COMP4801 Final Year Project
Individual Final Report

Augmented Reality Presentation Mobile Application

Li Chun Hang, 3035304259

Date of submission:
14th April, 2019
Abstract

Visual aid like presentation slides are an indispensable part of oral presentations. However, it is difficult to integrate into presentations well due to the separation between the presenter and the slides. By deploying augmented reality into visual aid, presenters and the visual elements in the slides will both appear in the same scene, leading to a more natural use of presentation slides. To achieve this, an easy-to-use presentation slides editor and player should be developed. With this application, skills in 3D graphics and programming are no longer required to produce such visual aid. It is expected that the application to be developed will increase the popularity of augmented reality in presentations.

Acknowledgement

We would like to thank Dr. Loretta Choi for the continuous support of this project.
Table of Contents

List of Figures .................................................................................................................. 4
List of Tables ..................................................................................................................... 4

1. Introduction .................................................................................................................. 5

2. Previous Works ............................................................................................................. 6
   2.1. Microsoft PowerPoint ............................................................................................... 6
   2.3. Nissan Teana Presentation in Guangzhou Auto Show ............................................. 8

3. Proposed Solution ......................................................................................................... 9
   3.1. Objectives ................................................................................................................. 9
   3.2. Use Cases ................................................................................................................. 9
       3.2.1. Education .......................................................................................................... 9
       3.2.3. Online Video Sharing ....................................................................................... 10

4. Application Design ....................................................................................................... 11
   4.1. Basic Flow .............................................................................................................. 11
   4.2. User interface Design ............................................................................................ 12
       4.2.1. Presentation Editor ............................................................................................ 12
       4.2.2. Presentation Player ............................................................................................ 13
       4.2.3. Remote Control ................................................................................................. 15
   4.4. Presentation Model ................................................................................................. 15
       4.4.1. Presentation ...................................................................................................... 15
       4.4.2. Slide .................................................................................................................. 16
       4.4.3. Visual Element .................................................................................................. 16
       4.4.4. Animation ......................................................................................................... 18

5. Technical Details ......................................................................................................... 19
   5.1. Unity ......................................................................................................................... 19
       5.1.1. Serialization and Deserialization of Presentation .............................................. 19
       5.1.2. Runtime Animation Generation ......................................................................... 20
       5.1.3. Object Selection in Editor .................................................................................. 20
       5.1.4. Object Movement in Editor ............................................................................... 20
       5.1.5. Camera Control in Editor .................................................................................. 21
       5.1.6. Imported Assets ................................................................................................. 22
   5.2. ARKit ......................................................................................................................... 23
       5.2.1. Inputs from ARKit ............................................................................................. 23
       5.2.2. Ground Plane Anchor ....................................................................................... 24
       5.2.3. Object Anchor .................................................................................................... 24
       5.2.4. 3D Object Scanner ............................................................................................. 26
   5.3. Comparison between ARKit and Vuforia ................................................................. 27
       5.3.1. Error Correction ................................................................................................ 28
       5.3.2. Ground Plane Geometry .................................................................................... 28

6. Test Results on ARKit .................................................................................................. 29
1. Introduction

Presentation slides are commonplace in oral presentations. Currently, they can be created using software such as Microsoft PowerPoint, Prezi and Apple Keynote. However, the slideshows made with these tools are isolated from the reality. This is especially problematic with the increased popularity of E-learning, where presentations are streamed online. In this setting, only one of the speaker or the slides is shown in the video, which may confuse the audience.

Augmented reality renders graphics on top of camera video, which allows the virtual objects to look and behave as they were real. This brings two advantages when used in presentations. First, it provides a more enjoyable and exciting experience to audiences. Second, it provides a way to display both the speaker and the slides in an E-learning session.

Augmented reality technologies has become mature in recent years. Compared to web-based product presentations, AR-based product presentations are said to be providing more effective communication benefit. Audience’s purchase intention and immersion is enhanced through enjoyment and excitement in AR experience [1]. Several software development kits have been released for making application with augmented reality support. However, this requires programming and thus not suitable for end-users. Therefore, it is beneficial to create a tool for the general public to take advantage of the technology in their presentations. Our solution to this problem is to develop an easy-to-use mobile application which allows users to create, edit and display augmented reality-enabled presentation slides.

This report first explores some popular presentation software and an example of augmented reality usage on presentation and highlight some of their features which should be considered in our project. Next, it introduces the design of our application on both the user interface and the technical details. Finally, it discusses the testing results of the application developed.
2. Previous Works

In this section, two presentation software and a presentation making use of augmented reality are examined. They all have their problem, but they also give ideas on what a successful presentation software should offer.

2.1. Microsoft PowerPoint

PowerPoint creates presentation slides which is displayed as a slideshow. Numerous templates and clip art are built into the application. Users can also apply animations on text, images and transition between slides. Because of these built in visual elements, users can quickly design aesthetically pleasing slides without having to possess good image processing and video editing skills.

Starting from PowerPoint 2016, users can also add 3D objects to the presentation. Rotation can be made to the model during the presentation to show different features of the object [2].

The disadvantage with PowerPoint is that it does not produce impressive presentation slides. Templates can be boring and it sometimes clashes with text. Animation and sound can also be a distraction to the audience [3]. PowerPoint also works poorly when the presentation is recorded since speaker and slides cannot be displayed together at the same time. Video editing has to be done to show both simultaneously, which is time-consuming.
2.2. Prezi

Prezi create presentation slides based on zooming in and out on a giant canvas. This provides more room for users to design unique presentation slides. Like Microsoft PowerPoint, it does not work well in recorded presentation. However, Prezi is planning to add augmented reality functionality to their product, which overlays the slides on the video [4]. This solves the problem with recorded presentation.

![Prezi AR](image)

**Figure 1** Prezi AR

Figure 1 shows one of the sample video of augmented reality in Prezi. Although this functionality has not been released yet, there are some possible limitations shown in the previews. First, the images do not react to speakers’ gesture. Also, the images are always on top of the video, so it may look unnatural when part of the images are supposed to appear behind the speaker.
2.3. Nissan Teana Presentation in Guangzhou Auto Show

Nissan has used augmented reality in their presentation of Nissan Teana during the Guangzhou Auto Show in 2013 [5]. They used it to present the internals of the car. Compared to showing the objects in a presentation slides, this is more immersive and will give a more believable experience to the audience.

![Figure 2](image_url) Usage of augmented reality in displaying car components

Compared to Prezi, this is a better use of augmented reality technologies. In this presentation, visual elements are aware of the camera angle and the movement of the speaker. This gives a more realistic result.

Similar presentations can also be found elsewhere, but it is often seen in big events done by professional. Our team hope to enable ordinary users to make use of augmented reality to enhance the visuals used in their presentations.
3. **Proposed Solution**

Regarding the current shortcomings of presentation software and augmented reality presentations, our team proposed a mobile augmented reality presentation tool. The tool should be versatile yet easy to use so that it can benefit the public.

3.1. **Objectives**

There are two goals which the augmented reality presentation tool should achieve.

1. **Popularize the use of augmented reality**

   With the existence of augmented reality libraries such as ARCore, ARKit and Vuforia, augmented reality application starts to become popular. It is mainly used in retail such as IKEA Place and Dulux Visualizer and games like Pokemon Go. However, these apps are targeting to very specific group of people. If a person does not shop at those shops and he is not a gamer, he is not benefitted by the technology.

   By developing an augmented reality presentation tool, more people will have the chance to use augmented reality technologies because presenting is a common activity among people around the world.

2. **Make augmented reality content creation possible**

   Currently, most of the augmented reality applications present predefined content to users. It is difficult to create custom augmented reality experiences without any knowledge on programming. An augmented reality presentation tool will diversify the type of content delivered with the technology.

3.2. **Use Cases**

3.2.1. **Education**

   Education is one area in which presentations are common. To deliver ideas clearly, props are often used. With the proposed presentation tool, props can be displayed electronically. This has the advantage of being easy to carry around since mobile devices are designed for mobility.

   Both teachers and students can use the application. Teachers may prepare 3D models such as a human skeleton or an erupting volcano to class and display them through the tool. This may liven the lecture up and increase students’ attentiveness.
Students also do presentation and they may need to present 3D objects. For example, students may have created a robot and they may be asked to present their work in front of class. Robots may be heavy and fragile and it may not be good to move them to the classroom. The proposed tool will make the use of props easier for those students.

3.2.3. **Online Video Sharing**

A lot of presentation nowadays are not carried out directly in front of people but are instead delivered through the Internet. Currently, a lot of editing work has to be done if the presenter uses visual aids or visual effects in the presentation. In an augmented reality presentation tool, both the visual elements and the video stream are blended together automatically, saving the time in editing videos.
4. Application Design

Our project consists of a presentation editor and a presentation player. The entire project will be implemented as a mobile application so that users can create and deliver presentations using the same device. This leads to an easier setup for presentations compared to a desktop application since it does not require installing additional software on any computer or connecting camera to a computer.

4.1. Basic Flow

![Flowchart of the program]

**Figure 3** Basic flow of the program

Figure 3 shows the flow of our program during normal usage. Users first have the choice to create or load a presentation. Then, the program brings users to the presentation editor. Apart from editing slides, users can also access the presentation player from it. The player finds the anchor points with user input. Finally, the presentation slides are displayed once anchors are found.
4.2. User interface Design
The application focuses on reducing user input in the process of creating an augmented reality experience. Therefore, its user interface only includes minimal options while presenting a way for users to set parameters needed for augmented reality in an understandable way. Visual elements in the presentation are anchored either to a ground plane or an object in the real world. Because of this, there are two corresponding versions of presentation editor and player.

4.2.1. Presentation Editor
Users create their slides in a 3D environment in both versions of the editor. In the presentation editor for ground plane anchor, a stage border is provided in the environment and users may insert text and 3D models within the border. The centre of this border will be the anchor point on the ground plane when the presentation is viewed. The border is displayed so that it gives users an idea of the position of visual elements in relation to the stage.

![Figure 4](image)

**Figure 4** Layout of presentation editor for ground plane anchor

Figure 4 shows the screen when editing a slide in the editor for ground plane anchor. At the top of the screen are the buttons for saving the slides and entering the presentation player as well as the filename of the presentation. The main part of the screen is splitted into left and right portions. The left portion is a list of slides in the presentation initially. When a visual element is selected, it turns into a menu for modifying the attributes of the element. The right portion is larger and it mainly shows the stage which is being worked on. Users select visual elements by tapping on their models.
on the screen. The three button at the lower left corner of the right portion adds text, models and animations to the slides respectively.

![Figure 5](image.png)

**Figure 5** Layout of presentation editor for object anchor

The presentation editor for object anchor shares most of the designs with the other version of editor. In this mode of anchor, visual elements are located around an object instead of on a ground plane. Therefore, a point cloud of the object is displayed instead of a stage border. Also, the list of slides has been changed to a list of object anchors, since visual elements appears whenever their associated objects are detected rather than being displayed sequentially.

### 4.2.2. Presentation Player

Presentation player displays slides created with the editor. It opens the camera on the device for capturing video on which objects are drawn on. In the ground plane anchor version of the player, the position of the stage has to be defined before the start of the presentation. This is done by pointing the device to the desired location of the stage. The stage border appears at the pointed location when a ground plane is found and users confirm the anchor position by tapping on the screen. Since the stage dimensions are defined without unit, users can also alter the scale of the stage using pinch-to-zoom gesture. Figure 3 shows the user interface design for user to locate the stage.
Figure 6  Stage finding before presentation (top); After confirming anchor position (bottom)

After setting up the ground plane anchor, the program renders visual elements on the screen. The presentation can be started and the scenes produced can then be projected onto a screen.
Figure 7 Object anchor (cup) detected

In the object anchor version of the player, visual elements appears as soon as their associated objects are detected. Therefore, users only have to point the device at the targeted objects to start the presentation. Since there are different animations for different anchors, users have to select an anchor by tapping on the detected object before playing animations of the selected anchor.

4.2.3. Remote Control
A remote control mechanism is needed when playing the presentation so that control button can be hidden and enhance the immersiveness of the augmented reality experience. This is currently implemented using the input from a Bluetooth keyboard, with “W” set to playing the next slide or animation and “S” set to go back to the previous slide or animation.

4.4. Presentation Model
This section focuses on how the presentation is modelled. This model is crucial in the storage of presentation.

4.4.1. Presentation
Presentation class holds all the features in a presentation together. It contains a list of slides and also the stage dimensions and whether the presentation makes use of ground plane anchor or object anchor. Then, the application can load the appropriate version of the presentation editor and player and construct the presentation with the given slides.
4.4.2. Slide

Slides describe what is being displayed at a given slide or around an object anchor. It contains a list of visual elements and a list of animations. These two lists are needed for both types of presentations. To support object anchor presentations, it also contains a reference to the targeted object’s data, which is only used in the object anchor version of editor and player.

4.4.3. Visual Element

1. Text

Text is a common feature in presentation slides. Because of depth in 3D slides, characters should be rendered with meshes to make them stand in a scene. Unity uses TextMesh Pro in accomplishing such effect by default. This package draws characters by putting textures into rectangles. It is capable of generating 3D text with different fonts and typographical emphasis. However, the characters TextMesh Pro creates are not volumetric, which looks unnatural when viewed at an angle.

The team has found another package on Unity Asset Store called “Simple Helvetica”. This package creates volumetric characters but it only supports one font and a limited set of characters. This is because all the character models are precomputed and it uses a script to bring the models to the scene. If there is no matching model to the character, errors may occur.

Despite the limitations of Simple Helvetica, it is used in the project since it gives a more natural look than TextMesh Pro does. It also has the benefit of being simple in design and so it is easier to extend its functionality.

Figure 8 and 9 shows the difference in results of generating 3D text using TextMesh Pro and Simple Helvetica.
2. **3D Model**

Augmented Reality is most beneficial to presentations which introduce 3D objects. Therefore, it is important to let users place 3D models in their presentations. Currently, only model presets can be loaded into presentations.

The challenge in implementing a 3D model importer is the mapping of materials to the mesh. Most 3D model file formats do not embed textures into the file and so separate texture files are needed. While it is possible to assign materials to meshes through scripts, model file distribution contains different number of texture files and materials can have different ways of representation. For example, Unity expects materials to store normal and
alpha maps in different bitmap images while some distributions may put such maps into
different colour channels in a single image file [6]. Because of the complexity of model
materials, it is difficult to generalize the model import procedure into an automated model
importer.

The problem of the lack of a model importer can be mitigated by including a good variety
of model presets. Not all users have their own 3D model for display and they will have to
rely on the presets contained in the application. Also, presets helps with the workload of
presentation creators since it avoid extra work in finding and processing models files.

3. **Stage**

Stage models the physical stage on which a presentation is delivered. It is a special type of
visual element found in the project such that there is always a single instance of it in a
presentation. In the slides editor for ground plane anchor, Stage is represented by a
horizontal plane. The same rectangle also appears in the corresponding slides player as a
marker of the detected ground plane.

Stage object stores the dimensions of the stage without unit since it is only used to visualize
space in the presentation editor. The scale of the visual elements in the presentation is
meant to be adjusted just before the start of the presentation.

4.4.4. **Animation**

Animations adds motions to the presentation. Each animation is modelled as a linear interpolation
between the start state and the end state over a duration of time. Each state stores positions,
rotations and opacity of the visual element.

Even though all animations are associated to a visual element, they are represented in Slide class
instead of Element classes. This is because animations are played in sequence within a slide.
5. Technical Details

This section focuses on the technical details in the implementation of the application design.

5.1. Unity

The proposed application is implemented on Unity. Even though the application is not a game, Unity is chosen because it comes with a rendering engine and it contains built-in functions for calculations in 3D geometry.

5.1.1. Serialization and Deserialization of Presentation

To save presentation into files, it is serialized into JSON format so that it can be saved in text documents. Since Presentation is implemented as a static class so that it can be accessed by all Unity scenes, there are no instances of Presentation object and thus it cannot be serialized by itself. The workaround the team has chosen is to define a non-static class with identical attributes as the static Presentation. When a presentation is saved, an instance of such non-static class is created with all the data from the static class. After this creation, the non-static class can be serialized and stored in a file. When a presentation file is deserialized, a presentation object is created and so the data can be copied to the static Presentation class.

“JSON .NET For Unity” from the Unity Asset Store is used instead of the built-in serializer in Unity. The problem for Unity’s serializer is that it does not support serialization of polymorphic classes [7]. When it encounters instances of such classes, they are treated as instances of their super class. In the application, visual elements are all extended from Element class and they have their unique attributes. Using the built-in serializer will cause data loss.

Serializing GameObject, the building block of a Unity scene, is also problematic. Because of some deprecated properties in the GameObject, “JSON .NET For Unity” was not able to handle GameObject by default [8]. Because of this, all the classes which need serialization are designed to work without having to save GameObject. These classes only store information needed to construct the GameObject in the scene and so the application never serializes GameObject directly.
5.1.2. **Runtime Animation Generation**

Unity’s Animation system is used in its implementation. Animation component is added when an Element object is created. To generate the animation during runtime, linear AnimationCurve are created using attributes of the two states and the duration. Then, the curves are assigned to relevant attributes in the associated Element object and put into an AnimationClip. Finally, the AnimationClip is assigned to the Animation component in the Element object.

5.1.3. **Object Selection in Editor**

Users may select any visual element by tapping on the related object in the scene. The program uses ray casting to check if an object is touched. A bounding box is created for every object in the scene. The ray shoots from the camera to the touch position on the screen and continues forward. If the ray hits a bounding box, the object of the box will be selected. If the ray does not hit anything, any selected object will be deselected.

5.1.4. **Object Movement in Editor**

Once an object is selected, users may change its attributes, including its position. Objects may either move horizontally or vertically. To move an object, users drags on the screen and the object will follow the touch position.

On the start of object movement, the program creates a logical plane which intersects with the touch position. A horizontal plane will be created for horizontal movement and a vertical plane will be created otherwise. For each update, ray casting is used to find the touch position on the plane. If such point exist, the object will move to the location.

Since there are two plane of movement and an object can only move along either of them, different gestures have to be defined to differentiate the two directions. Simple dragging on the visual element moves it horizontally. If the user press and hold the object before dragging, the object moves vertically.

Objects can also be rotated about the y-axis using two-finger gesture. When there are two fingers touching the screen, the application calculates the angle between a horizontal line and the line which the two fingers make. For each screen refresh, the object then rotates with the change in such angle.
Figure 10  Required angle denoted by $\theta$

5.1.5. Camera Control in Editor
To allow users to look around the scene, camera can orbit around the centre of the scene. Camera orbit is a movement such that the camera moves along a sphere around the centre of the scene and it is always facing the centre. Users can also zoom in and out by pinch-to-zoom gesture. This is accomplished by altering the field of view of the camera.

Figure 11  Camera moves along the surface of a sphere
5.1.6. Imported Assets

Unity is known for its Asset Store, where there are numerous useful assets for use with Unity projects. The team also leverages these resources to speed up the development process of the project. Below is a list of assets which are used in the project.

<table>
<thead>
<tr>
<th>Asset Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unity ARKit Plugin</td>
<td>Interfaces with the native ARKit API.</td>
</tr>
<tr>
<td>iOS for Unity UI</td>
<td>User interface widgets with the look and feel of a native iOS application.</td>
</tr>
<tr>
<td>UGUI Material Design Icon</td>
<td>Icon pack used for the control buttons.</td>
</tr>
<tr>
<td>JSON .NET For Unity</td>
<td>Serializer from Unity objects to JSON format.</td>
</tr>
<tr>
<td>Simple Helvetica</td>
<td>Volumetric 3D text.</td>
</tr>
<tr>
<td>3D Games Effects Pack Free</td>
<td>Effects for the Particle System as a part of the Animation implementation.</td>
</tr>
<tr>
<td>Double Sided Shaders</td>
<td>Allows for rendering both sides of mesh surfaces with transparency.</td>
</tr>
<tr>
<td>iOS External Input</td>
<td>Retrieves Bluetooth keyboard input.</td>
</tr>
</tbody>
</table>

Table 1 List of assets used in the projects
5.2. ARKit

ARKit is the augmented reality library for iOS devices maintained by Apple. All iOS devices with an A9 processor or newer running on iOS 11 or above can use this library [9].

5.2.1. Inputs from ARKit

ARKit describes objects’ pose, which is the position and rotation, in a 4 x 4 transformation matrix. This includes the camera, the detected planes and objects. From the transformation matrix, it is possible to recover the position and the rotation of the object. Given a transformation matrix,

\[ M = \begin{pmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \\ 0 & 0 & 0 & 1 \end{pmatrix} \]

The process can be described as follows.

1. **Position**

   The position, also known as translation, can be found in the last column of \( M \).

   \[ T = (d \quad h \quad l) \]

2. **Rotation**

   Since unity stores rotation in quaternion, the process aims at finding rotation expressed in quaternion.

   \( M \) contains only the object’s rotation after removing the translation portion of the transformation matrix. Since rotation matrix is orthogonal and its determinant equals to 1, the magnitudes of the components in the quaternions can be calculated as follows:

   \[ w = \sqrt{\frac{1 + a + f + k}{2}} \]

   \[ x = \sqrt{\frac{1 + a - f - k}{2}} \]

   \[ y = \sqrt{\frac{1 - a + f - k}{2}} \]

   \[ z = \sqrt{\frac{1 - a - f + k}{2}} \]
Finally, $x, y, \text{ and } z$ takes the signs of $(j - g), (c - i) \text{ and } (e - b)$ respectively [10].

After decomposing transformation matrices, the results can be applied to the Transform component of GameObject to form a virtual scenes similar to any other Unity scenes. The major difference is that the camera’s pose will follow the relative movement of the device and that the scene is blended into a video stream.

5.2.2. Ground Plane Anchor
ARKit scans ground plane based on the feature points detected. It is capable of scanning both horizontal and vertical planes as well as the geometry of the detected plane. In this project, only horizontal planes are detected as it focuses on placing visual elements on a stage, which is mostly a horizontal surface.

During the stage finding process in the ground plane anchor player, ray casting is used at each frame to determine the the point where the devices points at. The ray shoots from the centre of the screen in the forward direction of the device. If the ray intersects with any of the detected plane, the intersection point is considered as a possible anchor position and users can set this point as the anchor by tapping on the screen.

5.2.3. Object Anchor
Apart from ground plane detection, ARKit also support object detection and tracking. It takes a .arobject file as input, which is said to contain everything needed for object anchor feature. It is possible to access the 3D points and other data derived from it such as the bounding box size and the centre of the points from Unity. Since .arobject is a binary file format and it is only possible to decode using iOS devices, the actual content of such file remains unknown.

Using the point cloud data from the .arobject file, the point cloud is visualized in the object anchor editor. Since there is not colour data, all the points are drawn as white dots. Below are some examples of point cloud visualization in the editor.
When the object anchor version of the presentation is played, users have to tap and select an object before playing animation associated to the object anchor. This requires adding a bounding box around the object so that ray casting technique can be applied in object selection.
After an object is detected, ARKit returns the pose of the object anchor. As the centre of the point cloud in the .arobject is defined with respect to the origin, the centre of the bounding box equals to the position of the object anchor plus the centre defined in the file. After calculating the centre, a bounding box can be defined using the box size given in the .arobject file.

### 3.2.4. 3D Object Scanner

Apple has provided a sample iOS application on 3D object scanning and saving the results into .arobject files. To use this application, we first define the bounding box of the object to be scanned. Only points located inside this box will be recorded and all the other points are discarded. Then, we walk around the object slowly so that the application can detect feature points of the object from all angles.

**Figure 13** Defining bounding box (left); Scanning object (middle); Adjusting the origin of the model (right)

After the scanning, the scanned object can be tested with the object detection feature of ARKit. To get a better idea of the performance of the model, it is encouraged to test detection in different environment. It is also possible to merge the result with another scan.
Figure 14 Testing model after scanning. Note the different coloured points.

In the graphics of the object detector, feature points which are actually detected for ARKit to recognise a 3D object are rendered in green. Such feature points only account for a small amount of the points in the detected object. Therefore, it is better to get more feature points of an object so that there is a higher chance of finding matches between points from the scanned object and that from the current ARKit session.

To use the .arobject files generated by this sample in our application, we may first transfer the file to a computer. Then, transfer the file to our application using iTunes.

5.3 Comparison between ARKit and Vuforia

In the early stage of the project, Vuforia was the augmented reality library of choice due to its portability. However, it is decided that ARKit is a better solution for our project because of its ability to correct plane detection error and to output ground plane geometry.
5.3.1. Error Correction
Feature point detection is erroneous. Since ground planes are determined by feature points in both augmented reality libraries, errors may occur in the ground plane estimation. Vuforia make use of ARKit internally on ARKit-enabled devices and so the feature points it considers should be the same as ARKit.

During our testing, we observe that ARKit updates the ground plane detection results as new feature points get detected and bad feature points are removed. Whereas the detection results given by Vuforia remain unchanged throughout the session, including the erroneous results.

5.3.2. Ground Plane Geometry
ARKit has the capability to return the geometry of a detected ground plane. Taking this into account in the stage finding process of the presentation player, this gives a more robust feeling of the plane detection.

Vuforia allows ground plane anchor to be placed on the ground behind a wall. While this does not affect accuracy of the anchor, it does not give a good impression to the user.
6. Test Results on ARKit

This section focuses on the tests carried out regarding the performance of ARKit.

6.1. Stability under Motion

The immersiveness of augmented reality is given by how well objects are fixed in place. Therefore, it is important to have a stable motion tracking.

To test the stability, we first find a place with good amount of corners so that there are enough feature points for tracking. Then, we placed a 3D model on top of a cross stuck on the floor. Tracking is said to be stable if the model stays in place.

Figure 15 Initial position of the 3D Model
First, the camera is moved slowly. The drifting of the anchor position is not observable.

![Position of the 3D Model after slow movement](image)

**Figure 16** Position of the 3D Model after slow movement

Then, the camera is moved quickly. There was a slight drift in the anchor position.

![Positions of the 3D Model after quick movements](image)

**Figure 17** Positions of the 3D Model after quick movements

The result ARKit give is satisfactory. The model is still on top of the cross despite the drift. In a presentation setting, camera is not supposed to be shaken violently. Therefore, it is likely that ARKit will give good results in the normal use cases of the application.
6.2. Ambient Lighting

Lighting have an effect on feature point detection since it changes how images look. Therefore, the team tested the detection under normal and dim lighting.

Figure 18  Ground plane detection under different lighting conditions
The ground plane was detected almost instantly with good lighting, while it is not detected at all in a dim environment.

Good lighting is important in feature points detection. There must be changes in colour in the picture so that feature points can be detected. When there is not enough lighting, the contrast between colours is not high enough for ARKit to pick up feature points.

This can also extend to surfaces with no clear texture. Since there is not enough feature points on the ground, ARKit may not be able to detect the ground plane.
7. Future Works

7.1. Remote Control
Our application currently only support Bluetooth keyboards for remote controlling purpose. However, it is cumbersome for the speaker to hold a keyboard during presentation. Therefore, a remote control application for smartphones or a piece of hardware should be developed for users to switch slides.

7.2. Multi-user Player
To provide a more interactive experience to the audience, the presentation player should support multi-user viewing. ARKit supports this kind of functionality by providing an ARWorldMap in each session, which can be used to synchronize world coordinate between devices [11].

The presenter’s device can act as a server. Audience views the presentation by establishing a connection with the presenter’s device and download the ARWorldMap for synchronization. After that, whenever the presenter plays an animation or switches slides, the same action is also reflected on audiences’ screens.

7.3. User Interface Enhancements
Our application aims at lowering the learning curve of creating augmented reality experiences. Therefore, the user interface should be made as clear as possible to new users. There should be more instructions displayed on the user interface so that it helps users to get used to using our application.

Another possible enhancement is to avoid the need of typing in changing attributes of elements or animations. It is because mobile device screens tend to be small and typing using touchscreens can be frustrating. Typing can be replaced by gestures or sliders, which require less precision on touches.

7.4. 3D Model Import
Currently, only model presets can be inserted into the presentation. There is no way for users to add models which they need when there are no suitable presets. Therefore, there should be a way for users to import model files.
Ideally, there should also be a feature for users to scan objects which they want to use in their presentations. The challenge in such feature would be generating meshes from sparse point cloud ARKit generates and to create and map texture onto the mesh.

7.5. **Object Scanner for Object Anchor**

The team uses a sample code from Apple to generate .arobject files for use in the object anchor feature. Since this requires compiling of the sample code, it is not a solution for end users. Also, colour information is not available if this scanner is used. This makes the visualization in the presentation editor uncoloured and limits the clarity of the visualization. By including an object scanner in our application, both problems can be solved.

8. **Conclusion**

We have examined the current status of presentation application and the usage of augmented reality in presentation. The current presentation applications do not cope with the popularization of E-learning because of their way of displaying slides. Despite the maturity of augmented reality technologies, it is still too complicated to use for most people.

The team has successfully implemented a proof of concept of an augmented reality-enabled presentation tool. Basic features such as object placement and animation have been implemented, but more work has to be done before the product can be considered commercially viable.

Once the enhancements are made to the current implementation, augmented reality will become more accessible to normal people. Presenters will have a better way to display their slides while audiences can enjoy the new visual effects. Both of them will have greater exposure to augmented reality. With so many presentations delivered every day, the application could be an effective way to promote augmented reality.
References


