We do some storyboarding and sketching. Then we do some 'acting it out' because, paper prototyping kind of works, but it doesn't really get the feeling of what you want to do. So often we do kind of role-play of like, if you were in VR, what would you do? (P10-PD)

Our overall results revealed that prototyping for AR/VR was open-ended and non-representative of the real VR experience. Methods like role-playing or physical prototyping can simulate the real experience to some extent but were still not considered to be accurate in visual aspects (as has been shown in prior work [37]) and many other variables such as lighting and audio. For example, one UX designer reported the ineffectiveness of available methods in the representation of the real experience:

In either AR or VR settings, the world is all around you. So the tilt, frame, or angles to show actually matter compared to 2D [prototyping]…it's going to involve multiple people…it's inevitable if we're making [mock up] videos from the objects that we create with paper, those objects are relatively small compared to our body [when showing the interactions], so the whole scene will look a little bit messy. (P17-PD)

5. Difficult to plan and simulate motion

Another aspect of having limited control over users' actions was the difficulty designers faced when providing users with a targeted experience. Designing AR experiences can involve multiple users with different physical characteristics, different usage trends, and a variety of environments where the application may be used. Our participants reported having difficulty anticipating users' behavior and the way users hold their phones based on their different preferences in designing marker-based AR experiences:

We have very practical usability issues...It's really awkward to hold a phone above a page…I actually programmed it to hold it perpendicular. But a lot of people go directly above. (P12-Dx)

Another participant shared difficulties in simulating multiple use case scenarios as a limitation of existing AR prototyping tools. This participant described a potential workaround, but felt that it involved a lot more coding effort than she was willing to expend during prototyping:

… to demonstrate that kind of process [different user scenarios] we have to use a lot of animation tools to simulate that…I can make 2 to 3 simple codes to access turnarounds or the phone's orientations because I know how to code. But, that kind of thing would be more challenging for designers because if they don't know how to code, they have to simulate everything in animation tools. (P18-PD)

6. Difficult to design story-driven immersive experiences

Storytelling is a crucial aspect of creating immersive experiences [3]. In immersive experiences, end users are not just watching a story, but are actually a part of the story. While storytelling matters both in AR and VR, our participants explained some differences that they had experienced. In particular, participants who had worked on both AR and VR reported that they had an easier time authoring a compelling experience for VR applications. Compared to AR, VR lent itself more to storytelling due to the encompassing and limited nature of the experience:

I see VR more as a storytelling medium than AR...That's not always true, but AR tends to lend itself towards shorter experiences. A lot of AR experiences are collection-based experiences. So, they're short. They don't involve much story unless there's a background story to why you should be collecting an object. (P20-Dx)

Since the story in VR is driven by the context and the environment around the user, a key challenge in VR is creating a virtual environment that tries to provide the sensations and engagement of the real world.

On the other hand, the restricted environment of VR actually reduces the distractions of the real world. In contrast, AR relies on an uncontrolled physical environment to drive the story. In fact, AR creators gave several examples of problems that they had in understanding where augmentation would affect the user experience and how to maintain users' attention while experiencing the real world around them:

We have a lot of questions within AR; like, how do we want the user to look around and what do we want them to see while they're already experiencing the real world? How we are going to maintain their attention, and for how long before they're distracted by the real world. (P18-PD)

BARRIERS IN IMPLEMENTING AND TESTING AR/VR APPLICATIONS

Another set of barriers that emerged in our interviews was the nuances of implementing AR/VR experiences. In doing so, participants described various challenges in debugging and testing their applications.

7. Too many unknowns in development, testing, and debugging

Since the hardware and software needed for AR and VR development are constantly evolving, participants felt that they were always dealing with "*too many unknowns*" and had to plan ahead to anticipate and deal with problems:

I think it really is the unknown unknowns…you just don't know until you start to program…when you start to create, these problems surface…[we have to] anticipate and plan for problems. (P20-Dx)

Compared to hardware available for 2D applications, rapid changes in hardware made things become obsolete more quickly in AR/VR industries. Persistent changes in AR/VR industries made it hard for creators to keep up-to-date and survive when the application might not be supported by the next generation of hardware to come:

You're working in an environment where not everyone has figured out what's possible on that particular HMD. Or, you try your best to create an AR experience for the Samsung…and it doesn't work on any other Android phone. And, the client wants it on multiple phones. So, suddenly the team faces persistent changes. (P20-Dx)

Another aspect of having persistent changes in hardware was that AR/VR creators found it difficult to locate relevant technical support. In cases where most of the contributors to AR/VR technologies are start-up companies, tools can have a short lifespan and creators end up losing support:

I own headsets that you can't get [an] SDK for any more…you spend $2,500, get on the early adopter program of something that seems to be viable and you use it for a year and then next thing you know they go bankrupt because their venture capital funding is pulled out…If they get bought up, their IP may go away and you don't have access to it anymore. (P13-H)

Participants noted several times how current development tools were not flexible in supporting diverse interactions. This sometimes forced creators to switch platforms in the middle of development as new requirements came up, which introduced even more unknowns in the creation process:

When programming all these different interfaces…maybe I want to use an Apple watch that can change the visuals instead…I would have to go in and reprogram everything to include that. It would be great if there was something that's more flexible [such that] it recognizes the device and then you can just map it to whatever... (P8-Dx)

One of the main problems is that AR change is very fast...the technology, the SDKs, the platform, the library that you use to create changes very often...I had to work with 3 different libraries, just because every time I worked in a library, it got canceled and I had to switch to a different one. (P10-PD)

The issue of dealing with unknowns made it especially difficult to debug AR/VR applications. Participants identified many variables, including the dimension of motion and the complex structure of programming with Unity, as posing many difficulties in the debugging process. For example, one domain expert explained the difficulties she faced in systematically finding the location of errors:

I don't like the debugging experience in Unity…sometimes the bug comes from Unity…like if I didn't attach some piece of code to objects in Unity. Sometimes the bug actually is in the code itself. So, the debugging becomes confusing. (P2-Dx)

In another example, one professional AR/VR creator described the physical aspects of the debugging process that remain neglected in online tutorials:

It's good to see the person doing what they say they're doing physically. Maybe all the code is correct but what you're doing with your body in VR is incorrect. And usually people don't write about that aspect. (P10-PD)

An important part of debugging AR/VR experiences involves checking the application behaviour by testing and inspecting the interactions visually. Our participants reported problems in referencing bugs that manifested visually but were hard to pinpoint in code, expressing a lack of efficient ways to control multiple, often concurrent, events without losing track:

How am I going to make 400X number of targets? Also, every single target corresponds to a different audio clip: how am I going to keep a visual reference to what that audio clip is? What happens if my files get mixed up? Essentially, I had to create a way of keeping track of what was going on and then figure out a way that I could debug these targets. (P12-Dx)

In both AR and VR implementations, participants explained how locating the originating bug can be a difficult task. For example, the environment the application is being tested and the lighting can affect the object tracking process. Moreover, in marker-less AR with new tools like *ARCore* or *ARKit* decent knowledge in programming is usually required:

It's just like it either works for me or it doesn't work and then there's no way to fix it my background does not involve any sort of computer vision (CV) and stuff. I believe there's this part in AR [that] is CV and tracking or recognition... I have no idea how those work. So those are like a black box [for me]. (P5-H)

8. User testing and evaluation challenges

As described in the barriers above, AR/VR technologies are "bleeding edge" at this stage and most of the effort is expended on getting things to work. The sheer number of barriers we identified implies that creators are busy dealing with many other issues, leaving little time for formal user testing or evaluation.

When there was interest in doing user testing, most AR/VR creators did not know how to do it properly. In particular, hobbyists and domain experts explained that they were not familiar with any usability evaluation methods, even if they wanted to improve the user experience of their applications:

I pulled up old Xerox documents on user testing and pulled up their articles, and read about what they do. I picked up some books in the library and was like, "I need to learn how to do user testing. Let's read up on user testing, and how to do this." (P12-Dx)

Even for the professional designers who were invested in user-centered design and evaluation, there were major challenges in translating the UCD guidelines to AR/VR. They often attempted to test their applications with UX methods they had learned, but ultimately most participants in this group felt that their approaches fell short. Since most end users are still unfamiliar with AR/VR technologies, participants explained how there can be a long onboarding process for them. In addition, for many types of users, their lack of familiarity with the AR/VR technologies introduced unanticipated variables that affect the output of the experience:

The moment it [VR headset] is placed on a user's head, it's one of the biggest challenges…especially if it's a new user, you're suddenly asking them to be blind and reach out and find their controllers…they see a virtual representation of it, so they have problems to grasp that connection in their minds that what they touch is the equivalent of what they're seeing virtually. (P20-Dx)

As mentioned in the prototyping section, a key challenge for authoring AR applications was designing a compelling experience with minimum distractions. While the points of distraction are expected to be gleaned from user testing, a challenge resulting from low control over experiment variables was the lack of ability to pinpoint the specific sources of distraction.

Another point of difficulty in conducting user testing was the hardware used by both test participants and developers. The constant transition between the virtual world and the debugging console caused nausea and fatigue among AR/VR creators, often leading to either prematurely releasing an application or engaging in a long iterative testing process:

	Understanding the landscape	Designing and prototyping	Implementing and testing
Professional Designers (PD)	attended local meetups (5/8), asked technical colleagues (3/8), asked questions in internal Slack (2/8)	used their prior experience/resources in 2D design (6/8)	used their prior experience in testing 2D apps (8/8), took formal courses in testing and implementation online and in person (4/8)
Domain Experts (Dx)	sought inspiration via online search (5/6), asked social contacts (4/6)	skipped this phase (3/6), mimicked similar online projects (3/6)	followed implementation-focused online tutorials and patched together code examples, but had trouble with debugging (6/6), skipped usability testing (5/6), failed to implement the project (1/6)
Hobbyists (H)	inspired by seeing interesting online videos/posts (6/7), heard from or asked social contacts (3/7)	skipped this phase (4/7), some ideation by sketching code on paper (3/7)	followed implementation-focused online tutorials and had functional apps (7/7), skipped any form of usability testing (5/7), performed QA testing (2/7)

Table 3: Summary of different AR/VR creation approaches and key activities among different groups of creators

... in almost every way it's more difficult [in VR]…you can't look at what you're experiencing in VR, and then also look at what's happening on the screen on the Unity window. And also, you have controllers, it's a two-handed experience and so you can't use your keyboard and mouse at the same time as well. (P18-PD)

From the perspective of users testing a VR application, the heaviness and warmness inside the HMDs posed additional difficulties. In some cases, VR controllers were not deemed to be representative of interactions in the real world and were confusing for users, as shown in other research [34]. With a longer onboarding process to help users pick up the new methods of interaction, the actual testing sessions tended to be time-consuming not always insightful for creators.

DISCUSSION

Our findings overall illustrate the current state of practice of AR/VR creation in our relatively diverse group of participants in terms of how they design, implement, and test AR/VR applications (summarized in Table 3). In particular, we have highlighted 8 design and implementation barriers (Table 2) that were common between our participant groups. We now reflect on the implications of our findings for future research in HCI. In particular, we discuss the importance of considering end-user developers as a growing population of AR/VR creators, how we can build learning opportunities into AR/VR tools, and the need for building AR/VR toolchains that integrate debugging and testing.

Important to consider needs of AR/VR end-user developers

A lot of the current hype for AR/VR is among professional developers who can usually access cutting-edge tools on-the-job. But, as illustrated in our findings, hobbyists, domain experts, and designers can have different needs for prototyping, programming, debugging, and testing AR/VR applications. Given that there is already a lot of momentum in HCI to better understand and support end-user developers [5,24], we consider our study to be a starting point for looking at modern AR/VR development through this lens.

Most notably, we found that domain experts and hobbyists may not even know where to start and rely on ad-hoc social recommendations to select their authoring environments. This can result in choosing a tool that, while fitting their project need, may not fit their level of experience, and even if there are no major issues in the design phase, the issues tend to be aggravated during implementation and testing.

The frequent AR/VR hardware and software updates can make end-user developers feel especially left behind and struggle to keep up. One of our participants put it as:

The industry [is] trying to solve the problem to get as many headsets in consumer's hands as possible…but at the same time, they're leaving the developers behind. (P13-H).

Several of our participants expressed a similar level of frustration and considered giving up because of the dramatic hardware or software changes they experienced and the lack of relevant expertise that they had in getting back on track. This is an important finding for future tool developers, where it would be worthwhile to consider techniques such as progressive enhancement from web development (also suggested in [46]), to help users manage these transitions.

Building learning opportunities into AR/VR tools

Our results show that AR/VR creators used two main classes of authoring tools. The most prominent category consisted of professional, feature-rich frameworks, such as Unity, which was originally designed as a game engine and only recently grew into a popular platform for AR/VR. Since these tools are more established, there is often a larger community of AR/VR creators to provide support and examples for learning [29,36]. However, the large feature set poses issues with tool explorability and has a steep learning curve. The second class of tools was more targeted at AR/VR development, but consisted of tools created in start-ups (e.g., *Torch*), or tools developed in research (e.g., *Argon.js*). Our participants found these tools were often less refined and had a relatively smaller user community, with fewer accompanying examples and more limited support.

In light of the authoring-related issues described by AR/VR creators, we discuss potential avenues for HCI research.

Supporting early-to-middle-stage AR/VR prototyping

Some current work is already exploring methods for lowering the barrier to entry in AR/VR development. For example, Torch tries to provide a code-free experience for designers such that they can quickly prototype their ideas. However, such tools may, in fact, be too high-level and abstract away all the design and development challenges. This can lock creators into the tool and make it hard to

transition to more powerful platforms such as Unity, which they will ultimately need when going beyond the prototyping stage. One approach could be to adapt the principles from emerging prototyping tools, such as ProtoAR [38] that use Play-Doh props as 3D model stand-ins or 360proto [37] for new paper prototyping templates, and integrate them with advanced tools like Unity as a way of supporting early-to middle-stage prototyping even in developer tools.

Personalizing AR/VR authoring tools based on expertise

Our hope for future authoring tools is that they can find a better match between expressivity and learnability—end user developers in AR/VR can benefit from starting with a simple development environment but with the opportunity to learn the more advanced concepts directly inside the tool. One way to do this could be to draw upon the adaptive interfaces literature to tailor feature-rich interfaces of complex authoring environments according to users' expertise level [4,14]. Another direction could be to explore ways of making AR/VR authoring tools more collaborative such that novice creators could express ideas and explore interactions while more experienced developers could take the ideas through to implementation [17,33]. This could also be extended to use online and on-demand developer communities [7,16].

Integrating access to learning resources within implementation workflows

We identified several learning barriers experienced by AR/VR creators: lack of understanding and background knowledge in nomenclature, problems finding relevant tutorials, and figuring out what basic knowledge is important before jumpstarting an AR/VR creation task. Just like with the problem of constantly evolving tools, the updating rate for the tutorials and contents does not map with the update rate of the technology. This means that tutorials quickly become outdated and put the creation process at stake. Future work can draw upon learnability research for feature-rich software [18,22] to better understand and support the learnability of AR/VR authoring tools. An interesting challenge here would be the interplay between hardware and software and design of help for immersive experiences.

Building AR/VR toolchains with integrated debugging and testing facilities

A recent review of the AR/VR tool landscape [39] shows that there is a rapidly growing number of authoring tools, but only a few transition points between them. Our interviews confirmed this, highlighting many difficulties when designing for the physical aspect of immersive experiences and the need to plan for and react to users' motions. The need to constantly transition between a VR headset and the console made it especially difficult to debug and properly test applications.

This opens up the design space for new AR/VR tools where debugging and testing facilities could be an integral part of the authoring experience. Although it would be difficult and not even desirable to build a tool that fits all needs, it is worth exploring how to design transition points into authoring tools. For example, this could mean that AR/VR creators could move from a transition point focused on prototyping, to different ones focused on implementation and debugging, to again different ones focused on testing. Future work could also explore more interactive debugging tools like the *WhyLine* [23] and investigate how they can be extended in these virtual environments to help people locate bugs and discern why their applications are not behaving as intended.

A lot of promising work in HCI is already considering testing and evaluation issues for AR/VR. For example, Dey et. al's comprehensive review of ten years of AR usability [11] reported 369 AR user studies. However, we found that most AR/VR creators, even in professional design teams, are not using these "more research-style" approaches. It may be worth thinking about what could be the parallel "discount usability" [40] methods for testing AR/VR applications that can help practitioners. A starting point would be to revisit and reconcile heuristics [13,35,49] proposed in prior research for evaluating specific AR and VR applications.

Lastly, even when creators had user tests set up, they often struggled to get experienced AR/VR test participants. Although some participants had experience with a certain AR/VR headset, that experience did not always transfer to a different device. It would be worth exploring emulator designs that can help with parallel testing and level the playing field in AR/VR creation.

Limitations

One limitation of our study is that it presents perspectives of AR/VR creators from North America only. Given the qualitative characteristic of our study, there should be some caution used in generalizing the findings. Future research can complement the insights from this study with large-scale surveys or other approaches that include more geographically diverse groups of AR/VR creators.

CONCLUSIONS

We have presented insights from a study of 21 AR/VR creators of different backgrounds using today's authoring environments. The 8 barriers we identified present a number of opportunities for the HCI community to make AR/VR authoring more user-centered and to support emerging groups of end-user developers. Our long-term vision is to broaden participation in AR/VR authoring so that end-user developers can solve domain-specific problems and create more compelling and meaningful user experiences. Overall, there needs to be more research into understanding the needs of different types of consumers trying to get started with AR/VR development and better support their authoring experiences.

ACKNOWLEDGMENTS

We thank the Natural Sciences and Engineering Research Council of Canada (NSERC) for funding this research.

REFERENCES

[1] Mark Billinghurst, Adrian Clark, and Gun Lee. 2015. A Survey of Augmented Reality. *Foundations and*

Trends® in Human–Computer Interaction 8, 2–3: 73–272. https://doi.org/10.1561/1100000049

[2] Barry Boehm, Bradford Clark, Ellis Horowitz, Chris Westland, Ray Madachy, and Richard Selby. 1995. Cost models for future software life cycle processes: COCOMO 2.0. *Annals of Software Engineering* 1, 1: 57–94. https://doi.org/10.1007/BF02249046

[3] John Bucher. 2017. *Storytelling for Virtual Reality: Methods and Principles for Crafting Immersive Narratives*. Routledge, New York and London : Routledge, Taylor & Francis Group, 2018. https://doi.org/10.4324/9781315210308

[4] Andrea Bunt, Cristina Conati, and Joanna McGrenere. 2004. What role can adaptive support play in an adaptable system? In *Proceedings of the 9th international conference on Intelligent user interface - IUI '04*, 117. https://doi.org/10.1145/964442.964465

[5] Margaret Burnett, Curtis Cook, and Gregg Rothermel. 2004. End-user software engineering. *Communications of the ACM* 47, 9: 53. https://doi.org/10.1145/1015864.1015889

[6] John M. Carroll. 1990. *The Nurnberg Funnel: Designing Minimalist Instruction for Practical Computer Skill*. MIT Press. Retrieved from https://books.google.ca/books?id=lKcmAAAAMAAJ

[7] Yan Chen, Sang Won Lee, Yin Xie, YiWei Yang, Walter S. Lasecki, and Steve Oney. 2017. Codeon: On-Demand Software Development Assistance. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, 6220–6231.

[8] Parmit K. Chilana, Celena Alcock, Shruti Dembla, Anson Ho, Ada Hurst, Brett Armstrong, and Philip J. Guo. 2015. Perceptions of non-CS majors in intro programming: The rise of the conversational programmer. In *2015 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, 251–259. https://doi.org/10.1109/VLHCC.2015.7357224

[9] Juliet M. Corbin and Anselm Strauss. 1990. Grounded theory research: Procedures, canons, and evaluative criteria. *Qualitative Sociology* 13, 1: 3–21. https://doi.org/10.1007/BF00988593

[10] Sarah D'Angelo and Andrew Begel. 2017. Improving Communication Between Pair Programmers Using Shared Gaze Awareness. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems - CHI '17*, 6245–6290. https://doi.org/10.1145/3025453.3025573

[11] Arindam Dey, Mark Billinghurst, Robert W. Lindeman, and J. Edward Swan. 2018. A Systematic Review of 10 Years of Augmented Reality Usability Studies: 2005 to 2014. *Frontiers in Robotics and AI* 5: 37. https://doi.org/10.3389/frobt.2018.00037

[12] Brian Dorn and Mark Guzdial. 2010. Learning on the job: characterizing the programming knowledge and learning strategies of web designers. In *Proceedings of the 28th international conference on Human factors in computing systems - CHI '10*, 703. https://doi.org/10.1145/1753326.1753430

[13] Tristan C. Endsley, Kelly A. Sprehn, Ryan M. Brill, Kimberly J. Ryan, Emily C. Vincent, and James M. Martin. 2017. Augmented Reality Design Heuristics: Designing for Dynamic Interactions. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting* 61, 1: 2100–2104. https://doi.org/10.1177/1541931213602007

[14] Leah Findlater and Joanna McGrenere. 2010. Beyond performance: Feature awareness in personalized interfaces. *International Journal of Human-Computer Studies* 68, 3: 121–137. https://doi.org/10.1016/j.ijhcs.2009.10.002

[15] Maribeth Gandy and Blair MacIntyre. 2014. Designer's augmented reality toolkit, ten years later: implications for new media authoring tools. In *Proceedings of the 27th annual ACM symposium on User interface software and technology - UIST '14*, 627–636. https://doi.org/10.1145/2642918.2647369

[16] Max Goldman, Greg Little, and Robert C. Miller. 2011. Collabode: collaborative coding in the browser. In *Proceedings of the 4th international workshop on Cooperative and human aspects of software engineering*, 65–68.

[17] Saul Greenberg and Chester Fitchett. 2001. Phidgets: easy development of physical interfaces through physical widgets. In *Proceedings of the 14th annual ACM symposium on User interface software and technology - UIST '01*, 209. https://doi.org/10.1145/502348.502388

[18] Tovi Grossman, George Fitzmaurice, and Ramtin Attar. 2009. A survey of software learnability: metrics, methodologies and guidelines. In *Proceedings of the 27th international conference on Human factors in computing systems - CHI 09*, 649. https://doi.org/10.1145/1518701.1518803

[19] Mark Guzdial. 2015. Learner-Centered Design of Computing Education: Research on Computing for Everyone. *Synthesis Lectures on Human-Centered Informatics* 8, 6: 1–165. https://doi.org/10.2200/S00684ED1V01Y201511HCI033

[20] Taejin Ha, Woontack Woo, Youngho Lee, Junhun Lee, Jeha Ryu, Hankyun Choi, and Kwanheng Lee. 2010. ARtalet: Tangible User Interface Based Immersive Augmented Reality Authoring Tool for Digilog Book. In *2010 International Symposium on Ubiquitous Virtual Reality*, 40–43. https://doi.org/10.1109/ISUVR.2010.20

[21] 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)*, 85–94. https://doi.org/10.1109/IWAR.1999.803809

[22] Kimia Kiani, George Cui, Andrea Bunt, Joanna McGrenere, and Parmit K. Chilana. 2019. Beyond "One-Size-Fits-All": Understanding the Diversity in How Software Newcomers Discover and Make Use of Help Resources. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems - CHI '19*, 1–14. https://doi.org/10.1145/3290605.3300570

[23] Amy J. Ko and Brad A. Myers. 2004. Designing the whyline: a debugging interface for asking questions about program behavior. In *Proceedings of the 2004 conference on Human factors in computing systems - CHI '04*, 151–158. https://doi.org/10.1145/985692.985712

[24] Amy J. Ko, Brad A. Myers, Mary Beth Rosson, Gregg Rothermel, Mary Shaw, Susan Wiedenbeck, Robin Abraham, Laura Beckwith, Alan Blackwell, Margaret Burnett, Martin Erwig, Chris Scaffidi, Joseph Lawrance, and Henry Lieberman. 2011. The state of the art in end-user software engineering. *ACM Computing Surveys* 43, 3: 1–44. https://doi.org/10.1145/1922649.1922658

[25] 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 - CHI '18*, 1–17. https://doi.org/10.1145/3173574.3173610

[26] Gun A. Lee and Gerard J. Kim. 2009. Immersive authoring of Tangible Augmented Reality content: A user study. *Journal of Visual Languages & Computing* 20, 2: 61–79. https://doi.org/10.1016/j.jvlc.2008.07.001

[27] Gun A. Lee, Gerard J. Kim, and Mark Billinghurst. 2005. Immersive authoring: What You eXperience Is What You Get (WYXIWYG). *Communications of the ACM* 48, 7: 76–81. https://doi.org/10.1145/1070838.1070840

[28] Gilly Leshed, Eben M. Haber, Tara Matthews, and Tessa Lau. 2008. CoScripter: automating & sharing how-to knowledge in the enterprise. In *Proceeding of the twenty-sixth annual CHI conference on Human factors in computing systems - CHI '08*, 1719. https://doi.org/10.1145/1357054.1357323

[29] Henry Lieberman. 2001. *Your Wish is My Command: Programming By Example*. Elsevier Science. Retrieved from https://books.google.ca/books?id=a_kdwtoIAx4C

[30] Henry Lieberman, Fabio Paternò, Markus Klann, and Volker Wulf. 2006. End-User Development: An Emerging Paradigm. In *End User Development*, Henry Lieberman, Fabio Paternò and Volker Wulf (eds.). Springer Netherlands, Dordrecht, 1–8. https://doi.org/10.1007/1-4020-5386-X_1

[31] 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 - UIST '04*, 197. https://doi.org/10.1145/1029632.1029669

[32] John Maloney, Mitchel Resnick, Natalie Rusk, Brian Silverman, and Evelyn Eastmond. 2010. The Scratch Programming Language and Environment. *ACM Transactions on Computing Education* 10, 4: 1–15. https://doi.org/10.1145/1868358.1868363

[33] Nicolai Marquardt, Robert Diaz-Marino, Sebastian Boring, and Saul Greenberg. 2011. The proximity toolkit: prototyping proxemic interactions in ubiquitous computing ecologies. In *Proceedings of the 24th annual ACM symposium on User interface software and technology - UIST '11*, 315. https://doi.org/10.1145/2047196.2047238

[34] Mark McGill, Daniel Boland, Roderick Murray-Smith, and Stephen Brewster. 2015. A Dose of Reality: Overcoming Usability Challenges in VR Head-Mounted Displays. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems - CHI '15*, 2143–2152. https://doi.org/10.1145/2702123.2702382

[35] Rabia Murtza, Stephen Monroe, and Robert J. Youmans. 2017. Heuristic Evaluation for Virtual Reality Systems. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting* 61, 1: 2067–2071. https://doi.org/10.1177/1541931213602000

[36] Brad A. Myers. 1986. Visual programming, programming by example, and program visualization: a taxonomy. In *Proceedings of the SIGCHI conference on Human factors in computing systems - CHI '86*, 59–66. https://doi.org/10.1145/22627.22349

[37] 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*, 1–13. https://doi.org/10.1145/3290605.3300826

[38] 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 - CHI '18*, 1–12. https://doi.org/10.1145/3173574.3173927

[39] 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)*, 333–337. https://doi.org/10.1109/ISMAR-Adjunct.2018.00098

[40] Jakob Nielsen. 1994. Guerrilla HCI: Using discount usability engineering to penetrate the intimidation barrier. *Cost-justifying usability*: 245–272.

[41] Ahmed Patel, Liu Na, Rodziah Latih, Christopher Wills, Zarina Shukur, and Rabia Mulla. 2010. A Study of Mashup as a Software Application Development Technique with Examples from an End-User

Programming Perspective. *Journal of Computer Science* 6, 12: 1406–1415. https://doi.org/10.3844/jcssp.2010.1406.1415

[42] Marc Rettig. 1991. Nobody reads documentation. *Communications of the ACM* 34, 7: 19–24. https://doi.org/10.1145/105783.105788

[43] John Rieman. 1996. A field study of exploratory learning strategies. *ACM Transactions on Computer-Human Interaction* 3, 3: 189–218. https://doi.org/10.1145/234526.234527

[44] Christopher Scaffidi, Mary Shaw, and Brad A. Myers. 2005. Estimating the Numbers of End Users and End User Programmers. In *2005 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC'05)*, 207–214. https://doi.org/10.1109/VLHCC.2005.34

[45] Dieter Schmalstieg, Anton Fuhrmann, Gerd Hesina, Zsolt Szalavári, L. Miguel Encarnação, Michael Gervautz, and Werner Purgathofer. 2002. The *Studierstube* Augmented Reality Project. *Presence: Teleoperators and Virtual Environments* 11, 1: 33–54. https://doi.org/10.1162/105474602317343640

[46] Maximilian Speicher, Brian D. Hall, Ao Yu, Bowen Zhang, Haihua Zhang, Janet Nebeling, and Michael Nebeling. 2018. XD-AR: Challenges and Opportunities in Cross-Device Augmented Reality Application Development. *Proceedings of the ACM on Human-Computer Interaction* 2, EICS: 1–24. https://doi.org/10.1145/3229089

[47] Maximilian Speicher and Michael Nebeling. 2018. GestureWiz: A Human-Powered Gesture Design Environment for User Interface Prototypes. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems - CHI '18*, 1–11. https://doi.org/10.1145/3173574.3173681

[48] Suzi Stephenson. 2019. TIGA Survey Reveals that Unity 3D Engine Dominates the UK Third Party Engine Market. *TIGA*. Retrieved September 20, 2019 from https://tiga.org/news/tiga-survey-reveals-that-unity-3d-engine-dominates-the-uk-third-party-engine-market

[49] Alistair Sutcliffe and Brian Gault. 2004. Heuristic evaluation of virtual reality applications. *Interacting with Computers* 16, 4: 831–849. https://doi.org/10.1016/j.intcom.2004.05.001

[50] April Y. Wang, Ryan Mitts, Philip J. Guo, and Parmit K. Chilana. 2018. Mismatch of Expectations: How Modern Learning Resources Fail Conversational Programmers. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems - CHI '18*, 1–13. https://doi.org/10.1145/3173574.3174085

[51] Jürgen Zauner, Michael Haller, Alexander Brandl, and Werner Hartman. 2003. Authoring of a mixed reality assembly instructor for hierarchical structures. In *The Second IEEE and ACM International Symposium on Mixed and Augmented Reality, 2003. Proceedings.*, 237–246. https://doi.org/10.1109/ISMAR.2003.1240707

[52] Chi Zhang and Guangzhi Zheng. 2013. Supporting adult learning: enablers, barriers, and services. In *Proceedings of the 13th annual ACM SIGITE conference on Information technology education - SIGITE '13*, 151. https://doi.org/10.1145/2512276.2512323

[53] *Augmented and Virtual Reality Survey Report, 2019*. Perkinscoie. Retrieved from https://www.perkinscoie.com/images/content/2/1/v4/218679/2019-VR-AR-Survey-Digital-v1.pdf

[54] 5 Important Augmented And Virtual Reality Trends For 2019 Everyone Should Read. Retrieved September 20, 2019 from https://www.forbes.com/sites/bernardmarr/2019/01/14/5-important-augmented-and-virtual-reality-trends-for-2019-everyone-should-read/#20e558de22e7

[55] Creative professionals are struggling to implement augmented reality – Unity Blog. Retrieved September 17, 2019 from https://blogs.unity3d.com/2019/06/18/creative-professionals-are-struggling-to-implement-augmented-reality/