Sketching in AR

Mobile App

JACKY MOOC and ANTHONY LU, University of Washington

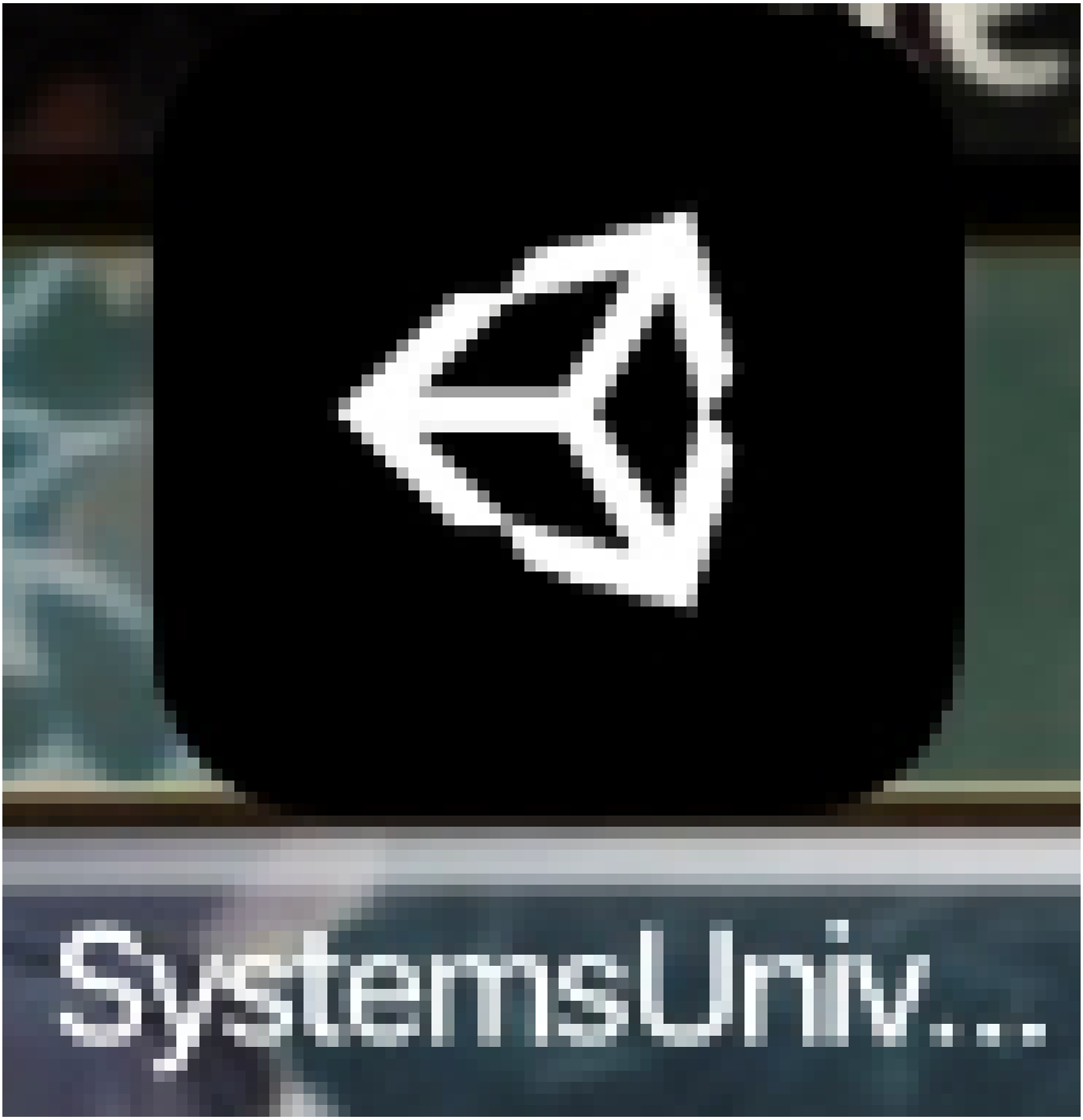

Fig. 1. The icon for our application: Systems Universal. The application can run on mobile devices and provides AR Sketching utilties.

Augmented reality gives the ability to recreate real life scenes and modify them, such as by changing the environment or adding objects. The Unity game engine has many popular tools that provide ways to create mobile apps that make use of the advantages of Augmented Reality, especially. tools such as ARFoundation and ARKit. In this project, we have create an app that supports AR drawing, and that can run on mobile devices by using techniques such as Universal Shader Graphs and Computer Vision tracking devices.

1 INTRODUCTION

Commercial products such as Clip Studio Paint and Photoshop have been popular in the years since PC's and Laptops have become commonplace. When thinking about AR and VR, a natural extension to these products would be some sort of drawing or image editing app for AR, which is what we have attempted in this project. In a sense, because the currently popular applications such as Photoshop provide 2D interfaces but still are often used to attempt to portray 3D, this extension to extended reality would likley help improve realism of these applications.

These prior applications are designed to run in laptops or PCs and use pixels, bitmaps, and raster/vector graphics. However, these use a different medium from the proposed AR drawing app, and thus would use a different set of primitives. Specifically, the implementation of AR drawing apps would focus more on detection and computer vision techniques relating to image detection, while the PC and laptop apps would focus on filters, base colors, and the math behind brushes. However, both would focus on the design of a user interface and the addition of features to allow users many options when using the application. However, many of the implementation details in extended reality would likely be dealt with by software such as Unity or Unreal Engine, and thus extended reality would most likely make use of more Computer Graphic and Computer Vision topics overall.

One existing method to sketching in AR, which is an application still in development, is the Sketch AR App. This application has the user draw on paper, and uses computer vision techniques to fix the line art and do some shading. This approach differs from our approach, as we would be using less vision/deep-learning based techniques and focus more on graphics techniques such as Universal Shader Graphs sketching in the app itself. However, many techniques such as the matrix transformations between view, model, world, and screen space, would be common because both these applications are based in AR which is 3D rather than 2D. This existing application makes use of some techniques that we do not make use of, but also has a slightly different focus which could be combined with our focus in future work.

1.1 Contributions

These were the major steps in making the mobile application.

- Choosing hardware (mobile phone over VR/AR headsets)
- Choosing software (Unity, ARKit/ARFoundation)
- Implementing the application in Unity (rendering pipeline, shader graph, adding components to the scene)

2 RELATED WORK

A prior application would be something like Adobe Photoshop or Clip Studio Paint. These softwares are designed to run in laptops or PCs and use pixels, bitmaps, and raster/vector graphics. However, these use a different medium from the proposed AR drawing app, and thus would use a different set of primitives. Specifically, the implementation of AR drawing apps would focus more on detection and computer vision techniques relating to image detection, while the PC and laptop apps would focus on filters, base colors, and the math behind brushes. However, both would focus on the design of a user interface and the addition of features to allow users many options when using the application.

Another similar application is an application (that seems to be currently improving/adding features) called Sketch AR, which is available on the Google Play Store. This application has the user draw on paper, and uses computer vision techniques to fix the line art and do some shading. This approach differs from our approach, as we would be using less vision/deep-learning based techniques and focus more on graphics techniques such as Universal Shader Graphs sketching in the app itself. However, many techniques such as the matrix transformations between view, model, world, and screen space, would be common because both these applications are based in AR which is 3D rather than 2D.

Finally, Google's ARCore framework has recently added additional features that would be useful/compliment AR Appplications. Features such as ambient lighting estimation, embedding 3d models, and augmented images (to track 2d images in real time), would be useful in a possible extension of our application.

3 METHOD

We are using universal shader graphs and device tracking to allow for drawing from the center of the screen. Most of these techniques are built in/scriptable from unity, such as a Graphics Raycaster, a Session Space (coordinate system for AR), but many parts of the application, such as the scene and components, were implemented by us in the Unity project.

Overall, by using many existing APIs, the goal is to allow users to sketch on scenes, such as in the image below. We would like to provide flexibility that is similar to those of existing, well known applications, such as paint brushes and colors, although it is unlikely that we would be able to provide all the features and nuances of the existing commercial software.

4 IMPLEMENTATION DETAILS

Put in all the specific implementation details here, including both hardware and software aspects. Even if you didn't build a piece of hardware, make sure to document what hardware you used, including your headset, computing environment, and major software libraries. If you implemented specific hardware devices, describe how you decided on the design parameters for the hardware. For example, if you built a VR headset, you'd apply the equations you previously introduced in Section 3 to decide on the values you used in your construction. If you implemented an algorithm, such as volume rendering, then you'd describe the function implementation

Authors' address: Jacky Mooc, moocjcs.washington.edu; Anthony Lu, student2@cs.washington.edu; University of Washington.

details here (e.g., GitHub projects you built on, libraries you used, or specific aspects you found challenging and how you resolved them).

The software used was Unity 3.0f6, ARKit, and ARFoundation. ARFoundation supports ARKit, and has features such as motion tracking and people occlusion, although those were not used in this project.

The Unity application was able to be built to phones (Iphone, Android), although in our project, only the Android build succeeded, as there were many steps in the IOS build process that failed/caused roadblocks. However, the android build is still relatively accessible and only requires a windows machine.

Within the unity editor, we mostly made use of the ARFoundation framework. This involved an AR Session Origin, a parent element of the AR Camera which is camera used when importing the application to a mobile device. Many features such as the AR Input Manager and RayCast Manager were also taken care of by the framework. The RayCast is used to determine where a ray intersects with a trackable object in the 3D world, and is useful for helping guide the user when sketching inside the 3D scene.

5 DISCUSSION OF BENEFITS AND LIMITATIONS

As with most applications that run on software, there can be many compatibility issues, namely between IOS and Android. While Unity allows builds for both IOS and Android, the IOS build requires access to a Mac, which is a potential drawback to the use of Unity and ARKit/ARFoundation.

The application also only supports limited sketching, such as basic colors, but does allow for drawing without necessarily needing a hard surface, due to the flexibility (3D) of AR and mobile apps.

A benefit of our approach is that by using the Universal Render Pipeline, we are able to support VR (and thus AR) platforms. The use of Shader Graph also complements Universal Render Pipeline, as Unity makes them easily compatible.

ACKNOWLEDGMENTS

We would like to acknowledge the work of Unity, ARKit, ARFoudnation, and those who build up these platforms/frameworks, as creating this application would not have been possible without the work of those who implemented these frameworks that we used.

Include a list of references (e.g., external websites and publications). See the example reports for guidance on formatting the bibliography.

Chopra, Shivang. “Using ARCore and Unity3D to Draw Lines in Augmented Reality.” Medium, Heartbeat, 11 Feb. 2020, heartbeat.fritz.ai/the-subtle-art-of-making-lines-in-augmented-reality-using-arcore-and-unity3d-e26718dffa03.

UAB, SketchAR. “SketchAR: Learn to Draw Step by Step with AR - Apps on Google Play.” Google, Google, 2020, play.google.com/store/apps/details?id=ktech.sketcharamp;hl=en$_US$.

Unity Technologies. “About Shader Graph: Shader Graph: 7.1.8.” Shader Graph | 7.1.8, 2020, docs.unity3d.com/Packages/com.unity.shadergraph@7.1/manual/index.html.

Palladino, Tommy. “Google’s ARCore Updates Bring Scene Viewer for AR on Web amp; Search, Improvements to Image Recognition amp; Ambient Lighting.” Next Reality, Next Reality, 8 May 2019, next.reality.news/news/googles-arcore-updates-bring-scene-viewer-for-ar-web-search-improvements-image-recognition-ambient-lighting-0197232/.