Trinity College Dublin
Coláiste na Tríonóide, Baile Átha Cliath
The University of Dublin

SmartPhotos

by Ujjawal Aggarwal

AR PROJECT

in completion of the BAI Computer Engineering

School of Computer Science & Statistics

Trinity College Dublin, Ireland

Declaration

I declare that this project has not been submitted as an exercise for a degree at this or any other university, and it is entirely my own work.
I agree to deposit this project in the Universitys open access institutional repository or allow the library to do so on my behalf, subject to Irish Copyright Legislation and Trinity College Library conditions of use and acknowledgment.

Ujjawal Aggarwal

May 22, 2021

Contents

Chapter 1

Introduction

This chapter will include the objective and motivation, i.e., why we chose this particular use case for our project and its impact on the user's life. This chapter will also detail the technology used in this project.
We have created an application for iOS mobile users that will help them to refresh their memory through their precious memories and videos.

1.1 Objective and Motivation

The motivation and objective behind this project are to disrupt the natural way of clicking pictures and storing precious memories. Whenever anyone goes through their picture albums, the only thing they regret is they cannot rewind their life and go back through in time and re-watch this priceless memory.

Keeping this goal in mind, we have created an app, SmartPhotos, capable of scanning pictures and playing the video on the top of the photo, given users have some 20-30secs clips related to that picture. SmartPhotos are the photos that can play the video when scanned by the application. Anyone can relive their memories without worrying about their mobile storage. We got this idea from the movie Harry Potter when Harry reads some newspaper and all the pictures in that paper start playing video related to the news. This gave us the idea that why can't we apply this concept to the Photos with the help of AR technology and helps users to enjoy the same vibes and relive the uncluttered memories.

1.2 Technology used

The technology used in this project and with their description is as follows:

- **Xcode**- This application is coded in Swift language and build in Xcode. Xcode is developed by

Apple and is mainly used to build iOS applications. The drawback of the Xcode is that it can only run on Apple's Macbook laptops. The key point of Apple's Xcode is that it provides almost every technology on one platform only and provides its boilerplate code too.

- **ARKit**- It is a framework that provides high-class tracking and recognition abilities. It is developed by Apple and allows the AR community to use it for research and education purposes. It is built upon modern computer vision techniques. Xcode came with an integrated ARKit and required no other installation. We have used ARKit capabilities to track the given image, and we have tested its tracking abilities in very extreme conditions.
- **SceneKit**- It is a framework developed by the Apple community to build 3D games and to add 2D and 3D contents to the given scene. SceneKit combines a very high-performance rendering engine with easy import APIs. SceneKit only requires the description of the scene and can perform the action we want to do. We have used this framework to put video, a 2D object, on the image tracked by ARKit.

Chapter 2

Implementation

This chapter will detail the implementation of our application with the help of small and important code snippets used. We coded this project in swift, and the code snippet shown below will be of language swift. This chapter will also detail the assets used in creating SmartPhotos.

2.1 Assets Used

We have created 2 pictures for this project, and all of the assets used in creating those pictures are taken from YouTube. 1st SmartPhoto comprises Leonardo DiCaprio's Oscar-Winning moment. 2nd SmartPhoto comprises actress Anuskha Sharma and cricketer Virat Kholi's wedding.

2.2 Implementation

The following code snippet is used to create the session used for tracking images. We have used the *trackedimage* variable to track the AR Resource group's images and set the maximum number of images that can be tracked simultaneously equal to 2. It means that the user can scan 2 images and see 2 videos together on a single screen.

```swift
// Create a session configuration
let configuration = ARImageTrackingConfiguration()
if let trackedimage = ARReferenceImage.referenceImages(inGroupNamed: "pics", bundle: Bundle.main){
configuration.trackingImages = trackedimage
configuration.maximumNumberOfTrackedImages = 2 }
```

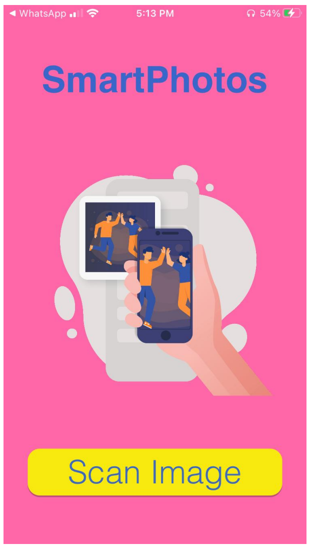
The subsequent code snippet is the most important code, and its main function is to identify the image and render the video on that image. We have created a function *renderer* that returns the SceneKit SCNnode type *node*.

This function will initialize the SKScene type *scene* with the given width and height. Next, we have created a *node* of type SCNNode, and all the future nodes will be added as a child node to this node. Afterward, we will be looking for the image anchor, and if found, it will create a SCNPlane type *plane* with the height and width of the tracked image. Next, we have created a SCNNode type *planenode*, and its geometry will be equaled to a *plane*. Then, the planenode will be rotated by -90 degrees for better visualization. Afterward, we will look for the video which is related to the tracked image. Finally, we have created a SKVideoNode type *videonode* that accepts the given video on the image and will play it as soon as the user scans the image.

```
func renderer(_ renderer: SCNSceneRenderer, nodeFor anchor: ARAnchor) -> SCNNode? {

        let scene = SKScene(size: CGSize(width: 640, height: 360))
        let node = SCNNode()

        if let imageanchor = anchor as? ARImageAnchor{
            let plane = SCNPlane(width: imageanchor.referenceImage.physicalSize.width, height:imageanchor.referenc
            plane.firstMaterial?.diffuse.contents = scene
            let planenode = SCNNode(geometry: plane)

            planenode.eulerAngles.x = -.pi/2
            node.addChildNode(planenode)

            var imagename = imageanchor.referenceImage.name ?? "unknown"
            var filename = imagename+".mp4"
            let videoNode = SKVideoNode(fileNamed: filename)

            videoNode.play()
            videoNode.position = CGPoint(x: scene.size.width/2, y: scene.size.height/2)
            videoNode.yScale =   -1.0

            scene.addChild(videoNode)

        }

        return node
    }
}
```

Fig2.1(a) shows the icon created for the application, and Fig2.1(b) depicts the front page build for the application.

(a) Icon



(b) Front Page

Figure 2.1: Front page and Icon

Chapter 3

Results and Future Work

This chapter will detail the results and analysis of our application, SmartPhotos. It will also include future work or room for improvement in the project.

3.1 Results and Analysis

Our application delivers auspicious results as expected. Our SmartPhotos app perfectly scans the image and plays the video on the top of the image. As our result is stored in the form of the video and cannot be shown here. We have uploaded our working application video on YouTube and can be found here.
As we can see in the YouTube video, moving mobile phones doesn't obstruct the video because of the tracking abilities offered by ARKit. Also, video is very stable when playing, and it possible with the SceneKit, which offers this service.

3.2 Future Work

Because of time and knowledge constraints, there is a lot of scope for improvement in our application. Some of these are the following:

1. **Double sound**- As it can be seen in the video, whenever the scanned image is changed, the previous image's sound was still playing in the background. This problem needs to be rectified.

2. **Database Capabilities**- The video and images used in this project reside on the app and increases the size of the application. If we need more images to scan, it utilizes more mobile storage. To solve this problem, we need to connect the database to the application. This helps to reduce the application size by a large difference.