BALL TOSS IN AUGMENTED REALITY

BALL TOSS IN AUGMENTED REALITY

A project report for ball toss in augmented reality submitted in partial fulfillment of the requirements for the degree of
Master of Computer Science

By

Devish Mundra
Jawaharlal Nehru Technological University Hyderabad, 2016
Bachelor of Science in Computer Science

May 2016

ABSTRACT

The project focus is to design an augmented reality game using swift and ARKit. One of the most significant and most elusive pieces of the augmented reality game is occlusion. In other words, the ability to hide virtual objects behind the real things. The idea of the game is to recreate the fun carnival game of ball toss in augmented reality with occlusion for IOS mobile devices. This means virtual objects (ball) that are behind a real object, should be occluded or hidden behind the real object. The game starts by flicking a ball at bin place at a distance in front. You need to toss the ball into the bin to implement occlusion; multiple built-in functions are used that ARKit provides. As a final product, I have developed an augmented reality game that is fun and enjoyable.

This Project Report is approved for recommendation to the Graduate Committee.

Project Advisor:

Dr. Choi Min-Hyung

TABLE OF CONTENTS

LIST OF FIGURES

1. INTRODUCTION

Augmented reality games are becoming a powerhouse in the gaming industry. Nowadays, this is not just a fantastical concept. AR games let you fight aliens, capture fantastical creatures, defend real-world kingdoms. All this is possible without expensive headsets, just an AR-enabled smartphone or console is required.

You can be a sniper in a zombie killing game or an enemy in an action hero game. But these belong to the genre of the third person shooting game, whereas the genre of my game is quite simple and addictive. This game has no age restriction and can be played all age groups in the office in your free time. You might have probably rolled up a few paper balls and tried to toss them into the bin. I know I have, that's why the simplicity of it all drew me to design this game in Augmented reality.

1.1 Problem

Augmented reality (AR) is a new way of human and computer interaction. It is a medium that allows a 2D or 3D computer graphics to be superimposed on real-time scenes. For AR to be fully accepted, real and virtual objects within the user's environment must be seamlessly merged where the real and virtual objects must interact realistically.

The occlusion problem occurs because virtual objects will always appear in front of the real world on the screen. For example, in Figure 1, a virtual statue is placed in a gallery. As the person is standing behind the statue, the experience is working because of the virtual object appears in front of the person

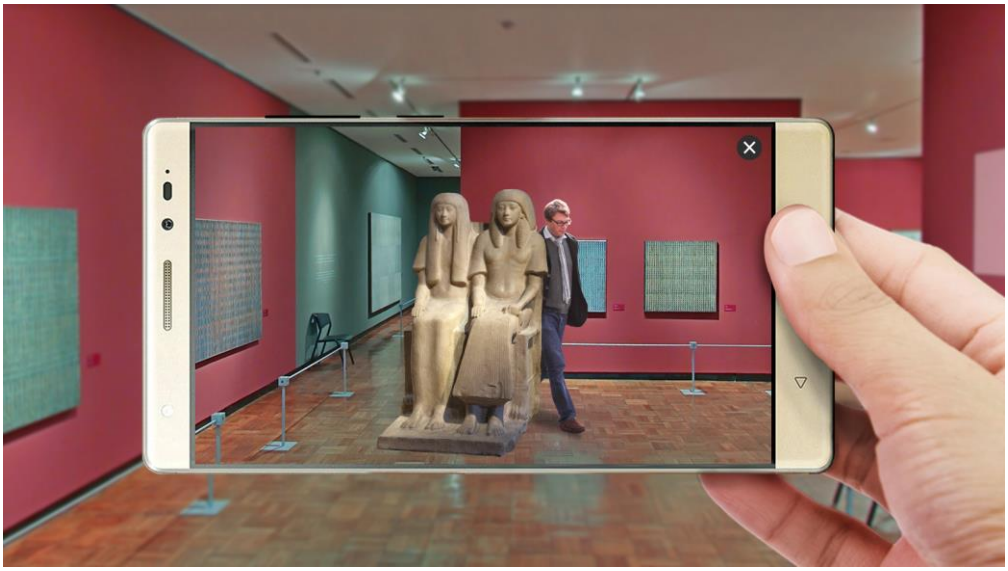

Figure 1: Virtual Statue Placed in a Gallery

However, if the person is standing between the device and the virtual statue, the immersion is broken because the statue will cover the person on the screen, which can be seen in Figure 2.

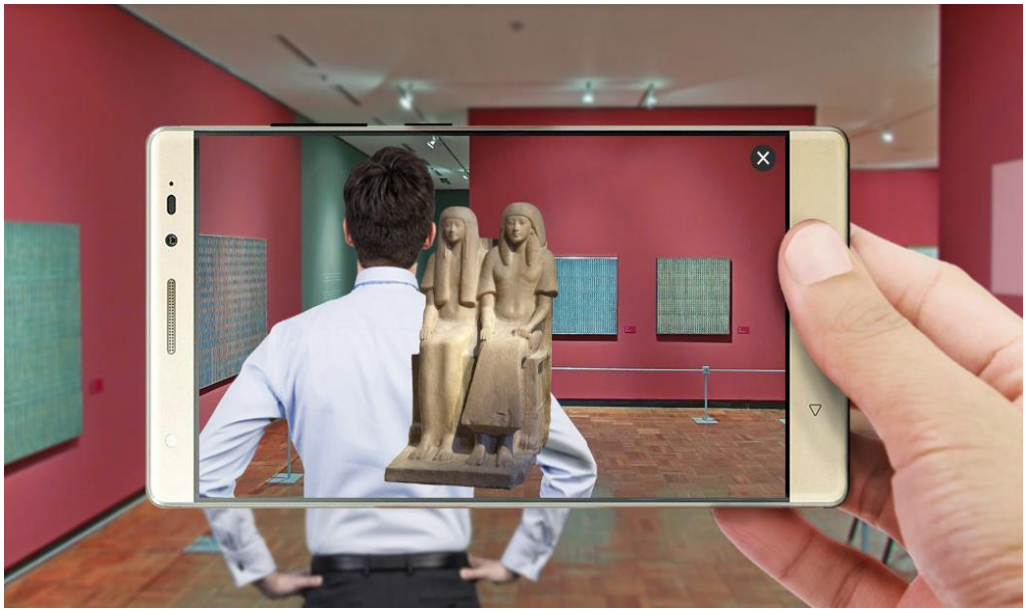

Figure 2: Person Standing Between the Device and Virtual Statue

In the AR-based game, virtual objects are overlaid in the real scene. When a real object is supposed to occlude a virtual object, the augmented image may confuse users'

perceptions. Users will have an illusion that the virtual object occludes the real object. Incorrect display on this situation may contribute to misconceptions of spatial properties by the user, improper operations of the task when trying to throw an object, thus would increase eyestrain and the probability of motion sickness [1], [2], [3]. These problems can affect the experience which should be more meaningful [4].

The goal of the occlusion is to preserve the line-of-sight rules when creating AR scenes. This means that any virtual object behind a real object should be occluded or hidden behind the real object.

1.2 Project Statement

This project aims to develop a ball toss game in augmented reality with lighthearted feel and graphics by preserving the rules of line-of-sight when creating the AR scenes. This game allows the player to play the fun carnival game ball tossing in real-world with the virtual objects (Bin & Ball)

1.3 Approach

Ball tossing is an ARKit physics-based game, and you need to throw the ball into a bin. The gameplay controls are simple. You need to use the swipe gesture to toss the ball. The distance depends on the length of the swipe and the direction of the swipe relative to the bin. But for this game to work, we need some hard-core math. We also required hardware that can track in real-world space. To achieve this, I am using apples ARKit, which provides both.

Using a process called Visual Inertial Odometry(VIO), ARKit uses motion sensors, combined with visual camera information, to track the real world [9]. Some quick math

takes this tracking information and maps in the real 3D world to the 2D screen, and we will discuss more this in chapter three of this paper.

The ARSCNView of ARKit renders the live video from the device camera as the scene background. I will use the plane detection functionality of ARKit, and then I can add virtual objects (bin) to the real world. After completing the plane detection and placing the virtual game objects, it’s time to add collision functionality and appropriate physics to the game objects. Now the ball can fall into the bin.

Occlusion in this game will be achieved by first letting the player map the vertical walls. The player needs to tell the game application about where the walls/real-world geometry is. Then the app sets up the appropriate masking planes and physics bodies. This masking of planes is then used to bounce the ball of the walls as wells as occlude when a player's aim misses the cans.

1.4 Organization of this Project Report

The paper starts with the introduction of augmented reality, followed by the core concept of the game and then the technique used to implement occlusion within the game. In the results section, I will be discussing the changes made to the game after the initial testing. Then, I will analyze the test players' feedback over time to determine the improvement of the game.

2. BACKGROUND

2.1 Key Concepts

Augmented reality is pushing the games to the next stage. Whereas virtual reality (VR) gaming takes an individual to a three-dimensional world through a headset and lets them connect with it, AR gaming incorporates the game's audio and visual elements into the user's environment, which is improved. The ability to digitally augmented surroundings in real-time has captured of both gamers and non-gamers

2.1.1 Augmented Reality

Augmented reality is tied to the idea of virtual reality (VR). VR seeks to create an artificial world that a person can perceive and navigate interactively, mainly through his or her sense of vision, but also through sound, tactile, and other types of input. AR also brings together an interactive experience but aims to complement the real world rather than create a complete artifact.

Augmented Reality attaches digital content to a live camera stream, rendering digi tal content as if it's part of the physical environment around you. Through reality, it could be anything from having your face appear like a giraffe or overlaying digital directions o n the actual streets around you. Augmented reality can let you see how furniture looks in your living room or paly a digital board game on a cereal box. All these scenarios include knowing the physical world from the camera feed, i.e., the AR device will consider where it is in the rea-world before applying the appropriate digital content to the right place. This is achieved by computer vision, which is what makes AR different from VR.

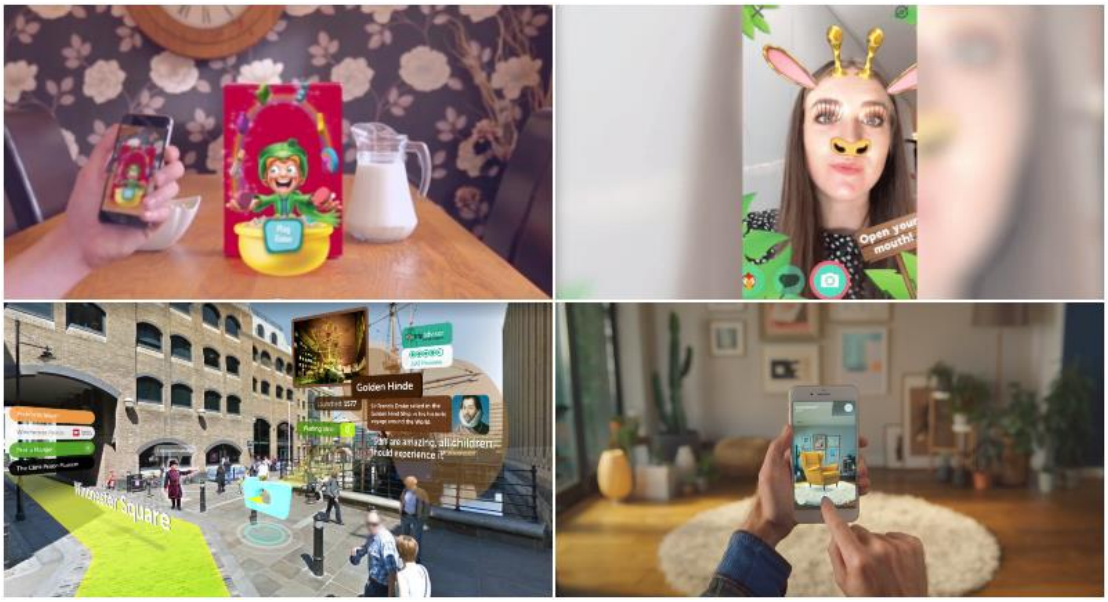

Figure 3: Augmented Reality for a digital board game, facial recognition, navigation, and furniture placement.

So now that we know the meaning of AR, how it is going to work? Next, the computer vision understands the world around the user from the content of the camera feed. This allows the display of digital content relevant to what the user is looking at. This digital content is then displayed realistically so that it appears to be part of the real world that is called rendering. Before I break this down into more detail, let's use a concrete example to make this clear. Consider playing an increased reality board game using a traditional cereal box as a physical object, as shown in the figure below. In the first instance, computer vision detects the raw picture from the camera and identifies the cereal box. This is triggering the game. The rendering module increases the original frame with the AR game, ensuring that it overlaps precisely with the cereal box. For this function, it uses the 3D location and direction of the box defined by the computer vision. Because augmented reality is live, all the above must happen every time a new frame comes from the camera.

Figure 4: Augmented Reality Board game using a cereal box as a physical object.

We need to define some logic beforehand for each augmented reality experience. This specifies which digital content should be triggered when something is recognized. The last step in the AR Pipeline is the live AR system, where the rendering module displays the relevant content on the camera feed. One way to explain how AR operates is to view computer vision as reverse rendering. Intuitively, computer vision knows and acknowledges the 3D environment form 2D picture (there's a face and where it is in the 3D environment) so that we can apply digital content (a 3D giraffe mask attached to the face) to the 2D phone screen.

2.1.2 Game Concept

The initial idea of the game is to recreate the fun carnival game of can knockdown in augmented reality with occlusion for mobile devices. This means virtual objects (cans, ball) that are behind a real object, should be occluded or hidden behind the real object. The game starts by flicking a ball at some cans arranged in front. You get one shot and a ball at first to rack up a score as the arrangement of cans progressively changes as you progress

ahead. I was able to design a level where I can place my designed crate, coke cans, and a tennis ball attached to the camera. But I was unable to implement the occlusion part for my original game due to lack of documentation for occlusion in unreal engine. I decided to go with Swift and ARKit to complete the scope of the game. With limited time in hand, I decided to change the game design instead of knocking down the cans I decided to toss a ball into a bin (Ball Toss AR).

The latest design is based on the same concept of the old model instead of knocking down the coke cans with a ball you need to throw the ball into the bin. This is an augmented reality game; therefore, you can play it wherever you like, at home, at the office, even on a bus.

The game starts by detecting the plane where you can position the virtual bin anywhere in the plane. After the bin is placed, the ARKit object detection and recognition tool identifies the real-world object and enabled occlusion for that object (a cardboard box, for instance), you can start playing. You need to use a swipe motion to toss the virtual ball. The distance depends on the length of the swipe and the direction of the swipe relative to the bin. So now the virtual object, i.e., the ball misses falling in the bin and goes behind a physical object (a cardboard box, for instance), then it will only be visible once the player moves his AR supported device behind the physical object in space.

2.1.3 Occlusion

Augmented reality in video-based displays overlays virtual objects in the real environment. In many cases, this does not represent the actual situation in the AR scene. When a real object supposes to occlude a virtual object, the augmented images may confuse users' perception. This incorrect display contributes to misconceptions and wrong operations of the task amongst users.

Occlusion problems occur because virtual objects will always appear in front of the real world on the screen. For example, adding a virtual object inside a real object (adding a virtual apple inside an actual bowl) can cause occlusion issues. The image below illustrates how the AR illusion works only for some angles. Again, the virtual object will always go on the front of the real one.

Figure 5: Augmented Reality working only for some angles

The goal of the occlusion is to preserve the line-of-sight rules when creating AR scenes. This means that any virtual object behind a real object should be occluded or hidden behind the real object.

2.2 Related Work

2.2.1 Importance of Augmented Reality

Augmented Reality (AR) is one of the main developments in technology right now and will only get bigger when AR's smartphones and other apps become more available around the globe. AR let's see the real-life environment right in front of us. Despite advances in AR technologies, these scenarios are not that far from what might already be accessible for your smartphone. Besides, augmented realism is readily available and used in a variety of ways, including Snapchat filters, in apps that help you locate your vehicle in a crowded parking lot, an AR app lets you see how a piece of furniture looks and blends into your room. [3], [4], [5]. and in several retail apps [6] that let you wear your clothes without even leaving home.

Perhaps the most famous example of AR innovation is the Pokémon Go [7] mobile app, which was introduced in 2016 and soon became an inescapable sensation. Players find and catch Pokémon characters in the game that show up in the real world on your pavement, or in a pond.

Augmented reality bridges the gap between users and game and developer of games. Its engineering has reached beyond the scope of filters and has been widely adopted by the organizations to provide the user with the ultimate gaming experience [8]. It

integrates technical features that make games highly addictive and empower designers while enabling them to improve their skills.

2.2.2 Occlusion in Augmented Reality

One of the most significant and most elusive pieces of the augmented reality game is occlusion. In other words, the ability to hide virtual objects behind the real objects. For AR to become fully accepted, real and virtual objects1must appear to be merged within the user's environment. For the augmented reality to be convincing, real, and virtual objects must interact realistically [8]. AR entities can communicate with each other in a variety of ways, which may be separated into two types, visual and physical.

ARKit uses Visual Inertial Odometry, where iOS device motion-sensing hardware data is combined with computer vision analysis of the world visible to the device camera. [9]. It then detects distinguishable features in the world, monitors differences in the positions of those key points across frames and compares that information to motion-sensing information.

3. ARCHITECTURE

3.1 High-Level Design

In this section, I will describe the approach for design the Ball Toss game in augmented reality using ARKit and swift. The game starts by detecting the plane where you use the plane surface as the base for the game and place the virtual bin anywhere in the plane on user selection. After the bin is placed, the ARKit object detection and recognition tool identifies the real-world scanned object and enabled occlusion for that object (a cardboard box, for instance), you can start playing. You need to use a swipe motion to toss the virtual ball. The distance depends on the length of the swipe and the direction of the swipe relative to the bin. So now the virtual object, i.e., the ball misses falling in the bin and goes behind a physical object (a cardboard box, for instance), then it will only be visible once the player moves his AR supported device behind the physical object in space. Figure 6 shows the architecture of the game.

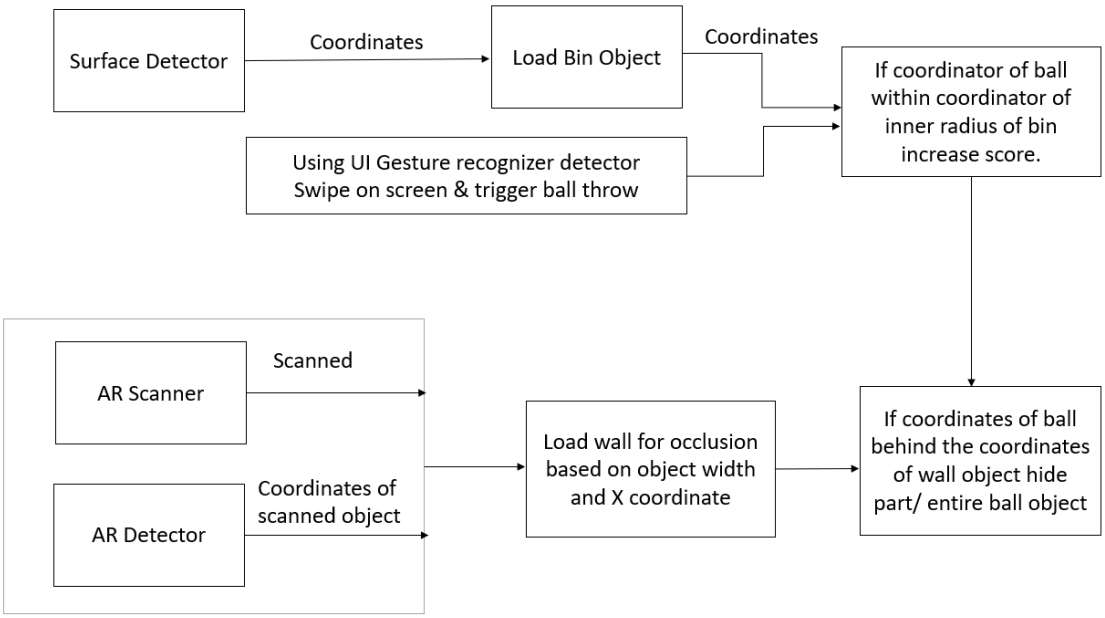

Figure 6: Architecture of the Ball Toss Game

3. 2 Implementation

To make the Ball Toss game work in augmented reality, I used Swift and ARKit. To design the game objects bin and sphere I used the default assets provided by swift. I used the basic sphere and added a mesh color to it. Then coming to the bin, I used the hoop object and converted it into a bin and added mesh color to it. I need to start with detecting the plane so that I can place my virtual bin.

Surface Detector: ARKit surface detector helps to detect the plane. Using the coordinates of the plane the virtual object bin can be placed.

AR Scanner: It is used to scan the real-world objects on the scene.

AR Detector: It is used to detect the coordinates of the scanned real-word object.

UI Gesture: Using UI Gesture recognizer detects the swipe on the screen to trigger the ball throw event. Swift provides UI Gesture recognizer.

Using AR scanner and AR detector, the real-world objects are scanned, and coordinates of the real-world objects are detected, then a wall will be loaded for occlusion based on the object width and X coordinates. Using swipe gestures when a player throws the ball towards the bin if the coordinated of the ball is within the coordinates of the inner radius of bin score will increase. If the coordinates of the ball are behind the coordinates of the wall, then ball is hidden behind the wall. Before we start the game, I have used the scanner module to make a 360-degree scan of the object under consideration and export this scanned file to use as input in the game. Once you launch the game based on the inputs file, it only looks for that particular object.

4. RESULTS AND ANALYSIS

4.1 Methodology

Game Object Design: The heart of any game is the game objects. I wanted to design the game objects for my game. So, I started developing my game objects (coke can, tennis ball, and crate). By using 3D MAX, I was able to design coke can, tennis ball, and crate.

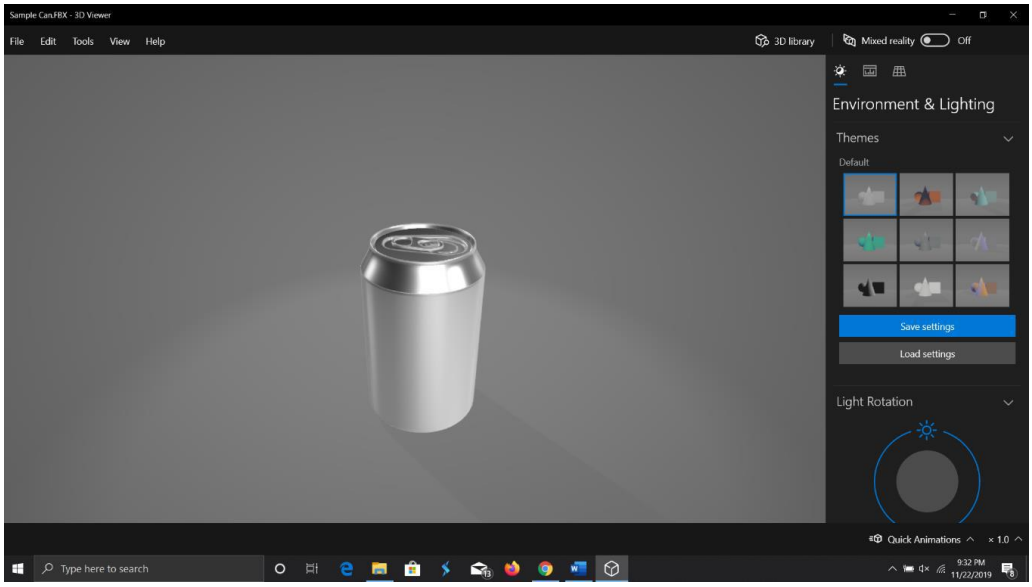

Figure 7: Coke Can 3D Model

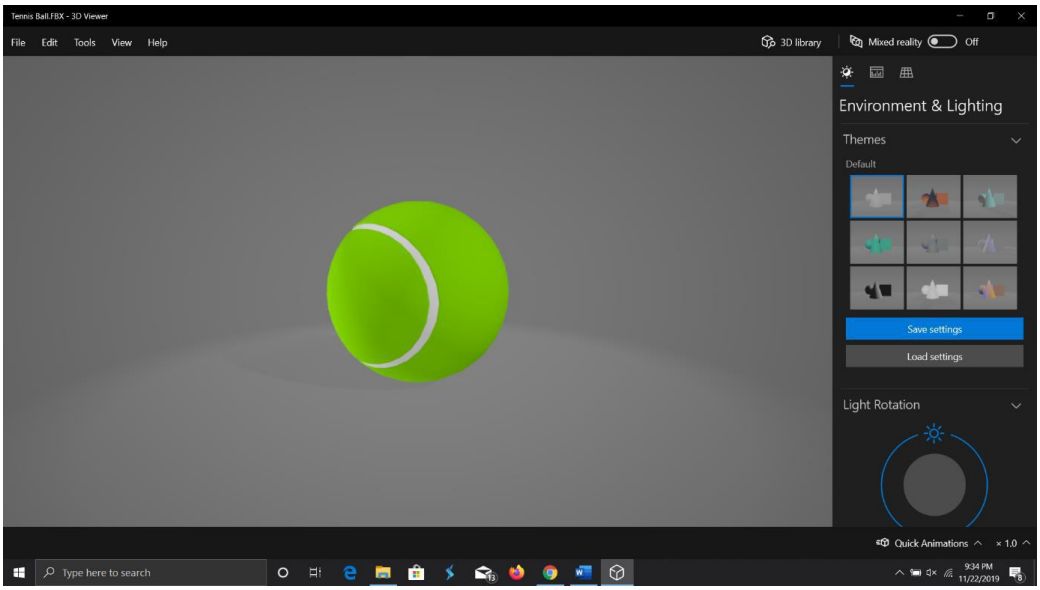

Figure 8: Tennis Ball 3D Mode

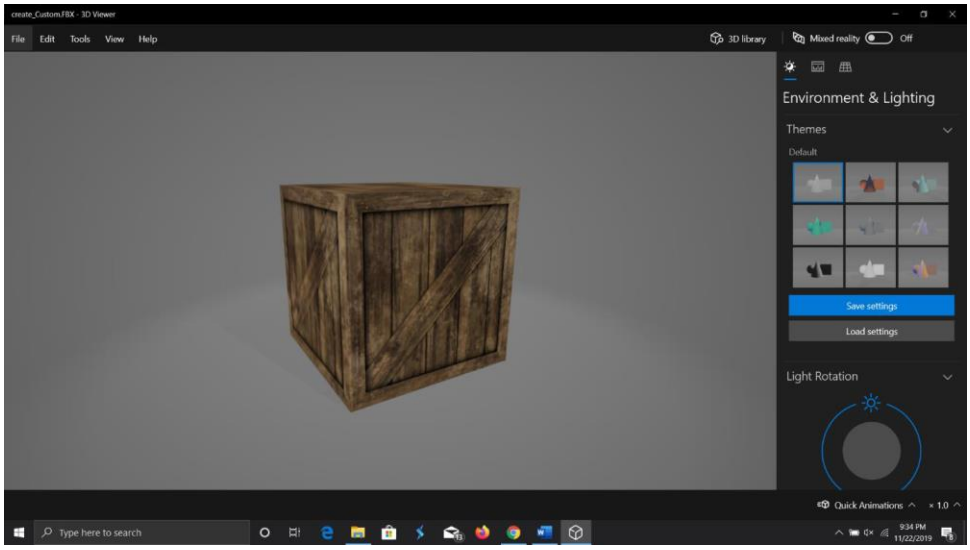

Figure 9: Wooden Crate 3D Model

Augment Reality & Plane Detection: As it is an Augmented Reality game, I need to start with detecting the plane so I could place my game object. To do this, I used Google AR Core with Unreal Engine. Here comes the challenging part. I need to test my work on an android device, the device which I have doesn't support the augmented reality. I need to buy a device that supports Augmented Reality. Then I Stated testing the plane detection.

Level Design: As per the plan, I thought I could design at least two levels. As mentioned earlier, I spend most of my time in detecting a plane and placing of my game objects. Due to lack of time, I decided to go with a single level. I was able to design a single level where I can place my designed crate, coke cans, and a tennis ball attached to the camera. Now the challenging part comes, I need to throw the ball at cans to knock them down. But the blueprint which I designed was not working, and I spent most of the time finding a solution where the

player can easily throw the ball. Finally, I found a solution. Then I was able to knock the cans down.

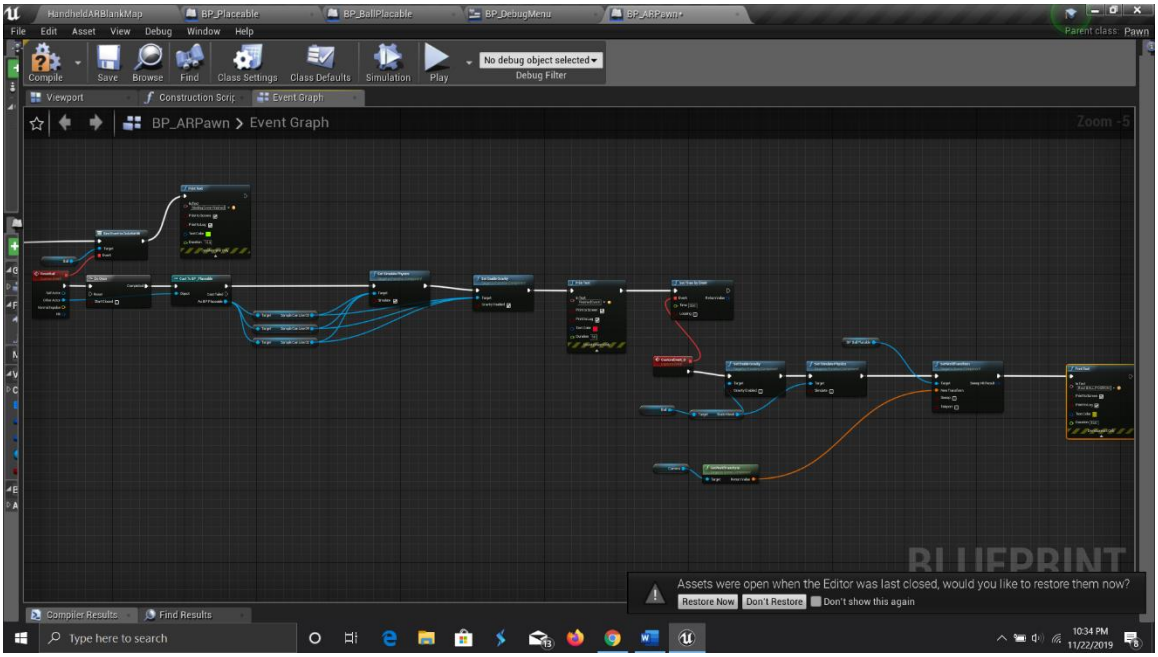

Figure 10: Blueprint for Can Knock Down

Occlusion: As there was no proper documentation for occlusion in the unreal engine. I decided to go with Swift and ARKit to complete the scope of the game. With limited time in hand, I decided to change the game design from knocking down the cans to tossing a ball into a bin. Though this was not the ideal approach, I managed to get the main core idea of the research project, i.e., occlusion.

The initial concept of the game was to make the virtual object, i.e., coke can be added in the real-world space hide behind a physical object in space, i.e., table or a chair once it has been hit by another object in space(ball). This all is done in the unreal engine using ARCore

The new concept is to toss the virtual ball in a virtual bin. I started by detecting a plane and placing the virtual bin in it. Then I used ARKit object detection and recognition tool to identify the real-world object and enabled occlusion for that object. So now the virtual object, i.e., the ball misses falling in the bin and goes behind a physical object (a cardboard box, for

instance), then it will only be visible once the player moves his AR supported device behind the physical object in space. Below are the screenshots of the development of the new approach. Where it detects a plane, place the objects and identifies the real-world object (box)

4.2 Results

I started testing my original planned game. i.e., Can knock down on an android device. I used Moto G running android 5.0 to test my game, and after loading the game on the device, I realized that the device doesn't support augmented reality. Then I looked for AR supported devices like Samsung Galaxy A5 running android 7.0 to test it

The game ran smoothly without any lags or glitches when run on Galaxy A5, the ARCore tool could detect the plane in the real-world space using the inbuilt device camera, and on this detected plane I managed to place the virtual object, i.e., coke can successfully. The following is the screenshot of the game running on Galaxy A

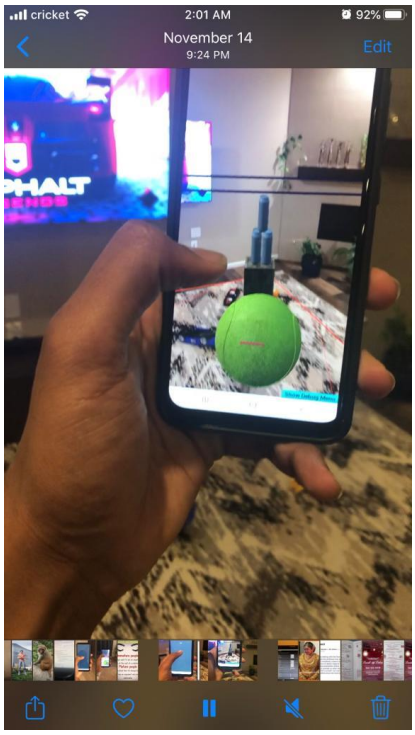

Figure 11: Can Knockdown Working in Samsung Galaxy A5

Once the virtual object i.e. ball is moved using the swipe gesture towards the Coke Can (another virtual object), it detects the Coke Can position and makes it fall on the side. This was achieved using the physics mechanism provided by the Unreal engine. The following is the screenshot of the virtual object i.e. ball and Coke can add to the real-world space.

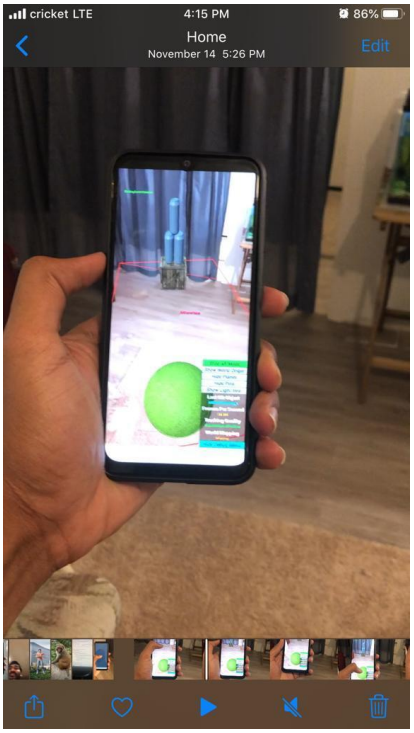

Figure 12: Ball and Coke Can in Real-World Space.

The second approach of the game is to run occlusion, which was achieved by using Swift ARKit.

To test the game, I used Apple iPhone 8 plus and Apple iPad 6th Gen both running IOS 13.2.3. To detect the physical object in the real world, ARKit object detection and recognition tool was used, I managed to detect multiple objects like Statue of liberty

decoration, a desk clock, a cardboard box, etc., which is done by detecting multiple edges of the physical object and making a 3D shape.

Now, the game is run by placing a virtual object, i.e., a trash can or a bin in the plane detected, and a virtual ball is tossed in the trash can. If the ball misses the trash can and goes behind the physical object, i.e. a cardboard box, then it is not visible unless the user holding the AR supported device moves the camera behind the physical object.

The following is the screenshot of the trash can and the physical object placed in the real world.

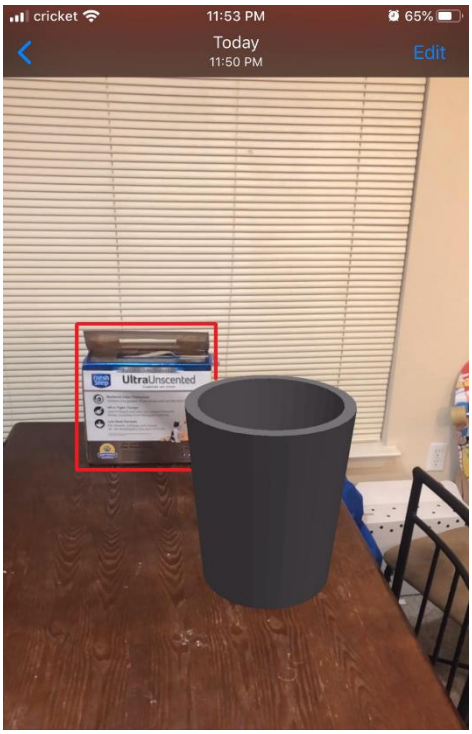

Figure 13: Physical Object Bin Placed in a real-world

The highlighted cardboard box in red is the physical object that was scanned by the tool. This helps the ARKit to create a virtual screen around the 3D physical object in space using the position from the origin. The location of the physical object is how the ball gets hidden behind the trash can which was achieved successfully by the ARKit.

The following screenshot shows how the user moves behind the physical object and find the ball.

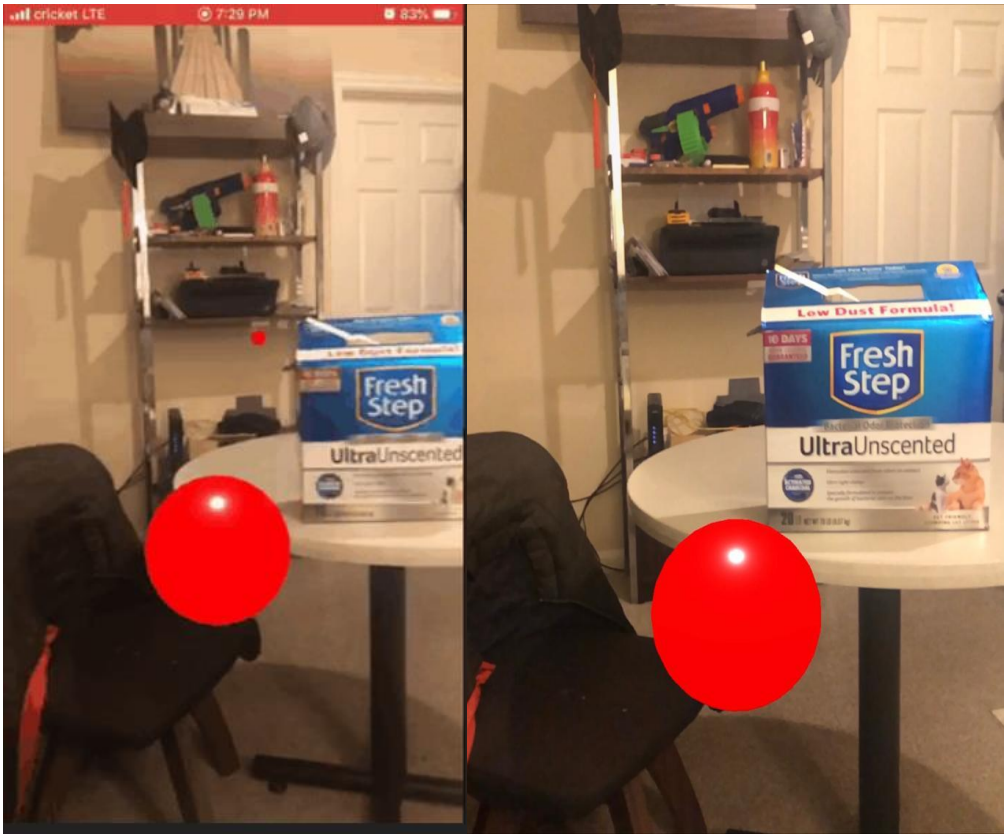

Figure 14: Ball Occluding behind the Real-World Object Cardboard Box

5. CONCLUSIONS

5.1 Summary

I was able to develop a game in augmented reality called Ball Toss. When I game application, it uses AR scanner and AR detector to scan the real-world objects, and coordinates of the real-world objects are detected, then a wall will be loaded for occlusion based on the object width and X coordinates. Using swipe gestures when I throw the ball towards the bin if the coordinated of the ball is within the coordinates of the inner radius of bin score will increase. If the coordinates of the ball are behind the coordinates of the wall, then the ball is hidden behind the wall.

5.2 Contributions

Essentially, occlusion implies that digital images may be concealed or occluded by real-world artifacts. It ensures that if a real-world object (like a person) stands between the computer camera, the virtual object will now be concealed behind that real-world object. Whereas virtual objects would have resided on a flat plain before, irrespective of the real world and the surroundings.

The occlusion technique implemented in the ball tossing can also be used in other games to scan the real-word objects like cardboard boxes, chairs, etc. to occlude the game objects. The occlusion part can also be used in other applications like augmented reality furniture apps.

5.3 Future Work

The ball toss game can be further improved by increasing the difficulty of tossing the ball. Makin the virtual object, i.e., bin to swing to increase the complexity of the game.

REFERENCES

[1] W. Strunk, E. B. White, R. Angell, *Elements of Style*, Fourth Edition, Longman Publishing, Ithaca, N.Y., 1999.

[2] A. Author, "Article in Conference Proceedings," *Title of Conference Proceedings*, City, State, Month, Year.

[3] Deepak Uplaonkar, Saurabh Saoji, "Virtual Furniture Application Using Augmented Reality," IJARCST, Vol.3, issue 1(Jan.-Mar.2015), pp. 156-160.

[4] Taiki Fuji, Yasue Mitsukura, Toshio Moriya, "Furniture Layout AR Application using Floor plans based on planar Object Tracking," IEEE 2012,pp.1-10.

[5] Mai Le, Aaron Zarraga, Kangrong Zhu, "An Augmented Reality Application for Previewing 3D Décor Changes", 2012

[6] Scott G. Dacko, "Enabling Smart Retail Setting via mobile augmented reality shopping apps", 2016

[7] Pokemongo , 2016, https://www.pokemongo.com/en-us/

[8] Ronald Azuma. "A survey of augmented reality". Presence, 6:355–385, 1995.

[9] About Augmented Reality and ARKit | Apple Developer Documentation In-text (Developer.apple.com, 2018) Apple Developer Documentation. [online] Available at: https://developer.apple.com/documentation/ARKit/about_augmented_reali ty_and_ARKit [Accessed 20 Apr. 2018]