Using AR Enhanced Instructional Materials to Help Novice UX Designers Get Started with XR Development

Abbey Roelofs
aheinlei@umich.edu
University of Michigan
Ann Arbor, Michigan, USA

ABSTRACT

Creation of extended reality applications has been shown to be challenging for developers across a range of experience levels and backgrounds. In this project, we explore how the known challenges relate to those faced by novice UX designers who are getting started with XR development and implement interventions to overcome these challenges. We conduct an interview session with previous graduate student instructors for a course in beginning XR development for UX students and from this identify common challenges. Based on these challenges, we implement three interventions: two AR-enhanced paper handouts and one AR sample project with demonstration video. We share these interventions with students in the XR course, use surveys to evaluate their use and perceived helpfulness for learning basic XR concepts, and discuss the benefits and challenges of these interventions, as well as possible future extensions.

1 INTRODUCTION

With the increasing availability and affordability of devices capable of producing realistic virtual- and augmented-reality experiences, the number of prospective designers interested in creating content for these devices is on the rise. This can be seen in the growing number of schools and universities offering courses in creating this content, as well as the number of youtube channels and websites dedicated to tutorials on XR content creation and troubleshooting XR projects. The latter in particular indicates that new XR creators are looking for help in the creation process, and research studies have shown that creators do face challenges when getting started with XR, regardless of their technical background. In this study, we aim to identify and categorize the challenges novice XR developers face, and to create and explore the effectiveness of two types of learning materials in overcoming these challenges: AR-augmented paper handouts, and a complete sample AR project.

One key finding of our work was that learners new to XR face a number of common challenges, which can be categorized into the areas of scoping, devices and deployment, 3D design, code and interactions, and information gaps. A second key finding was the relative value of selected interventions in addressing these challenges. Concise reference documents can be useful in addressing the uncertainties of 3D design and code and interactions, and an example project can be useful for overcoming issues of devices and deployment, scoping, and code and interactions. The value of corresponding AR enhancements, however, is less certain. The contributions of this work are 1) a learner-focused assessment of the challenges of XR development, and 2) the creation and evaluation of two potential intervention methods, AR-augmented handouts and sample projects.

Research and interventions in this study were carried out in the Introduction to AR/VR Development (SI659) course in the School of Information at the University of Michigan. Students in this course come from a broad range of backgrounds, both technical and non-technical, and have varying degrees of experience with 3D design and computer programming. Because of this, class members represent a diverse sample of novice XR developers. In addition, this course has been taught for four years by Dr. Michael Nebeling, the advisor of this project, and so has sufficient history to allow examining learner experiences in past years.

As a first step, we interviewed past teaching assistants and students in the class and analyzed their feedback to compile a list of challenges novice XR designers face and group these into recurring themes. Based on this list and recent literature on XR instructional methods, we developed three interventions and implemented them during the winter 2022 semester of the class to gather feedback. During the semester, the class had 33 participants consisting of a mix of graduate and upper-level undergraduate students.

For the first two interventions, we created AR-augmented handouts covering key concepts in the course. While the handouts themselves were traditional paper-based reference documents, we sought to determine whether adding interactive illustrations of the concepts using augmented reality could make them more effective in conveying XR concepts. We created apps to accompany the handouts, using images on the handouts as markers to open content in the app. For example, a traditional illustration of a 3D axis and a cube labeled with its coordinates on the axis brings up a 3D model of an axis and cube when viewed through the app, and the learner can adjust the cube's position to explore how its coordinates change. For the third intervention, we created an example AR project to serve as a reference and foundation for the AR development portion of the class, along with an instructional video on installing this project and deploying to a device. Our goal with this intervention was to evaluate whether novice developers benefit from seeing an example finished product, and if so, in what way they use this sample, both of which could help to inform where and how to spend effort on future learning materials for XR novices.

All materials were distributed during the lectures corresponding to their content. Both the course instructor and graduate student instructor provided input on all intervention materials before distribution to the class. We followed up each of these interventions with a survey to gather feedback on how students used the materials, how useful they found the content, and how challenging they found the topics covered in the materials.

2 BACKGROUND

With regard to these challenges, research to date has found that there are commonalities across XR developers regardless of their background. Ashtari and colleagues [1] found that developers new to creating XR experiences face a similar set of challenges whether or not they have previous experience in UX design, computer science, or 2D app development. In this study, the researchers interviewed novice developers from a variety of fields and backgrounds and identified eight common challenges that they grouped into three categories related to the stage of project development: first, getting started in AR/VR, second, designing and prototyping for AR/VR; and finally developing and testing AR/VR content. A second recent paper by Veronika Krauß and colleagues [6] also explored challenges faced by XR developers; however, this paper focused on professionals working in collaborative teams and so is less relevant to our work here.

As far as interventions to overcome these challenges, traditional paper-based learning materials continue to offer advantages–notably availability and flexibility of use–despite the existence of electronic alternatives. Recent research has shown that for some learning cases, enhancing traditional materials with augmented reality can improve learning outcomes. Radu and Schneider, for example, designed a learning experience that teaches physics concepts through a hands-on activity of building a speaker [9]. They delivered this activity to groups of participants in both AR-augmented and non-AR-augmented forms and measured the differences in learning outcomes through post-activity tests. They found that learning outcomes did improve for some concepts when delivered with the AR enhancements. For these reasons of availability of paper and potential effectiveness of AR, we decided to build on this AR-enhanced paper research.

There's also been considerable work on tools to create AR-enhanced content, for purposes ranging from education [10] to productivity improvement [7] to creative design [13]. One particularly promising tool that we considered in the implementation of our interventions here is the recent PaperTrail, by Rajaram and Nebeling [10]. They present an AR authoring system that lowers the technical barriers to creating AR instructional materials by allowing instructors to capture content for AR and attach enhancements including video recordings and clipping masks to paper materials. We ultimately decided not to use any of the existing authoring tools, as none allowed the level of customized interaction and reference to internal object properties that we wanted to illustrate here.

Finally, in spite of the work that has been done on authoring tools, it's still the case that many customized XR experiences will require some level of coding. In considering interventions here, we borrow from the well-known practice in computer programming of learning code by example. Numerous studies have examined this practice [2, 4, 8]. Notably, Ichinco and Kelleher [4] is particularly relevant to our work because it examines how novice programmers use provided examples to learn programming concepts, demonstrating that in most cases, given an example illustrating a particular concept, beginners are able to adapt this example and implement a solution in their own code.

While there is existing research on teaching the concepts of XR development, much of the literature to date is concerned with the challenges of creating such a course, mainly regarding course design and logistics: curriculum, prerequisites, assessment techniques, and the acquisition of necessary hardware [3, 5, 11, 12]. Our work here differs from this existing literature in that we focus on specific challenges novices face in learning XR concepts, rather than curricular aspects; while our work is framed in the context of a University course, the focus is not on the course itself, but rather on the learners' experience.

3 KEY CHALLENGES FOR NOVICE XR DESIGNERS

In order to identify key challenges for novice XR designers specifically in the context of an introductory AR/VR class, we held a 90-minute group interview session. The stated goal of the interview, shared with participants at the beginning, was "to identify concepts or tasks that caused common difficulty or confusion in SI659, determine how they were addressed or overcome, and identify techniques or solutions that might have made these challenges easier to overcome." The interview was designed in four main blocks. The first block was an introduction section for participants to share when and how they had been involved in the course. Second was a set of questions considering the course as whole, for example what concept or assignment came to mind as the one where students struggled the most, and what if anything they did or saw students do to overcome these struggles. The third block considered the final project for the course specifically, with questions focused on what stage of the project was the most challenging and what specific hurdles students faced in creating their projects, as well as how these were addressed. The last block was a single question for discussion: If you could change one thing about the class to make it easier for students to learn the important information, what would it be?

The interview had four participants who were past and current graduate student instructors (GSIs) for the class in question. Three of these participants were masters students at the time of their involvement in the class; one was a PhD student. Two of the masters students had also taken the class in semesters prior to acting as GSIs. While the interview was structured as described above, conversation was encouraged, allowing participants to expand on one another's ideas and identify shared experiences. We took extensive notes during the interview, which was also recorded for later reference. Following the interview, we analyzed the notes by manual thematic coding. We grouped comments by common themes, ultimately identifying five general challenge areas, which are summarized in Table [?] and described in more detail below.

3.1 Scoping

The first identified challenge area was difficulty in scoping projects for the class. Interview participants agreed that this stemmed from a general lack of understanding as to how complex or time-consuming any given idea would be to implement. Students "don't always know the limitations to know whether something can be done or not" and end up "wast[ing] too much time exploring something really cool that would be hard or impossible to do, rather than developing the user experience." A number of comments reflected this theme of students choosing to implement something that ended

Table 1: Challenge areas for novice XR developers

Challenge Type	Description
Scoping	Lack of understanding as to how complex or time-consuming an idea would be to implement
Devices &Deployment	Difficulty accessing the required devices or transferring content to the devices
3D Design	Lack of familiarity with basic concepts of computer-based 3D design
Code & Interactions	Difficulty understanding, implementing, and debugging code, especially for user interactions
Information Gaps	Lack of familiarity with or difficulty finding and using available help resources

up being more challenging than expected, often resulting in "getting caught in a technical bug rather than getting experience with designing something." Participants also suggested a number of potential solutions to this issue, such as setting more strict project guidelines to "limit the creativity to some extent", or providing some framework for the assignments by providing, "parts to use–existing objects and environment–so they can focus on the learning goal rather than going off on a tangent about other topics or details."

This was an area that was unique to our investigation here and hadn't been identified in previous literature, likely because previous investigations were not framed in the context of a specific course. Their participants had open-ended projects, whereas students in this course had to create and implement a final project on a strict deadline.

3.2 Devices and Deployment

The next identified challenge area was difficulty accessing the required devices for deployment (AR-compatible mobile devices, AR or VR headsets) and the process of transferring content to the devices. Interview participants agreed that the access challenges were largely due to the remote learning environment of the class during the COVID-19 pandemic, and that providing access to devices as early in the course as possible could overcome this issue. The problem of deploying to devices, however, was more complex. Particularly in relation to projects developed in Unity, one participant stated that "configuration is always a headache," while another agreed that "hav[ing] to manually configure so much stuff" is a distraction from the actual development process, but postponing this step leads to problems when a large portion of the project is developed before any on-device testing takes place. The main suggestion for improvement here was that "a template for the [Unity] project would have been useful." This overlapped to some extent with the challenges found by Ashtari and colleagues, although their focus was more on the unknowns related to device availability and complications involved in using the devices, rather then access and the act of deployment.

3.3 3D Design

Students in this class come from a range of backgrounds, and many have never worked in a 3D design environment before. This lack of familiarity with basic concepts of computer-based 3D design was another identified common challenge. Interview participants commented on the wide range of specialized skills involved in creating a 3D environment; doing this well requires understanding coordinates and scale, material properties, lighting design, and animation. For students without previous experience, learning the critical aspects of all these topics quickly is a major challenge, but interview participants did not mention any suggestions for how to overcome this.

While our interview participants identified these specific challenges in the development stage, Ashtari and colleagues identify a broader category of "too many unknowns in development and debugging". Interestingly, the sentiment that "nobody can be an expert in all of it" *was* echoed by participants in the Krauß study on the challenges faced by professional designers, and was given as a reason why collaborative teams are advantageous in AR/VR development, so the fact that novice developers find this to be a challenge is not surprising.

3.4 Code and Interactions

A further challenge students experienced was difficulty understanding, implementing, and debugging code, especially for user interactions with the project. Interview participants attributed this in part to a lack of prior experience with computer programming and in part to the complexity of programming XR experiences, especially when working in the Unity environment. One participant commented that "workflow and execution flow of Unity was a common problem," and that students often used code from other sources without really understanding how it should work. There was a general consensus that GSIs "are mostly working as a human debugger," as students come to office hours with code they've copied from a website and want to know why it isn't working. One comment reflected a potential way to improve this situation: "One thing that was really helpful was to take things that were already built and then rip them apart" to see how they work. These categories were also broadly identified by Ashtari and colleagues, although with more of a focus on design vs. implementation of interactions, and debugging vs. understanding of code.

3.5 Information Gaps

The final challenge was a recurring theme in discussing all of the others: in cases where students encountered any of the above challenges, their difficulties were compounded by a lack of familiarity with or difficulty finding and using available help resources. This included not knowing what specific terms to search for to find help, not being aware of (or forgetting about) documents provided by the development environment (Unity or A-Frame), and incorrectly identifying a problem, leading to a search for the wrong information. Participants had a number of suggestions for improvement in this area, including "a decision tree to help...target the resources they need, rather than endlessly searching on the internet" or "a knowledge base to share information so multiple students aren't struggling with the same issue but everyone is too shy to ask publicly." This is where we had the largest overlap with prior work: Three of the eight areas identified by Ashtari are what we consider here to be information gap; they go into more detail differentiating the types of gaps.

3.6 Summary

After reviewing the information gathered during the interview, we brainstormed potential interventions to address these challenges, identifying which challenge(s) each intervention would address. We evaluated these interventions based on their potential usefulness compared to the difficulty of implementation and ultimately decided on three interventions: two AR-enhanced reference handouts with corresponding augmented-reality applications targeted at specific lectures covering fundamental topics of XR design, and a sample Unity project demonstrating how the AR functionality of one of these apps was implemented. The first AR-enhanced handout addresses the challenge of 3D design, aiming to provide a quick, interactive way for students to grasp some of the foundational concepts. The second AR-enhanced handout further addresses the challenges of 3D design while also addressing some of the challenges of interactions, illustrating key points of basic interactions and augmented reality concepts while tying these into the 3D design concepts of the previous handout. Both of these handouts also addressed the challenge of information gaps by providing a reference for terminology and important ideas. The sample project addresses the challenge of scoping, providing a template project to give learners an idea of the amount of code and configuration required to implement some basic tasks. It also addresses the challenge of code and interactions, by providing sample code to accomplish certain common interactions. Finally, we included a video demonstrating how to install the project and deploy it to a device, in order to address the challenge of devices and deployment.

4 MAKING FUNDAMENTAL CONCEPTS MORE ACCESSIBLE

We discuss each intervention and resulting feedback below.

4.1 Intervention 1: AR-Enhanced 3D Design Handout

Based on interview feedback that students struggle with quickly grasping the fundamental concepts of 3D design, the first intervention we designed was an AR-enhanced reference handout (Appendix A) addressing some of these concepts: coordinate systems, position, rotation, and scale. For the AR portion of the intervention, we designed and built a cross-platform app using Unity and the ARFoundation package, which also addresses each of these concepts. The opening screen of the app is a menu (Figure 1a). The images on the handout act as markers, so when viewed with the app, they will bring up a corresponding interactive component; the app also functions without these markers, so learners can experiment with the concepts even if the handout isn't available.

The coordinate system image displays a 3d coordinate system with a selection of primitive objects arranged in a Unity-like hierarchy at roughly the same scale as the image itself (Figure 1b). A similar mode but at a larger, room-size scale can also be accessed without the paper marker, allowing similar exploration when the handout is not available (Figure 1c). In both modes, objects can be selected, moved, rotated, and hidden, and the position and rotation coordinates of the selected object will be shown on the bottom of the screen (Figure 1d). The user can change the coordinate system so the color-coded axes correspond to one of three popular development environments (Unity, Unreal Engine, or A-Frame) (Figure 1e), and can also switch between this position mode and a scale mode, where a scale factor can be applied to the selected object (Figure 1f).

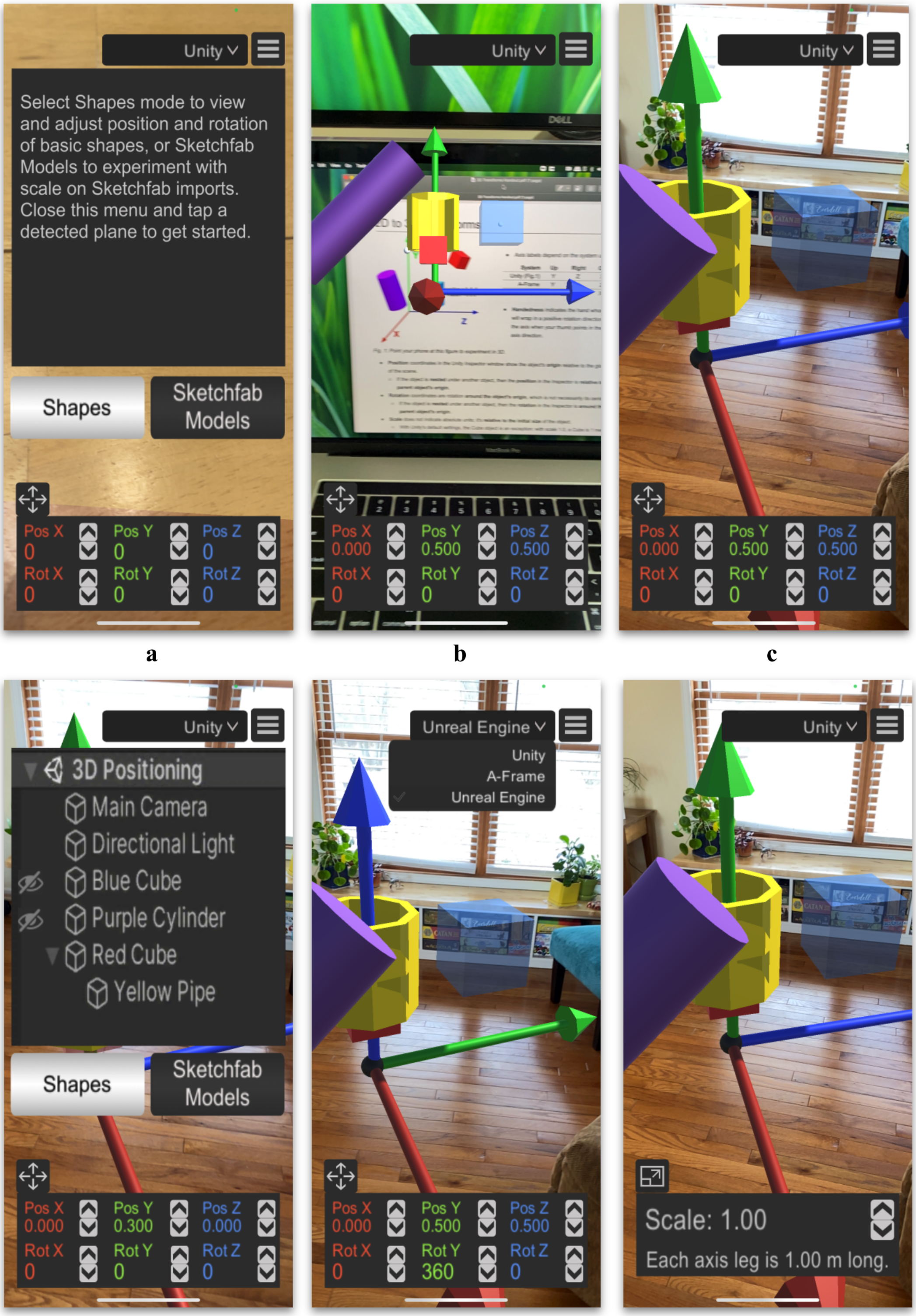
The object images at the bottom of the handout are screenshots of models downloaded from Sketchfab. These models were specifically chosen for their wide range of default sizes when imported: while the book and plant are roughly life-size, the dinosaur is extremely small, and the truck is extremely large, demonstrating two of the reasons why models might not initially be visible when imported to a scene. Viewing these images with the app allows the user to select one of the models, which will then be placed in the real world at its default imported scale, along with a 1-meter axes model for scale comparison. Controls at the bottom of the screen allow the user to adjust the scale to bring the model to a correct real-world size and see the different scale factors that might be required to do this on their own models (Figure 2).

4.2 Distribution and Feedback

We presented the 3D Transforms handout and app to students during the second lecture of the SI659 course. We included a link to both of these in the lecture slides, and we gave a short demonstration of the app functionality during the lecture. Students were encouraged but not required to refer to these materials in completing the quiz and assignment for the week's lecture, and we included two questions in the quiz that specifically referred to the materials as a potentially useful reference source. Two weeks later, following the due date of these assignments, we distributed a feedback survey to the class. This survey covered whether they used the handout and/or app, what they did or did not find useful about these materials, how challenging they found the topics and assignments of the week's lecture, and their prior experience with 3D design, programming, and the Unity, A-Frame, and Unreal Engine development environments.

Eighteen students responded to the survey; of those that responded, seventeen referred to the handout at least once, and seven referred to the app at least once. Slightly under half of the respondents who referred to the handout rated it either useful or very useful (4 or 5 on a scale of 1 to 5), and slightly under half of the respondents who used the app rated it very useful (5 on a scale of 1 to 5); see Figure 3. Student comments in the survey suggested that the most useful aspects of both the handout and app were the portions covering the different coordinate systems and scale. Students responded that "The applications helped me understand the coordinate system and the difference between platforms", and "The table with the XYZ was very straightforward! It was easy to read and I was able to retrieve the information I wanted very quickly," opinions that were echoed in many of the responses.

As far as perceived difficulty of related topics, students tended to find coordinate systems, scale, and position, which were covered

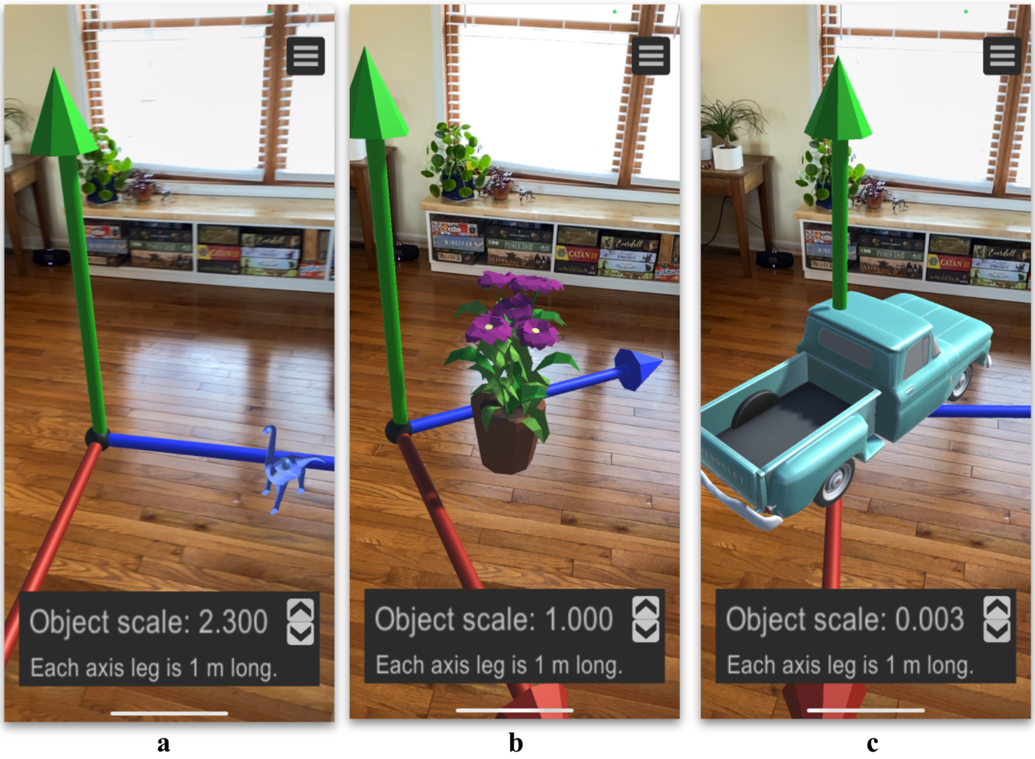


Select Shapes mode to view and adjust position and rotation of basic shapes, or Sketchfab Models to experiment with scale on Sketchfab imports. Close this menu and tap a detected plane to get started.

Scale: 1.00

Each axis leg is 1.00 m long.

Figure 1: 3D Transforms AR Enhancement functions: a) opening mode selection menu b) 3D objects corresponding to paper image c) paperless version of 3D object model d) hide and show objects e) change axis representation f) adjust model scale



Object scale: 2.300
Each axis leg is 1 m long.

Object scale: 1.000
Each axis leg is 1 m long.

Object scale: 0.003
Each axis leg is 1 m long.

Figure 2: Sketchfab model examples: a) extremely small model (note increased scale) b) approximately life-size model c) extremely large model (note decreased scale)

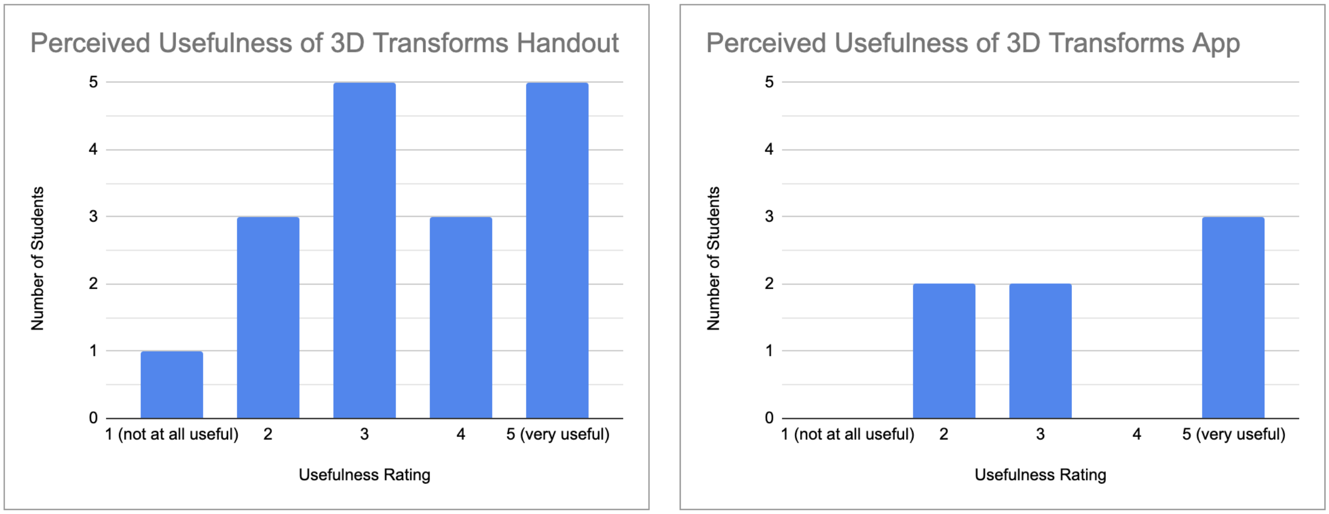


Figure 3: Student ratings of perceived usefulness of 3D Transforms handout and app

in the handout, less challenging than interactions and raycasting, which were not. They also seemed to find A-Frame slightly more challenging than Unity for creating 3D scenes, and were much less likely to use Unreal Engine (Figure 4).

4.3 Intervention 2: AR-Enhanced AR Interactions Handout

For the second intervention, we designed an additional AR-enhanced reference handout (figure X) to address the challenges of

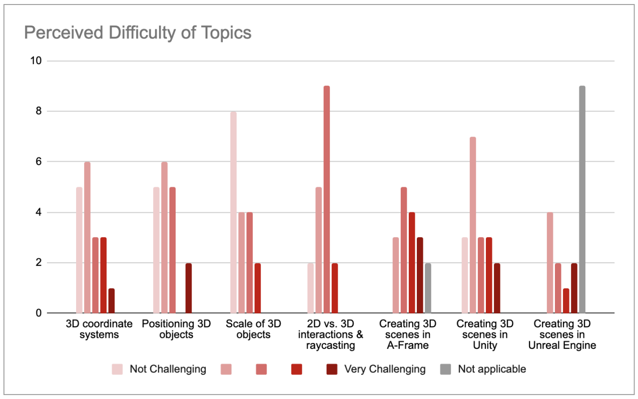


Figure 4: Students' perceived difficulty of 3D concepts

3D design, interactions, and information gaps, specifically when designing for augmented reality. For this handout, we focused on differences in the environment between marker-based and markerless AR, as well as raycasting and behavior of colliders. As in the first intervention, we designed and developed a cross-platform app using Unity and the ARFoundation package to add the AR component. We created the app to address each of the concepts listed above through a different app mode, selected via the buttons near the bottom of the display.

Since we implemented the app with ARFoundation, it is using a markerless environment; we wanted to illustrate the use of a marker image in this environment. We do this through the app's' marker mode, where the flower image on the handout acts as an image marker and demonstrates how this is different from a purely marker-based implementation. Viewing the image through the app brings up the corresponding flower object, as well as an axis model centered at the scene origin and a coordinate display showing the image's location, the camera's location, and the image tracking status (Figure 5a).

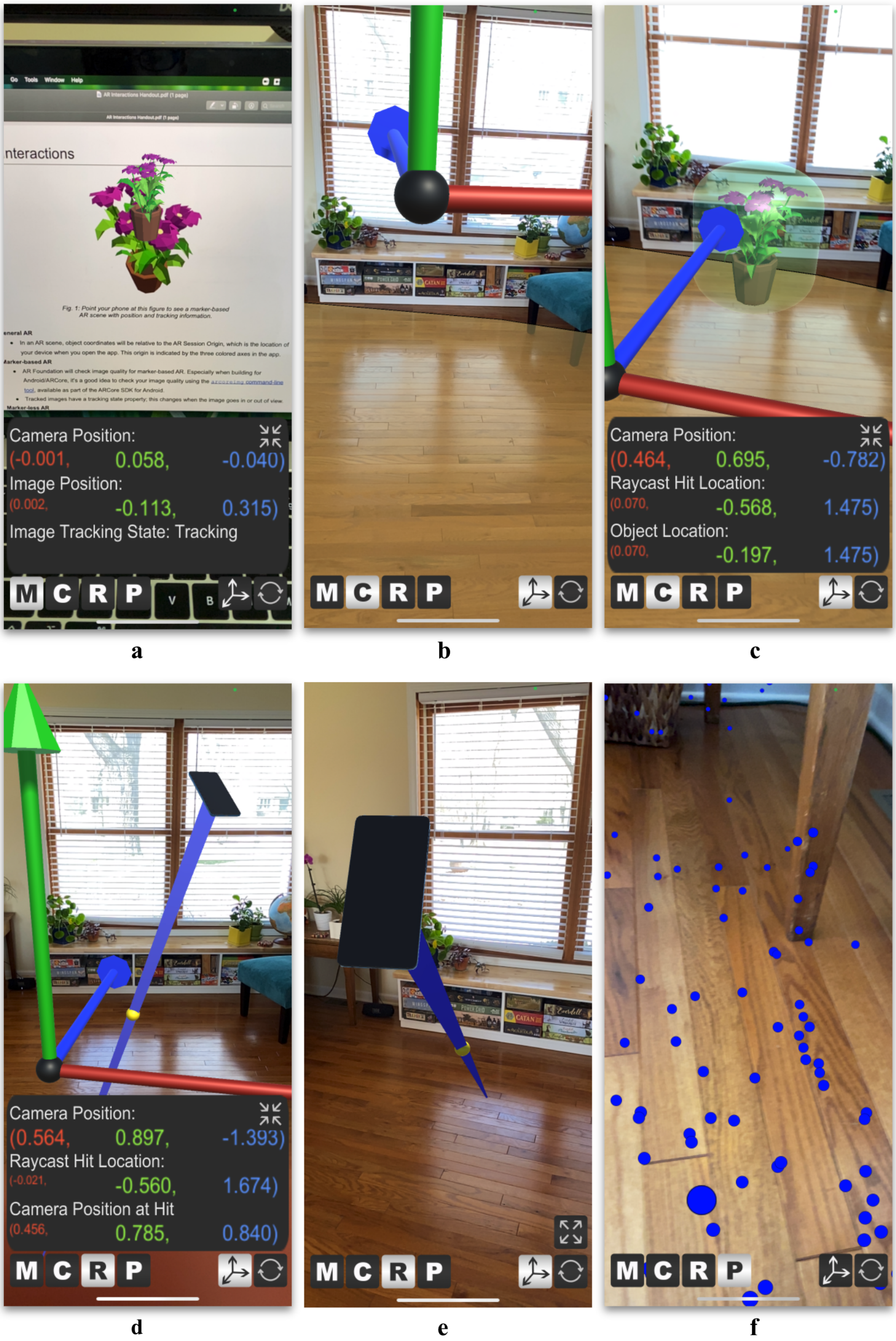
To illustrate markerless AR concepts, we created three app modes that are independent of the handout: collider mode, raycast mode, and point cloud mode. In collider mode, plane detection is enabled, and detected planes are displayed on the screen (Figure 5b). When the user taps a location on a plane, the same flower model is added to the scene two meters above the selected location, its collider envelope is shown with a translucent green material, and it falls until it lands due to a collision between the collider and the detected plane (Figure 5c). Throughout this process, the object location, raycast hit location from the user's screen tap, and camera location are shown in the coordinate section of the screen. In raycast mode, plane detection is again enabled and planes displayed. When the user taps a plane, the ray from the tap to the raycast hit location on the plane is drawn as a bright blue line. The intersection of the ray and the plane is shown with a yellow ball, and the device location at the time of the tap is shown as a model of a smartphone (Figure 5d). The user can then move around to look at how the raycast worked from different angles; the current camera (device) location as the user moves, camera location at the time of the raycast hit, and location of the raycast hit are all shown in the coordinate section. In both collider and raycast mode, the user can toggle the axes and coordinate displays and off for a clearer view of the scene (Figure 5e). Finally, in point cloud mode, the point cloud model of the user's environment is represented by a cloud of blue spheres (Figure 5f). This mode does not have any user interaction; the point cloud is automatically updated as the user moves the device. In all of these modes, the user can toggle on or off an axis model located at the world origin.

4.4 Distribution and Feedback

Distribution of the AR Interactions handout and app was similar to the first handout and app. We presented both to students during the seventh lecture (Markerless AR) of the SI659 course; we included a link to both in the lecture slides, and we gave a short demonstration of the app during the lecture. Students were encouraged but not required to refer to these materials in completing the quiz and assignment for the week's lecture, and we wrote one quiz question that specifically referred to the materials as a potentially useful reference source. Two weeks later, following the due date of these assignments, we distributed a feedback survey to the class. In this survey we asked about whether they used the handout and/or app, what they did or did not find useful about these materials, how challenging they found the topics and assignments of the week's lecture, and their prior experience with 3D design, programming, and the Unity, A-Frame, and Unreal Engine development environments. We also asked about the usefulness of the third intervention, a sample Unity project, which is discussed below. Fifteen students responded to the survey; of those that responded, fourteen referred to the handout at least once, while nine referred to the app at least once. Of the respondents who referred to the handout at least once, over half (nine) rated it useful or very useful (4 or 5 on a scale of 1 to 5), and slightly under half (four) of the respondents who used the app at least once rated it either useful or very useful; see Figure 6. While comments in the first intervention showed a strong trend in what aspects were most useful, responses on this intervention were more varied. Multiple comments mentioned plane detection: "Understood a great deal about plane detection and it was helpful to see working demos of functionalities", "I found the plane detection part useful, it helped me visualize the output." Other responses mentioned that "Seeing how the ray is cast into the environment was useful" and that the descriptions of the differences between markerless and marker-based AR were helpful.

For both AR-enhanced handouts, students were equally likely to use the app with or without the handout; for each app, only one student reporting using it both ways. For the first intervention, perceived usefulness was approximately equal for both use cases (app used with or without the handout). For the second intervention, perceived usefulness of the AR enhancement was higher for those using it with the handout than those using it in the standalone mode, reaffirming previous findings that the combination of paper and AR can be useful.

Overall, students found the topics presented here more challenging than those presented in the first survey. Marker-based AR seemed to be considered less challenging to implement overall than markerless AR, confirming that markerless AR was a useful focus for our third intervention (Figure 7).



Go Tools Window Help
AR Interactions Handout.pdf (1 page)
AR Interactions Handout.pdf (1 page)

nteractions

Fig. 1: Point your phone at this figure to see a marker-based AR scene with position and tracking information.

eneral AR

- In an AR scene, object coordinates will be relative to the AR Session Origin, which is the location of your device when you open the app. This origin is indicated by the three colored axes in the app.

arker-based AR

- AR Foundation will check image quality for marker-based AR. Especially when building for Android/ARCore, it's a good idea to check your image quality using the arcoreimg command-line tool, available as part of the ARCore SDK for Android.
- Tracked images have a tracking state property; this changes when the image goes in or out of view.

Marker-less AR

Camera Position:
(-0.001, 0.058, -0.040)
Image Position:
(0.002, -0.113, 0.315)
Image Tracking State: Tracking

Camera Position:
(0.464, 0.695, -0.782)
Raycast Hit Location:
(0.070, -0.568, 1.475)
Object Location:
(0.070, -0.197, 1.475)

Camera Position:
(0.564, 0.897, -1.393)
Raycast Hit Location:
(-0.021, -0.560, 1.674)
Camera Position at Hit
(0.456, 0.785, 0.840)

Figure 5: Sketchfab model examples: a) Marker mode: virtual representation of handout image b) Collider mode: plane detection and AR origin axis c) Collider mode: object location post collision d) AR axis hidden e) Raycast mode: displaying AR scene state at time of hit detection f) Point cloud mode: representing real-world environment detection

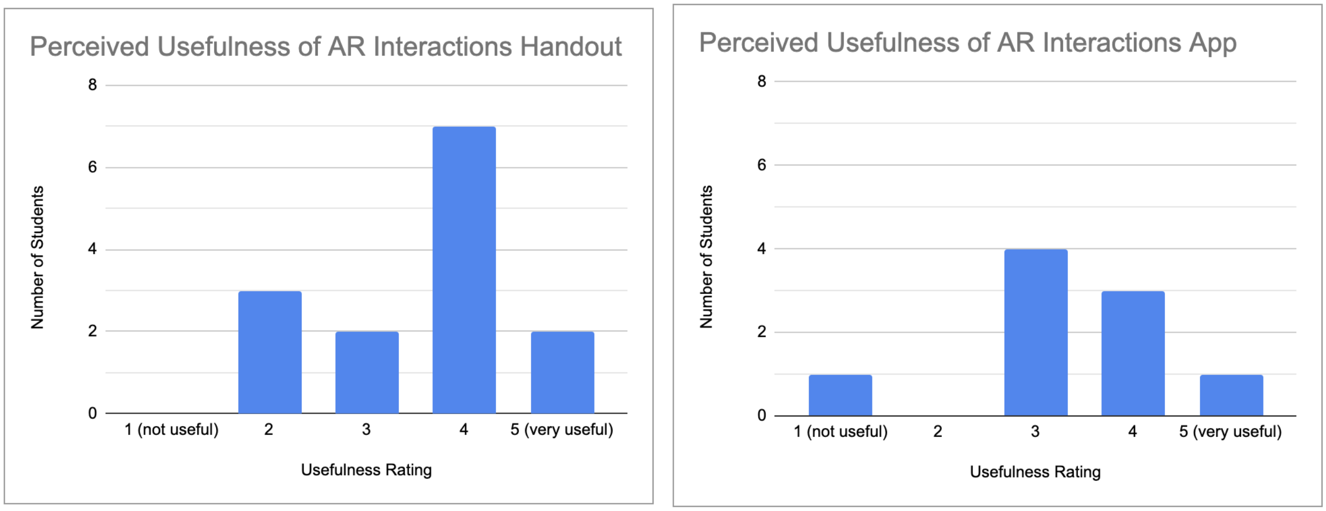


Figure 6: Student ratings of perceived usefulness of AR Interactions handout and app

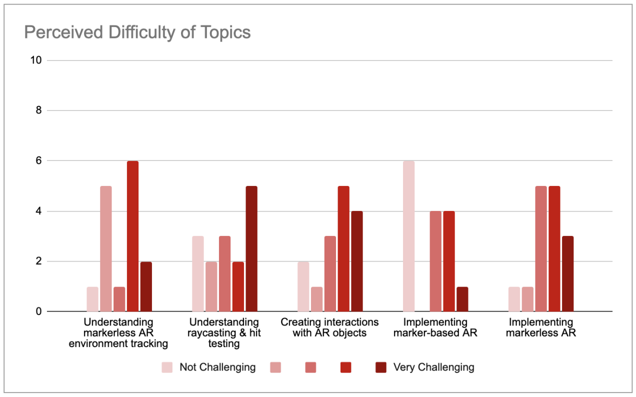


Figure 7: Students' perceived difficulty of AR topics

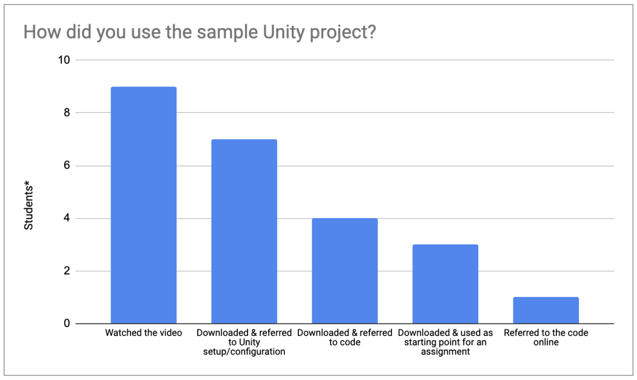


Figure 8: Student use of sample project and video

4.5 Intervention 3: Sample AR Unity Project

Since participants in our group interview session mentioned that students often used sample code they found online, and learning by example has been shown to be an effective method of learning programming [13, 14, 15], we chose to distribute an example Unity project as our third intervention. Our goal with this intervention was to address the challenges of scoping (by providing a template project), code and interactions (by providing sample code), and devices and deployment (by demonstrating the installation and deployment process). We shared the Unity project files for the AR Interaction app created above with the class via a public GitLab repository and advised that this could be used as either a template for their own project or as a reference for how to implement and code the various interactions demonstrated. We wrote a readme document for the repository containing a description of the functionality of the app, a list of the required Unity packages, and an overview of the project code, as well as a link to an installation page describing how to deploy the app directly from the build files without the Unity project. We structured the code to be as easy to understand as possible, with each major interaction type in its own dedicated file and app control code (menus, buttons, etc.) in an additional file. To address the challenge of deployment, a screen recording video of the entire process of downloading the project files to a Mac, installing the required Unity packages, adjusting the project settings, and deploying the project to an iPhone was also provided.

4.6 Distribution and Feedback

We gave students access to this project and installation/deployment video at the same time as the AR Interactions handout. In the survey on the AR Interactions handout and app, we also included a section with questions about whether and how students used the sample project, as well as its perceived usefulness.

Eleven students reported that they referred to the sample project; ten of these students found the project either useful or very useful (9). The most common modes of interaction were watching the

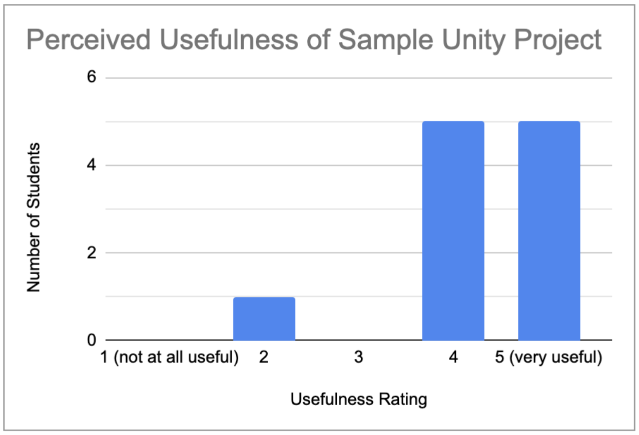


Figure 9: Student ratings of perceived usefulness of sample project and video

installation video and referring to the Unity configuration (Figure 8), which echoed the comments from the initial GSI interview that students have trouble with configuration and deployment and are seeking ways to overcome these challenges. Students found having access to the code and project itself useful, commenting that "It helped me to construct a AR project in Unity" and "I actually really enjoyed looking at the Unity project and seeing how some of the AR interactions were created. They were very different than the ones I had created."

5 DISCUSSION

Survey responses showed that all three interventions were useful to some portion of the students; however, their usefulness varied. In considering the success of each intervention, it is also worth considering the level of effort required to create the intervention relative to the proportion of the class it helped. For example, the reference handouts created for the first two interventions were quite straightforward to create and required little time or effort, and students rated them as being useful even without the corresponding apps. Because of this ratio of low instructor effort to relatively high perceived usefulness for students, we would consider these a successful intervention method that shows promise for further development: creating reference handouts to support lectures would be relatively easy for an instructor to implement and could help students key in on and quickly understand the most critical concepts of the lecture.

The corresponding AR enhancements were less used and received fewer ratings of positive usefulness from students, while also requiring considerably more effort to implement, test, and deploy. In short, the value students gained from these AR materials came at a very steep cost in terms of instructor effort. Given the promise AR-enhanced materials have shown in other areas of learning, however, this would be worth further study as tools to facilitate more streamlined creation of AR content become more developed and research continues to explore the most valuable use cases for this type of content.

While the sample project received high usefulness ratings, it also required significant effort to create, much of this effort was the overhead of creating a framework to control the various modes of the apps. By creating sample projects with separate scenes or separate projects for each concept to be illustrated, the level of effort required here could be greatly reduced. Furthermore, instructors often demonstrate how to create key concepts during class; expanding these class demonstrations into more formalized sample projects could be a way to increase the value to students while reducing the overall level of instructor effort. Installation videos would also not be necessary for every project, since the installation and deployment process is basically the same for every project; at most, additional videos to demonstrate other operating systems and device types could be useful. While creating these videos is a simple process, it does require taking the time to record the full process, as well as redoing this process whenever significant changes are made to program interfaces.

There were a number of limitations in this initial study, most notably the fact that it was focused on a single course. Because of this, the sample size of both the initial information-gathering interview session and the number of participants testing the interventions were quite small. Consequently, challenges that would be encountered in courses with a different focus or a different curricular design may have been overlooked. In addition, many students in this course have non-technical backgrounds, which likely impacted the type of difficulties they experienced, as well as the relative usefulness of the interventions. However, this does mean the usefulness of interventions should translate to novice designers with similarly diverse backgrounds, such as those described in Ashtari et al [1].

6 CONCLUSIONS AND FUTURE WORK

While the methods of AR-enhanced handouts and sample projects show some promise for overcoming student difficulties in the initial stages of learning XR development, further work is needed to optimize their usefulness. Additional investigation of common challenges with a larger sample group of students and instructors could provide a more comprehensive picture of where interventions are needed, as well as how these needs vary based on the technical background of the student population and goals of the curriculum. Consultation with experts in instructional design could improve future revisions of the handouts, and the content of these revisions could be further informed by the feedback from the additional investigation mentioned above. The apps' usefulness could likewise be improved by this additional investigation, as well as by refinements to the user interfaces based on initial feedback. Alternately, implementing different types of AR materials using existing creation tools such as PaperTrail could provide a better balance between required instructor effort and benefit to learners. Finally, sample projects and video demonstrations were helpful for the initial student group; exploring how AR enhancements could be added to learners' interactions with the sample projects themselves could provide further insights on both the use of sample projects and the benefits of AR interactions with different modes of learning.

ACKNOWLEDGMENTS

I would like to thank Dr. Michael Nebeling for making this project possible and for his support and advice over the course of the project. I would also like to thank Shwetha Rajaram, who was the graduate student instructor for SI659 during the semester in which the interventions were delivered. Her prior work on AR-enhanced instructional materials and her assistance in reviewing and supporting the materials created for this study were invaluable in completing the work.

REFERENCES

[1] Narges Ashtari, Andrea Bunt, Joanna McGrenere, Michael Nebeling, and Parmit K. Chilana. 2020. *Creating Augmented and Virtual Reality Applications: Current Practices, Challenges, and Opportunities.* Association for Computing Machinery, New York, NY, USA, 1–13. https://doi.org/10.1145/3313831.3376722

[2] Peter Brusilovsky. 2001. WebEx: Learning from Examples in a Programming Course. 124–129.

[3] Daniel C. Cliburn, James R. Miller, and Michael E. Doherty. 2010. The design and evaluation of online lesson units for teaching virtual reality to undergraduates. In *2010 IEEE Frontiers in Education Conference (FIE).* F3F–1–F3F–6. https://doi.org/10.1109/FIE.2010.5673163

[4] Michelle Ichinco and Caitlin Kelleher. 2015. Exploring novice programmer example use. In *2015 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC).* 63–71. https://doi.org/10.1109/VLHCC.2015.7357199

[5] Alexandra Klimova, Anna Bilyatdinova, and Andrey Karsakov. 2018. Existing Teaching Practices in Augmented Reality. *Procedia Computer Science* 136 (2018), 5–15. https://doi.org/10.1016/j.procs.2018.08.232 7th International Young Scientists Conference on Computational Science, YSC2018, 02-06 July2018, Heraklion, Greece.

[6] Veronika Krauß, Alexander Boden, Leif Oppermann, and René Reiners. 2021. Current Practices, Challenges, and Design Implications for Collaborative AR/VR Application Development. https://doi.org/10.1145/3411764.3445335

[7] 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* (Glasgow, Scotland Uk) *(CHI '19).* Association for Computing Machinery, New York, NY, USA, 1–14. https://doi.org/10.1145/3290605.3300917

[8] L. R. Neal. 1989. A System for Example-Based Programming. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '89).* Association for Computing Machinery, New York, NY, USA, 63–68. https://doi.org/10.1145/67449.67464

[9] Iulian Radu and Bertrand Schneider. 2019. What Can We Learn from Augmented Reality (AR)? Benefits and Drawbacks of AR for Inquiry-Based Learning of Physics. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (Glasgow, Scotland Uk) *(CHI '19).* Association for Computing Machinery, New York, NY, USA, 1–12. https://doi.org/10.1145/3290605.3300774

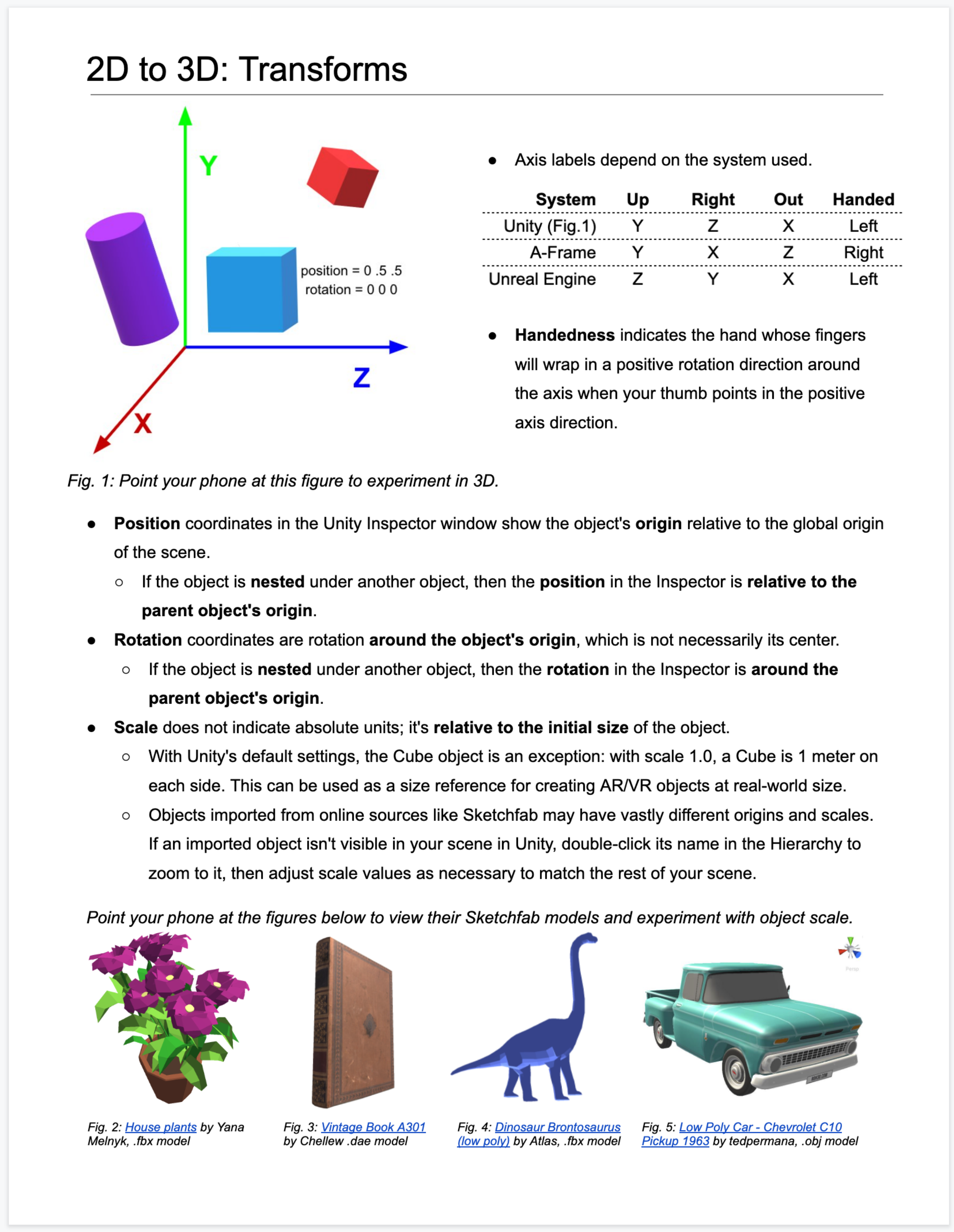
[10] Shwetha Rajaram and Michael Nebeling. 2022. PaperTrail: An Immersive Authoring System for Augmented Reality Instructional Experiences. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems* (New Orleans, LA, USA) *(CHI '22).* Association for Computing Machinery, New York, NY, USA, 16 pages. https://doi.org/10.1145/3491102.3517486

[11] Nancy Rodriguez. 2016. Teaching Virtual Reality with Affordable Technologies. In *Human-Computer Interaction. Theory, Design, Development and Practice*, Masaaki Kurosu (Ed.). Springer International Publishing, Cham, 89–97.

[12] Tuukka M. TAKALA, Lauri MALMI, Roberto PUGLIESE, and Tapio TAKALA. 2016. Empowering Students to Create Better Virtual Reality Applications: A Longitudinal Study of a VR Capstone Course. *Informatics in Education* 15, 2 (2016), 287–317. https://doi.org/10.15388/infedu.2016.15

[13] Theophanis Tsandilas, Catherine Letondal, and Wendy Mackay. 2009. Musink: Composing Music through Augmented Drawing. *Conference on Human Factors in Computing Systems - Proceedings* (04 2009), 819–828. https://doi.org/10.1145/1518701.1518827

A 3D TRANSFORMS HANDOUT



Y
Z
X
position = 0 .5 .5
rotation = 0 0 0

2D to 3D: Transforms

- Axis labels depend on the system used.

System	Up	Right	Out	Handed
Unity (Fig.1)	Y	Z	X	Left
A-Frame	Y	X	Z	Right
Unreal Engine	Z	Y	X	Left

- **Handedness** indicates the hand whose fingers will wrap in a positive rotation direction around the axis when your thumb points in the positive axis direction.

Fig. 1: Point your phone at this figure to experiment in 3D.

- **Position** coordinates in the Unity Inspector window show the object's **origin** relative to the global origin of the scene.
  - If the object is **nested** under another object, then the **position** in the Inspector is **relative to the parent object's origin**.
- **Rotation** coordinates are rotation **around the object's origin**, which is not necessarily its center.
  - If the object is **nested** under another object, then the **rotation** in the Inspector is **around the parent object's origin**.
- **Scale** does not indicate absolute units; it's **relative to the initial size** of the object.
  - With Unity's default settings, the Cube object is an exception: with scale 1.0, a Cube is 1 meter on each side. This can be used as a size reference for creating AR/VR objects at real-world size.
  - Objects imported from online sources like Sketchfab may have vastly different origins and scales. If an imported object isn't visible in your scene in Unity, double-click its name in the Hierarchy to zoom to it, then adjust scale values as necessary to match the rest of your scene.

Point your phone at the figures below to view their Sketchfab models and experiment with object scale.

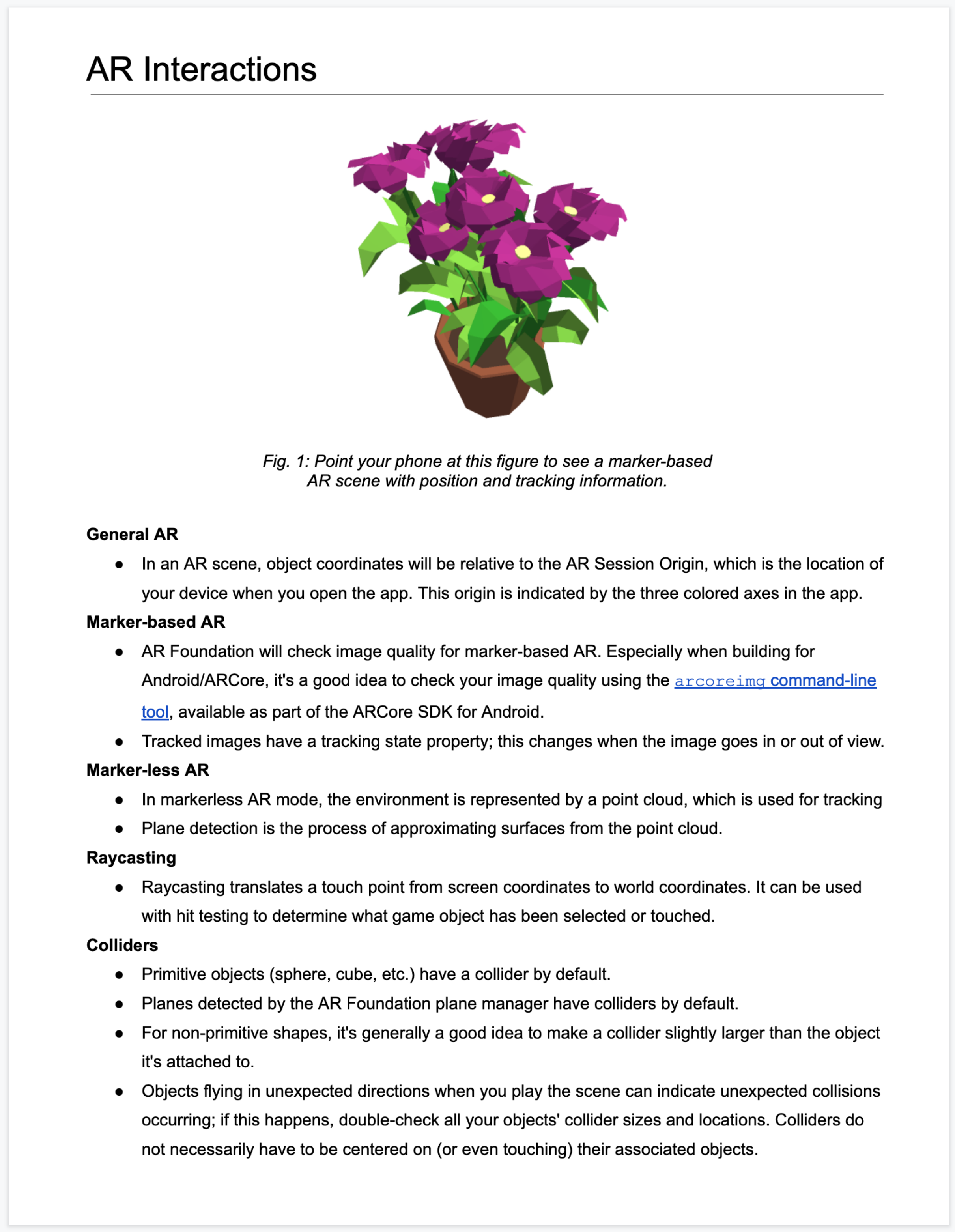
Fig. 2: House plants by Yana Melnyk, .fbx model

Fig. 3: Vintage Book A301 by Chellew .dae model

Fig. 4: Dinosaur Brontosaurus (low poly) by Atlas, .fbx model

Fig. 5: Low Poly Car - Chevrolet C10 Pickup 1963 by tedpermana, .obj model

B AR INTERACTIONS HANDOUT



AR Interactions

Fig. 1: Point your phone at this figure to see a marker-based AR scene with position and tracking information.

General AR

- In an AR scene, object coordinates will be relative to the AR Session Origin, which is the location of your device when you open the app. This origin is indicated by the three colored axes in the app.

Marker-based AR

- AR Foundation will check image quality for marker-based AR. Especially when building for Android/ARCore, it's a good idea to check your image quality using the `arcoreimg` command-line tool, available as part of the ARCore SDK for Android.
- Tracked images have a tracking state property; this changes when the image goes in or out of view.

Marker-less AR

- In markerless AR mode, the environment is represented by a point cloud, which is used for tracking
- Plane detection is the process of approximating surfaces from the point cloud.

Raycasting

- Raycasting translates a touch point from screen coordinates to world coordinates. It can be used with hit testing to determine what game object has been selected or touched.

Colliders

- Primitive objects (sphere, cube, etc.) have a collider by default.
- Planes detected by the AR Foundation plane manager have colliders by default.
- For non-primitive shapes, it's generally a good idea to make a collider slightly larger than the object it's attached to.
- Objects flying in unexpected directions when you play the scene can indicate unexpected collisions occurring; if this happens, double-check all your objects' collider sizes and locations. Colliders do not necessarily have to be centered on (or even touching) their associated objects.