AR Sandbox for Construction Planning

Tech Review

Prepared for Dr. Joseph Louis
Oregon State University
School of Civil and Construction Engineering
November 2, 2018

McKenzie Gray
Group 56
Team ACE

Abstract

This document will explore possible technologies to be used as part of the AR sandbox, including the benefits and drawbacks of each. Specifically, this document will review the technologies necessary for creating AR simulations: the engine that will run the simulations, software capable of recognizing physical markers, and the markers themselves.

CONTENTS

I. Engine

The game engine is the core piece of the software side of the AR sandbox. It will be responsible for rendering the scene and simulations based on information received from the marker recognition software and depth sensor. This section will review three game engine candidates for use in the AR sandbox.

A. Unity

Unity, formerly known as Unity3D, is a massively popular game engine used to create 2D and 3D games and simulations for a wide variety of platforms. Included with Unity is an advanced physics system and an intuitive user interface which makes development on the engine simple [1].

Games created with Unity can be built for virtually every major platform. This includes native support for virtual reality (VR) and augmented reality (AR) systems, such as Oculus Rift, Steam VR, Apple ARKit, Vuforia, and others [3]. Native AR integration is an especially important consideration for this project, and Unity has some of the best support for AR.

The Unity website provides hundreds of first-party tutorials for teaching developers how to use the engine [2]. This is in addition to the large community of Unity-users providing their own guides and input. The built-in Unity Asset Store is another great resource where assets created by members of the Unity community can be purchased and downloaded straight into the Unity editor. Available for purchase are sprites and 3D models, sound effects and music, shaders and particle effects, demos and tutorials, and tools for animation, physics, UI, networking, AI, and much more [5]. All of this makes Unity a great tool for creating games and simulations quickly and easily.

B. Unreal

Epic Games' Unreal Engine is a popular, powerful game engine with support for many platforms including VR and, somewhat recently, AR. It has an extensive set of features, especially for a free engine. It touts advanced graphics capabilities, a robust multiplayer framework, and a powerful audio engine [6]. In terms of graphics, especially, Unreal is on the cutting edge, with support for advanced dynamic lighting and complex particle systems. One of the most popular features of Unreal is Blueprints, a visual scripting system that (theoretically) allows developers to script scenes without writing any code [9].

Assets can be purchased on the Unreal Marketplace, which includes models, materials, sounds, and code plugins, among other things, but it seems a bit less robust than Unity's Asset Store [8]. Epic Games provides tutorials for Unreal, as well, and these cover everything from the basics of the Unreal editor and game design to sound, lighting, modeling, and architecture [7].

Unreal is a leader for VR game creation and even includes a fully-featured VR editor, which allows developers to build scenes using a VR headset and controllers instead of a monitor and mouse [10]. In terms of AR, however, Unreal is somewhat lacking, especially in comparison to Unity. While Unreal provides a framework for AR development, the only supported systems seem to be Android, iOS, and Magic Leap [11].

C. Godot

Godot is a free and open-source, cross-platform game engine released in 2014. As a newer competitor to Unity and Unreal, it has a smaller user base, but it includes a wide array of features, thanks to the large development community.

The Godot engine is fully-featured, with everything necessary for a game engine: dynamic lighting, shaders, physics, animations, etc. It supports several platforms, including desktop, mobile, and web. In addition, Godot includes a built-in debugger and profiler, supports several scripting languages, and provides scene instancing for easier use across teams [12].

Unfortunately, Godot does not have any native AR support. Such a feature set is in the works, and AR development may still be possible with Godot, but a lack of built-in support will make it more difficult than with Unity or Unreal.

D. Conclusion

The current sandbox is already built with Unity, so Unity is the obvious choice for game engine. Additionally, while Unreal seems like a superior engine overall, with a much more impressive feature set, Unity has, by far, the best support for augmented reality than either of the other engines considered. This makes it the clear choice for the purposes of the AR sandbox.

II. Marker Recognition Software

As this project focuses on adding marker-based enhancements to the existing sandbox, we will need software that is capable of recognizing and tracking physical markers through use of a camera. This section will explore three possible AR software development kits (SDKs) that can accomplish this task. Importantly, the only SDKs considered will be those that can be built to Microsoft Windows, as that is the operating system we will be using. As such, tools like Apple's ARKit and Google's ARCore will not be considered, since they are exclusively for iOS and Android development, respectively.

A. Vuforia

Vuforia is a widely-used SDK that employs computer vision technology to detect and track targets through a camera. It is the most popular AR engine in the world, used by more than half a million developers [13]. Vuforia is free to use during development, but includes steep licensing costs upon deployment.

Vuforia's vision technology allows for dynamic recognition of both images and 3D objects. Image tracking is the simplest to employ, but requires the camera to view the image from the correct angle. An object, on the other hand, can be tracked from any angle, but Vuforia requires a digital version of the object to use as a reference. This can be done by saving either a scanned version of the object or a digital 3D model of the object into the Target Manager [14].

The Target Manager is a tool provided as part of Vuforia that allows targets to be stored within databases, either locally or in the cloud. Image targets can be sent directly to the Target Manager, and Vuforia provides tools for creating 3D targets. Object targets can be created by scanning the object using Vuforia's Object Scanner, and model targets can be created from digital 3D models using the Model Target Generator [15].

In addition to target tracking, the Vuforia Engine includes a "Virtual Buttons" feature, that allows for physical interaction with digital objects [14]. While this is not an essential part of the AR sandbox, it could be a very useful feature.

Finally, Vuforia comes with Unity integration, meaning that the Vuforia SDK can be used from within the Unity editor.

B. EasyAR

EasyAR is a multiplatform SDK developed by VisionStar Information Technology. The basic version is completely free for commercial use. Like Vuforia, EasyAR supports tracking of both images and 3D objects. However, the latter requires the paid "Pro" version [17].

EasyAR is available for multiple platforms, including Android, iOS, Windows, and OS X, and has support for Unity [18]. It is not as integrated into Unity as Vuforia, but a plugin is available to download from the Asset Store.

Documentation for EasyAR is limited and somewhat disorganized. Additionally, VisionStar is based in Shanghai and their website and documentation consist of less-than-perfect English. On the other hand, there are project samples available for Android, iOS, Windows, and Unity, which will be helpful for starting a project [19].

C. ARToolKit

ARToolKit is an open-source AR library originally developed at the Human Interface Technology Laboratory at the University of Washington. It is primarily targeted toward researchers, but has been used to create a variety of impressive AR programs.

Like the previous AR technologies, ARToolkit uses computer vision to track images. Prepared images must have a certain pattern. Specifically, markers must be square and consist of a large black border surrounding an image or design [22]. Along with image tracking, ARToolKit also allows for tracking of unprepared planar surfaces [23].

ARToolKit is available for multiple platforms, including Android, iOS, Windows, OS X, and Linux. There is also a plugin for Unity.

D. Conclusion

Vuforia is the most robust AR SDK in terms of features and is integrated into the Unity engine (which, from section I, will be our game engine of choice). Of the options considered, Vuforia seems like the most powerful and easiest-to-use AR engine. The Virtual Buttons feature further tips the scales in favor of Vuforia, and could provide very useful functionality in the AR sandbox.

III. AR Markers

This section surveys possible options for the physical markers that will be used within the AR sandbox. The markers themselves will not require any special technology, but there are several considerations to make when deciding how to design and implement them.

A. Coded Markers

QR codes and barcodes are among the simplest and most effective markers to use in any AR application. They consist of patterns of solid black and white boxes that are easy for computer vision software to identify and read. Normally, such markers contain encoded data, and therefore have somewhat complex patterns. However, AR markers need not contain any data; they could use much simpler, easier to recognize patterns and still work effectively.

B. Images

Thanks to computer vision, which is employed by each of the AR SDKs explored earlier, any arbitrary image can be used as a marker. Not all images will be effective, however. Vuforia recommends the following as attributes of a good image target:

- Rich in detail (sharp edges)
- High contrast
- No repetitive patterns

In addition, Vuforia suggests that a target should be no smaller than one tenth of the camera-to-marker distance [16].

EasyAR similarly recommends that the image be ”clear and rich-textured” [20].

Using image targets would be helpful to users of the AR sandbox as they would provide more useful context than coded markers. For example, a marker that represents a traffic light could be of an image that is reminiscent of a traffic light. On the other hand, they could potentially be more difficult for the AR engine to detect than coded markers.

C. Objects

Of the three options explored here, 3D objects are by far the most difficult to detect and track, even using advanced computer vision. Furthermore, EasyAR only offers object tracking with the paid Pro license [17], and ARToolKit does not support object tracking at all [23].

Object tracking is useful when an object must be able to interact with a digital scene, but is not necessary if the marker will simply represent a digital item in the scene, as will be the case with the AR sandbox. Objects can also be useful when a marker must be viewable from all angles. This will not be a factor in the AR sandbox, however, since the camera will be stationary and always pointed at the same location.

D. Conclusion

Images seem to be the ideal choice for use in the AR sandbox. As already explained, they would provide users with more visual context than coded markers, but should be much easier to detect and track than 3D objects. That said, it may be necessary to use coded markers, depending on how well our AR engine is able to track small images from a distance. The camera is fixed two or three feet above the sandbox, and markers will ideally be no more than a couple of inches wide. Whether or not the AR engine will be able to track such small images will need to be tested, and ”downgrading” to coded markers may be deemed necessary. In fact, it may be safest to begin with using coded markers and plan to eventually switch to image targets.

REFERENCES

[1] Unity. (2018). *Unity*. [online] Available at: https://unity3d.com/ [Accessed 2 Nov. 2018].
[2] Unity. (2018). *Unity Learn Tutorials*. [online] Available at: https://unity3d.com/learn/tutorials [Accessed 2 Nov. 2018].
[3] Unity. (2018). *Unity - Multiplatform*. [online] Available at: https://unity3d.com/unity/features/multiplatform [Accessed 2 Nov. 2018].
[4] Unity. (2018). *Partners - Vuforia*. [online] Available at: https://unity3d.com/partners/vuforia [Accessed 2 Nov. 2018].
[5] Unity Asset Store. (2018). *Unity Asset Store*. [online] Available at: https://assetstore.unity.com/ [Accessed 2 Nov. 2018].
[6] Unreal Engine. (2018). *About Unreal Engine 4*. [online] Available at: https://www.unrealengine.com/en-US/features [Accessed 2 Nov. 2018].
[7] Unreal Engine. (2018). *Unreal Academy*. [online] Available at: https://academy.unrealengine.com/ [Accessed 2 Nov. 2018].
[8] Unreal Engine. (2018). *Marketplace*. [online] Available at: https://www.unrealengine.com/marketplace/store [Accessed 2 Nov. 2018].
[9] Unreal Engine. (2018). *Blueprints Visual Scripting*. [online] Available at: https://docs.unrealengine.com/en-us/Engine/Blueprints [Accessed 2 Nov. 2018].
[10] Unreal Engine. (2018). *Unreal Engine VR Mode*. [online] Available at: https://docs.unrealengine.com/en-us/Engine/Editor/VR [Accessed 2 Nov. 2018].
[11] Unreal Engine. (2018). *Augmented Reality Development*. [online] Available at: https://docs.unrealengine.com/en-us/Platforms/AR [Accessed 2 Nov. 2018].
[12] Godot Engine. (2018). *Features*. [online] Available at: https://godotengine.org/features [Accessed 2 Nov. 2018].
[13] Vuforia. (2018). *Vuforia - Engine*. [online] Available at: https://www.vuforia.com/engine.html [Accessed 2 Nov. 2018].
[14] Vuforia. (2018). *Vuforia — AR Features*. [online] Available at: https://www.vuforia.com/content/vuforia/en/features.html [Accessed 2 Nov. 2018].
[15] Vuforia Developer Library. (2018). *Overview*. [online] Available at: https://library.vuforia.com/content/vuforia-library/en/tools/overview.html [Accessed 2 Nov. 2018].
[16] Vuforia Developer Library. (2018). *Optimizing Target Detection and Tracking Stability*. [online] Available at: https://library.vuforia.com/articles/Solution/Optimizing-Target-Detection-and-Tracking-Stability [Accessed 2 Nov. 2018].
[17] EasyAR. (2018). *Easy AR SDK*. [online] Available at: https://www.easyar.com/view/sdk.html [Accessed 2 Nov. 2018].
[18] EasyAR Developer. (2018). *Platform Requirements*. [online] Available at: https://www.easyar.com/doc_sdk/en/Getting-Started/Platform-Requirements.html [Accessed 2 Nov. 2018].
[19] EasyAR documentation. (2018). *EasyAR documentation*. [online] Available at: https://www.easyar.com/doc/index.html [Accessed 2 Nov. 2018].
[20] EasyAR documentation. (2018). *Best practice for adding new target image*. [online] Available at: https://www.easyar.com/doc/EasyAR%20CRS/newtarget.html [Accessed 2 Nov. 2018].
[21] ARToolKit. (2018). *ARToolKit Home Page*. [online] Available at: http://www.hitl.washington.edu/artoolkit/ [Accessed 2 Nov. 2018].
[22] ARToolKit. (2018). *ARToolKit Documentation (Developing your First Application, Part 2)*. [online] Available at: http://www.hitl.washington.edu/artoolkit/documentation/devmulti.htm [Accessed 2 Nov. 2018].
[23] ARToolKit. (2018). *ARToolKit Documentation (Features List)*. [online] Available at: http://www.hitl.washington.edu/artoolkit/documentation/features.htm [Accessed 2 Nov. 2018].
[24] GitHub. (2018). *ARToolKit for Unity v5.x*. [online] Available at: https://github.com/artoolkit/arunity5 [Accessed 2 Nov. 2018].