Figma plugin, the user interface can also be quickly changed to test different implementations, enabling rapid VR prototyping of smart devices.

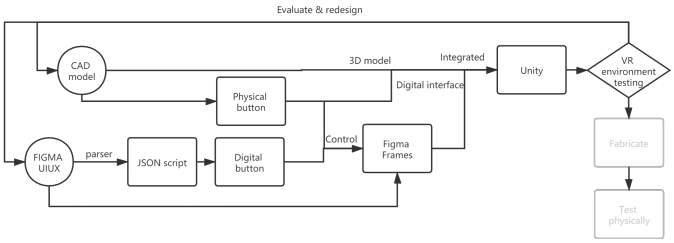

Figure 4. A detailed flowchart describing how our plugins feed data through our system.

Details about how the Figma plugin and how the Unity plugin reads the file to create a working VR prototype are presented as follows.

Figma Plugin

Our Figma plugin*takes advantage of the plugin API Figma provides. We first find the starting frame of the Figma prototype which the plugin API provides and store the name in the JSON data. This will be used to tell Unity which screen to show first. Starting from the current page that the plugin is executed, the plugin iterates through each frame on the page and fetches the name, id, dimensions, and all the prototype reactions, which we call hotspots, in the frame.

Each prototype reaction, or hotspot, contains an interactable element that performs some action to the user interface. The hotspots are located by recursively searching within each frame and seeing if an element in the frame has interactable properties. We obtain the element's name, position, dimensions, whether or not it is visible, properties of the interaction, and the id of the destination frame.

For the purposes of this project, we assume that each hotspot is a button, which has the interaction type ON_CLICK and transitions the current screen to a different frame. We take note of the visibility of the hotspots so that the Unity plugin would know which buttons to display and which buttons to map to physical objects in the prototype.

Lastly, we obtain the actual visual screens by manually exporting the screens to PNG images and upload them into the Unity project.

Unity Plugin

Once the Figma plugin has provided us with JSON data and we've also captured the PNGs of the Figma frames, we now feed in this data into our Unity plugin. The plugin first maps the Figma PNG files to a Unity canvas (which we assume would be placed somewhere on the prototype CAD model) determined by the user, resulting in the Unity canvas rendering the Figma PNG photos. When the Unity project starts running, we first load the starting frame to the Unity canvas. Whenever we render a Figma frame, we loop through the hotspots of the frame to be loaded, and add button interactions for each hotspot. As previously mentioned, our system supports physical button interactions with submeshes of our prototype as well as on-screen button interactions, and as we add button interactions we process these hotspots differently based on whether the hotspot is visible or not on the Figma screen.

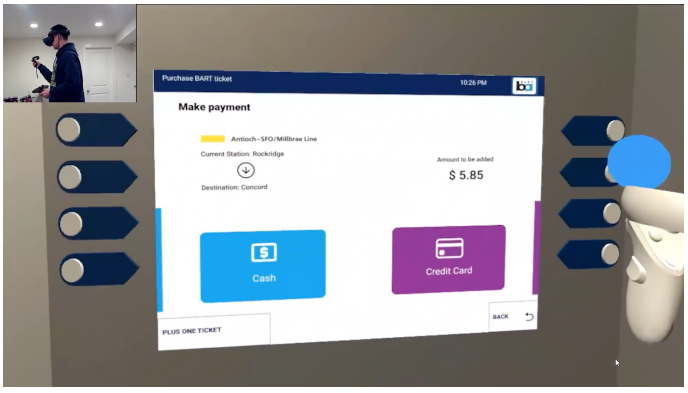

Figure 5. Experiment process of VR group.

*Link to project code: github.com/Joshuayangg/PrototypeVR

If a hotspot is invisible in Figma, the Unity plugin will try to find a 3D submesh of our prototype so that the designated physical submesh will behave as a physical button. We do this by searching our Unity scene for a name match between the Unity mesh and the invisible hotspot name. Once we've found the match, assuming the Unity object has a FigmaInteractable script attached to it, as well as a collider that helps detect collisions, we simply set the transition ID element in the FigmaInteractable script to the transition ID of the hotspot. With the ID set, when the user collides with the object in question, we look for the transition ID in the associated FigmaInteractable script and then switch our frame to the frame with the new ID.

If a button is visible in Figma, then we take a different approach to putting it in Unity. First, we instantiate an invisible Unity UI button (which is also a FigmaInteractable with an attached collider) and scale the dimensions of it to match the width, height, and position of the Figma hotspot. We achieve this by taking into consideration the width and height of both our Figma frame (provided in the JSON) and our Unity canvas, and scaling accordingly. Then, we set the transition ID of the invisible Unity button to the transition ID specified in the hotspot as we do in the other case, and also set a custom onClick function for the invisible button to render the next frame as well to support user interactions that involve clicking the buttons in contrast with simply colliding with them.

With our implementation of mapping JSON hotspots and transitions to on-screen and physical buttons and objects, as well as using name matching to determine the right png file to load onto our Unity canvas at any given state, we successfully utilize both our Figma and Unity plugins to recreate a seamless user experience designed in Figma and executed on Unity.

EVALUATION

Prototype Methods Comparison

We compare the PrototypeVR with 3D printing, physical prototyping and AR prototyping methods to identify the privileges and limitations of these prototype methods, which hopefully helps us revise existing functions and explore the potential features needed to improve PrototypeVR.

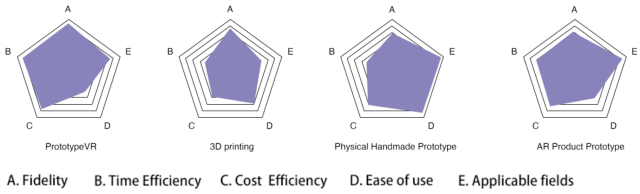

Figure 6. Various methods of rapid prototyping and their effectiveness based on five axes.

Evaluation test

Because of COVID-19, we were unable to conduct a full user study for our product. However, we tested PrototypeVR internally and made estimations and hypotheses of potential results we would see if we were to carry out our experiment. In our proposed study, we would have testers make an assessment report for two prototyping methods for designing the Bart machine based on ergonomics, accessibility and visual effect aspects and quality. We would put the test audience into two groups. One group would use PrototypeVR for testing and the other would make handmade physical models instead. We would invite each participant to conduct a low-fi prototype test in the Jacobs lab, record the time and any difficulties they had during the test. After they finish the experiment, we would then ask them to fill a post-test survey to rate the ease of use, test effect and money efficiency from 1 (low) to 5 (high). Based on estimations and predictions from members of our group who have both tested PrototypeVR and have extensive experience designing prototypes without VR, we hypothesize results that look similar to what is shown in the chart below:

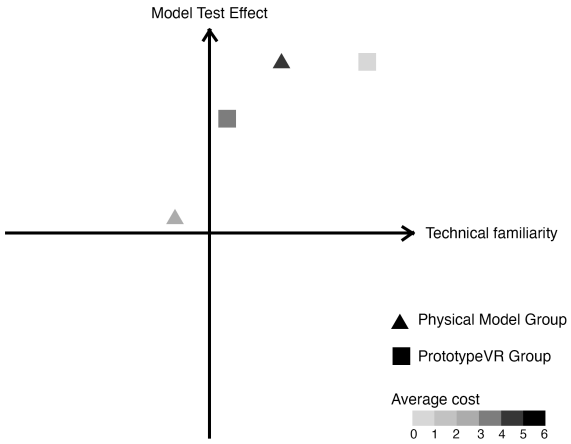

Figure 7. Model test effect given various degrees of technical familiarity.

The result in section 3.1 shows that compared to the physical handmade prototype, PrototypeVR provides higher fidelity, time efficiency and cost efficiency, which contribute to the improved performance. Aim to quantitatively calculate the efficiency of product testing under the same time and money cost unit, we estimate the final performance using the formula:

$$P = \frac{E}{\sqrt{mt}}$$

Where E is the mean of model test effect, m is the mean of money cost and t is the mean of time cost. Each participant gets 5, 1.27, 0.90, 1.06 respectively. We hypothesize that PrototypeVR provides a better test effect than the physical handmade prototype. Also, we predict that PrototypeVR would save a lot of time when compared to other methods that achieve similar effects. However, PrototypeVR may not be applicable for people who don't own an Oculus device, which greatly increases the prototype cost and hinders accessibility.

DISCUSSION & LIMITATION

From the performance evaluation, PrototypeVR demonstrates various advantages to other comparable prototyping methods: easy modifications, undemanding interdisciplinary cooperation and repetitive use. Though it requires certain background knowledge to work with Unity and virtual reality, it still takes significantly less time than 3D printing physical models, which

involves an overwhelming workload with trial and error testing. Besides, once users find deficiencies in their prototyped models, they can edit and re-import them easily within several minutes, which contributes to the overall prototyping efficiency of PrototypeVR. Another advantage PrototypeVR has is that the business logic from Figma can be directly imported into Unity without considering the hidden mechanics and transitions. Usually when designing prototypes, the Figma logic has to be replicated by code on a microcontroller or in a software environment, but with PrototypeVR, the user is able to directly use the Figma transitions in Unity without having to reprogram them, thus accelerating the interdisciplinary corporation on product design.

This tool also has drawbacks that its inaccessibility to users who are not familiar with Unity 3D because using the plugin does require Unity knowledge. However, this can be improved by further plugin automation. PrototypeVR is also not accessible to users without VR equipment. Though the design market is increasingly adopting more VR technology, currently VR equipment is not widely available and only serves niche markets. For students who want to use PrototypeVR, the university should be able to provide them with VR headsets and compatible computers. With this barrier to entry, this may leave a lot of potential users of PrototypeVR out of the equation.

On the usability side, using VR controllers to interact with prototypes can limit the fidelity of interaction, as individual finger movements are not easily mapped in VR, and even with hand tracking technology, it is still very difficult to effectively simulate haptic feedback when users interact with VR prototypes. For now, PrototypeVR can send haptic pulses to the controller upon impact with a physical or on-screen button, but the system is inflexible to simulate other types of sensations. Furthermore, unless the user is using an Oculus Quest, or utilizing a third party hand tracker like the LeapMotion, tracking fingers and using fingers to interact with VR prototypes is infeasible.

Lastly, as previously mentioned, PrototypeVR currently only supports ON_CLICK transitions from Figma, and simulating other types of transition triggers may be difficult. For example, prototyping smart fridge interface in VR with opening fridge door as a trigger can only be simulated with collision with the fridge handle, which is inaccurate in timing and low in fidelity. However, despite all of these limitations to the PrototypeVR system, there seems to be reasonable compromises to be made. The benefits of prototyping in VR outweigh the cons in many situations, particularly ones that don't involve all aspects of the prototyping experience needing to have very high fidelity.

CONCLUSION

PrototypeVR provides a time-efficient and cost efficient product prototyping tool. It facilitates interdisciplinary design because it enables designers to avoid the mechanical prototyping aspects of prototyping, and have more freedom in design and user interaction.

PrototypeVR also has potential in the learning and professional environments because universities do not have the financial ability to access current prototyping technologies, and courses involving testing and exploring design prototypes would be time and material costly. For example, in a design class that involves user testing, PrototypeVR would be a ideal option to accelerate the testing process, and give students more time and flexibility to change elements of the prototype such as user interface, and the physical appearance. This would drastically speed up the design process and lessen the cost of prototyping. The only caveat to this is the initial cost of VR setups, but we believe that PrototypeVR can be the more cost effective alternative in the long run due to the recurring costs and longer timeframe of 3D printing.

We also see prototyping in VR being a groundbreaking tool to introduce hobbyists and designers to the world of physical product prototyping. Prototyping in VR can test their products without investing a lot of money, material or time.

Looking at the potential growth and impact of PrototypeVR, we are certain that this tool will positively impact the prototyping space by making it more accessible for users unfamiliar with product prototyping as well as facilitate product prototyping for those who are familiar with the process.

FUTURE WORK

Although we were able to build a working pipeline of creating prototypes in VR, there are many improvements that can be made. One crucial step would be conducting in-person usability tests with at least 10 people to obtain actual data in the ease of use of the workflow and determine best applicable scenarios for users employing PrototypeVR rather than other prototype methods. The COVID-19 pandemic has hindered plans to conduct these tests, so we were only able to perform the quantitative and qualitative analysis of PrototypeVR with group members. We would likely follow the same evaluation process as detailed previously with the multiple trials giving us more data points.

Another logical next step would be to incorporate more types of interactions, including grab and pull, taps with multiple finger counts, swipe gestures, etc. to provide the end user with more options in creating a realistic, complex smart device. By enabling nearly all the prototyping interactions Figma provides, Prototype VR will be able to provide a complete realistic user interface in a virtual reality environment.

Improvements to the Figma plugin will also allow a more seamless user experience by exporting all the necessary files all at once rather than having to copy the JSON file and manually exporting the screen image files. Also, making the Unity plugin more accessible to people without Unity knowledge would be great as well; abstracting away and automating collider and script setups would be really helpful for people with less Unity experience and for speeding up the prototyping timeline as well.

In addition to improvements on the user interface, we hope to incorporate more of the previous research done on the pipeline from 3D models to VR. This will allow Prototype VR to have simple workflows from CAD models and the 2D interfaces to VR that we currently have. This would allow users to test more complex smart devices that include joints and constraints.

We also hope to add the ability to quickly modify some of the properties of the 3D model such as the dimensions, tilt, and elevation. Therefore, should there be any last minute changes, those can be edited directly in Unity rather than going back to the CAD model itself. These changes would be primarily for improving the VR experience and helping it be more realistic rather than major changes to the 3D model.

We acknowledge that this research is not complete and is missing vital information, particularly with regards to the lack of a comprehensive user test. We hope to incorporate these improvements in the future.

REFERENCES

[1] Lionel Bennes, Florence Bazzaro, and Jean-Claude Sagot. 2012. Virtual reality as a support tool for ergonomic-style convergence: multidisciplinary interaction design methodology and case study. In *Proceedings of the 2012 Virtual Reality International Conference (VRIC '12)*. Association for Computing Machinery, 1–10. DOI: http://dx.doi.org/10.1145/2331714.2331742

[2] Costas Boletsis, Jarl Erik Cedergren, and Stian Kongsvik. 2017. HCI Research in Virtual Reality: A Discussion OF Problem-solving. (2017).

[3] Euan Ross Coutts, Andrew Wodehouse, and Jason Robertson. 2019. A Comparison of Contemporary Prototyping Methods. 1, 1 (2019), 1313–1322. DOI: http://dx.doi.org/10.1017/dsi.2019.137

[4] Matt Dombrowski, Peter A. Smith, Albert Manero, and John Sparkman. 2019. Designing Inclusive Virtual Reality Experiences. In *Virtual, Augmented and Mixed Reality. Multimodal Interaction*, Jessie Y.C. Chen and Gino Fragomeni (Eds.). Springer International Publishing, Cham, 33–43.

[5] Jorge Garza, Devon J. Merrill, and Steven Swanson. 2020. Appliancizer: Transforming Web Pages into Electronic Gadgets. In *Adjunct Publication of the 33rd Annual ACM Symposium on User Interface Software and Technology (UIST '20 Adjunct)*. Association for Computing Machinery, New York, NY, USA, 142–144. DOI:http://dx.doi.org/10.1145/3379350.3416158

[6] Tatu V. J. Harviainen, Lauri Svan, and Tapio Takala. 2007. Usability Testing of Virtual Reality Aided Design: Framework for Prototype Development and a Test Scenario. DOI:http://dx.doi.org/10.1.1.589.7937

[7] D. Karki, A. Kaliki, and R. P. Rustagi. 2015. Zygote: A Framework for Prototyping Smart Devices. In *2015 International Conference on Advanced Computing and Communications (ADCOM)*. 1–6. DOI: http://dx.doi.org/10.1109/ADCOM.2015.7

[8] Daniel F. Keefe, Daniel Acevedo, Jadrian Miles, Fritz Drury, Sharon M. Swartz, and David H. Laidlaw. 2008. Scientific Sketching for Collaborative VR Visualization Design. *IEEE Transactions on Visualization and Computer Graphics* 14, 4 (July 2008), 835–847. DOI: http://dx.doi.org/10.1109/TVCG.2008.31

[9] Han-Jong Kim, Chang Min Kim, and Tek-Jin Nam. 2018. SketchStudio: Experience Prototyping with 2.5-Dimensional Animated Design Scenarios. In *Proceedings of the 2018 Designing Interactive Systems Conference (DIS '18)*. Association for Computing Machinery, 831–843. DOI: http://dx.doi.org/10.1145/3196709.3196736

[10] G. Ko, S. Ryu, S. Nam, J. Lee, and K. Suh. 2019. Design of Virtual Reality Prototyping System and Hand-Held Haptic Controller. 11, 4 (2019), 72–75. DOI: http://dx.doi.org/10.7763/IJCTE.2019.V11.1245

[11] Cameron L. McCormack, Kim Marriott, and Bernd Meyer. 2004. Constraint SVG. In *Proceedings of the 13th international World Wide Web conference on Alternate track papers & posters (WWW Alt. '04)*. Association for Computing Machinery, 310–311. DOI: http://dx.doi.org/10.1145/1013367.1013450

[12] R. Meenakshi, G. Jayalekshmi, S. Hariram, Shiju Sathyadevan, and M. G. Thushara. 2015. Visualization with Charting Library Based on SVG for Amrita Dynamic Dashboard. 58 (2015), 371–379. DOI: http://dx.doi.org/10.1016/j.procs.2015.08.036

[13] Stefanie Mueller, Sangha Im, Serafima Gurevich, Alexander Teibrich, Lisa Pfisterer, François Guimbretière, and Patrick Baudisch. 2014. WirePrint: 3D Printed Previews for Fast Prototyping. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology (UIST '14)*. Association for Computing Machinery, New York, NY, USA, 273–280. DOI: http://dx.doi.org/10.1145/2642918.2647359

[14] 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 - CHI '19*. ACM Press, 1–13. DOI: http://dx.doi.org/10.1145/3290605.3300826

[15] Deborah Nolan. *Interactive and Animated Scalable Vector Graphics and R Data Displays*.

[16] Daniel Paes and Javier Irizarry. 2018. A Usability Study of an Immersive Virtual Reality Platform for Building Design Review: Considerations on Human Factors and User Interface. DOI: http://dx.doi.org/10.1061/9780784481264.041

[17] Hyungjun Park, Hee-Cheol Moon, and Jae Yeol Lee. 2008. Tangible augmented prototyping of digital handheld products. (Dec 2008). https://doi.org/10.1016/j.compind.2008.09.001

[18] D.T Pham and R.S Gault. 1999. A comparison of rapid prototyping technologies. (Jun 1999). https://www.sciencedirect.com/science/article/pii/S0890695597001375

[19] Jesús Sánchez Cuadrado and Jesús Molina. 2006. *A Plugin-Based Language to Experiment with Model Transformation*. DOI: http://dx.doi.org/10.1007/11880240_24 Pages: 350.

APPENDIX

A. COMPARISON OF DIFFERENT PROTOTYPING APPROACHES

	PrototypeVR	3D Printing	Physical prototype	AR Prototype
Function	3D with 2D	physical	physical	3D overlays
Pipeline	CAD - Unity - Figma - JSON - Unity	CAD - STL file- Gcode - Materials - Print	Sketch - Handwork - Physical model	CAD - AR overlay
Tools	CAD, Figma, Unity, Oculus	CAD and 3D printer	Cardboard, Laser cutting, Microcontroller, etc	C4D, AR Software, mobile device or AR headsets
Fidelity	High	High in physical model	Low	Medium
Time	High	Super low	Low	Medium
Cost	High	Low	Medium	Medium
Ease of use	Low	Medium low	Low	Medium
Fields	Mechanical engineer, product design, experience design	Product design, Architectural design	Mostly all fields	Mostly all fields
Pro	cost and time efficient	physical embodiment	lower learning curve	prototyping additions to existing products, real-world environment
Cons	expensive equipment, background knowledge	no digital functions, error prone	cost, time, and material inefficient	low interaction, low tracking quality

B. EVALUATION OF VR PROTOTYPING AND PHYSICAL PROTOTYPING

The evaluation matrix is scaled between 1(lowest) to 5 (highest) if not specified.

Evaluation	PrototypeVR		Physical modeling	
	User 1	User 2	User 3	User 4
Technical Familiarity	5	3	4	2
Time Cost/h	1	2	10	4
Money Cost	1	5	3	2
Model Test Effect	5	4	5	3
Overall Performance	5	1.27	0.9	1.06