VIT
UNIVERSITY
(Estd. u/s 3 of UGC Act 1956)

HUMAN COMPUTER INTERACTION

J COMPONENT REPORT

Submitted To:

Prof. Divya Udyan

Group Members

Arun Giri	17BIT0216
Sahil Banthia	17BIT0185
Chirag Lakhani	17BIT0135
Shubham Totla	17BIT0185

TABLE OF CONTENTS

ABSTRACT:

Augmented reality is an interesting feature in newspapers that takes readers beyond the printed page. It enables people to see a video, animation, or other unexpected content that is apparently located on a page of their newspaper. The illusion is created when an augmented reality program and the camera software of a smart phone or tablet work together. The newspaper must be viewed through the camera of the mobile device in order for the illusion to work. In addition, the device must be connected to the Internet.

INTRODUCTION-

Newspapers around the globe have introduced Augmented Reality in their newspapers in an attempt to enhance their product and attract people especially youth to increase their waning number of readers.

When augmented reality is used with a newspaper, a photo in an article is scanned with the camera of a mobile device. The AR software identifies the photo and then loads related digital content. The content is displayed in the device's camera view. It's often positioned over the scanned photo so that the digital content appears to have replaced the photo.

Although the content that is loaded is frequently a video, there are many other possibilities. A photo gallery, the latest news updates or sport scores, related social media information, educational animations, additional facts, a relevant map, a restaurant menu, a competition entry, or a reservation page for a special event are all possible uses for newspaper AR.

PROPOSED SYSYTEM-

In these AR newspapers, moving scenes appeared on a page containing otherwise static content. This type of newspaper would be wonderful in real life, but it doesn't exist (yet). The appearance of movement on a page of today's newspapers is a trick that requires special equipment.

The reader must have a smartphone or tablet with a camera as well as an Internet connection in order to use the AR features of a newspaper. In addition, the augmented reality software must be able to link to the software controlling the camera of the mobile device. This should be no problem when using an iOS or Android device.

Augmented reality in a newspaper works via image recognition. With the aid of the camera app in the mobile device, the AR program identifies a photo and loads the digital content that is linked to that photo. As viewed through the camera, the linked content often appears as an overlay on top of the image that was scanned. If the digital content is a video, it does give the newspaper page a slightly Harry Potter-like appearance as it plays. Once the content is loaded, when the mobile device is moved away from the newspaper the content stays in view.

OBJECT DETECTION AND TRACKING

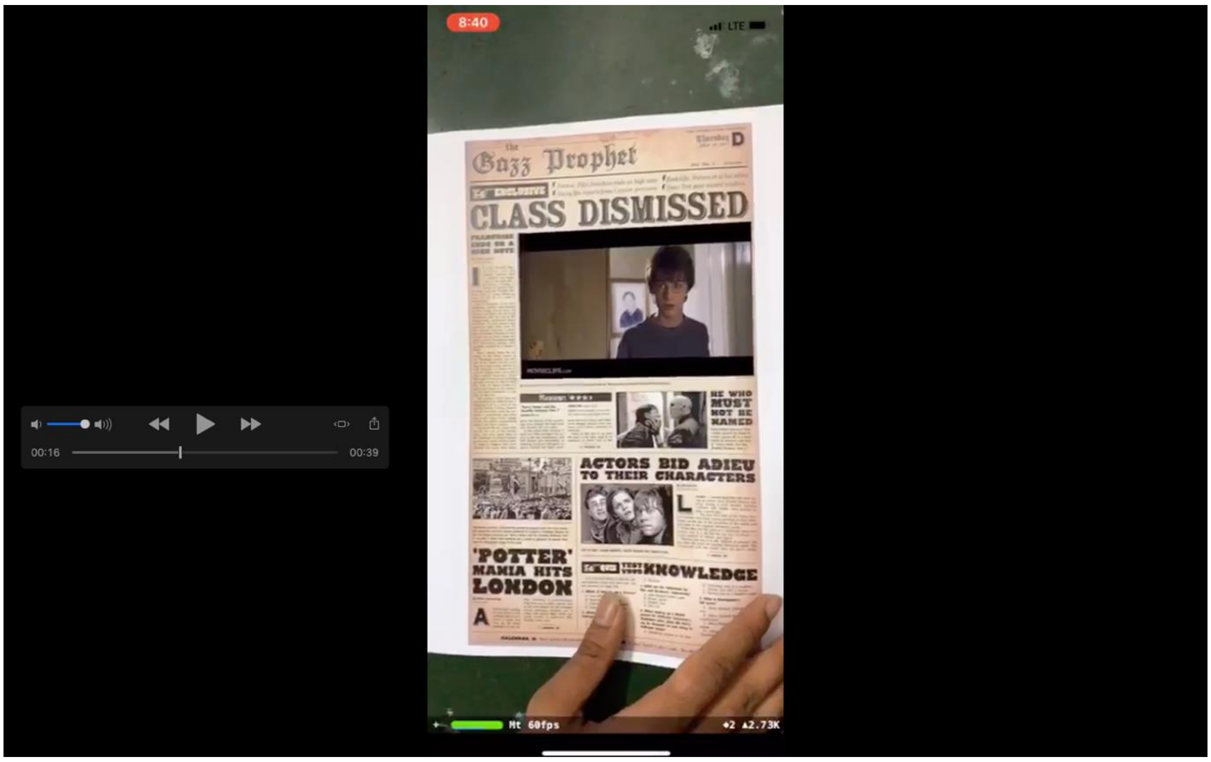
ARKit 1.5 added support for 2D image detection, letting you trigger an AR experience based on 2D images like posters, artwork, or signs. ARKit 2 extends this support to offer full 2D image tracking, so you can incorporate movable objects like product boxes or magazines into your AR experiences. ARKit 2 also adds the ability to detect known 3D objects like sculptures, toys, or furniture.

Marker-Based AR

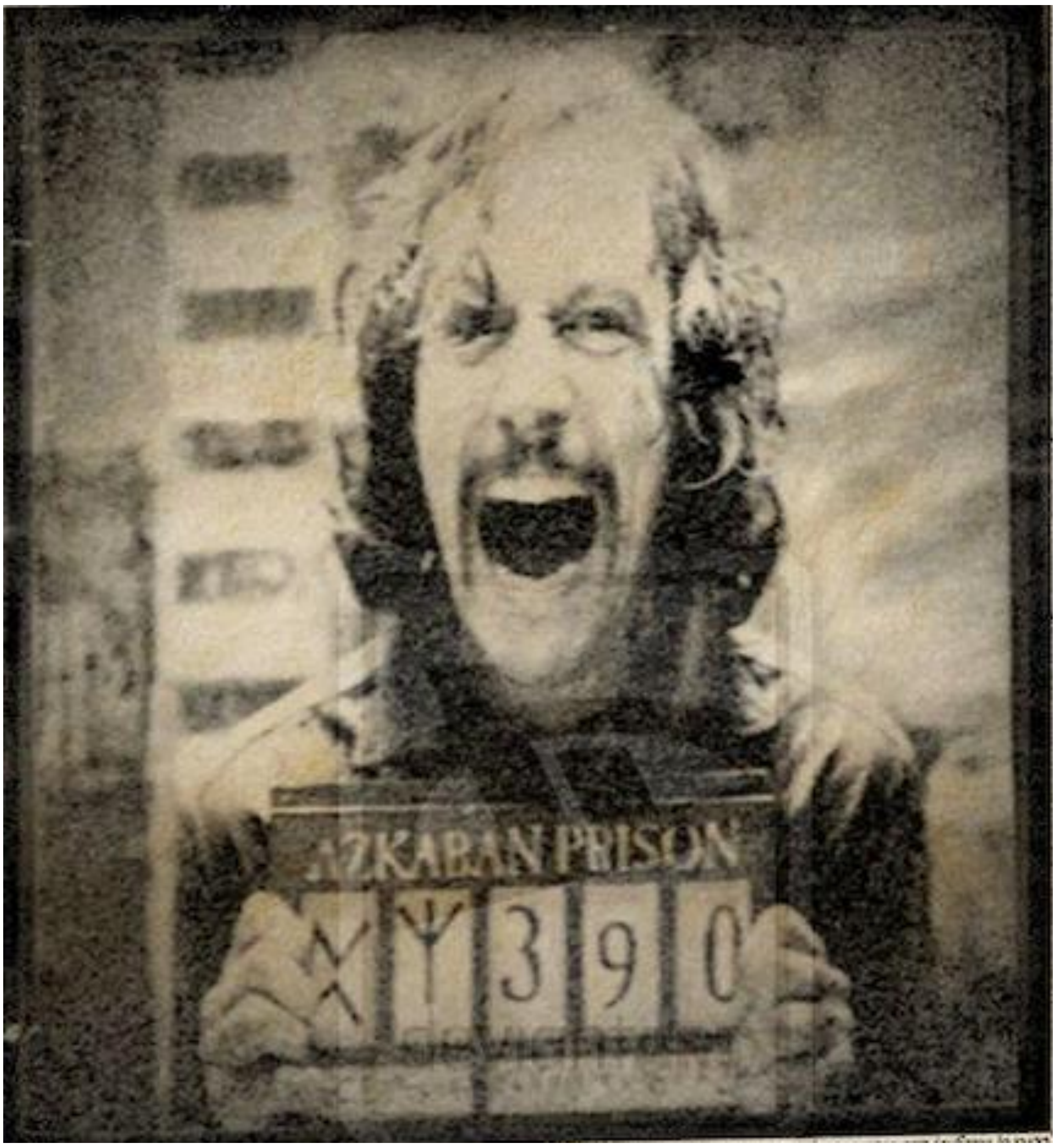
The augmented reality in this newspaper is marker based. The steps in using the AR from this newspaper on a mobile device such as my iPhone were as follows. The newspaper app was available as a free download at the Apple Store.

1. Open the newspaper app.
2. Click the AR symbol on the front page of the app.
3. The app opens the camera application on the iPhone in scan mode. Fill the screen with the photo to be scanned.
4. The photo is automatically scanned as a vertical green line moves over the photo. Markers (are temporarily laid down as the scan line moves.
5. Information from the markers is sent to a newspaper computer.
6. The computer compares the information from the scanned photo with the photo information stored in its database until it finds matching data.
7. Once a match is found, the computer performs the action that it's programmed to carry out when that photo has been identified (such as loading a particular video or slide show).

SCREENSHOT:


8:40
The Buzz Prophet
EXCLUSIVE
CLASS DISMISSED
HE WHO MUST NOT BE NAMED
ACTORS BID ADIEU TO THEIR CHARACTERS
'POTTER' MANIA HITS LONDON
KNOWLEDGE
00:16
00:39

ANCHORS:



Application Icon:

Hardware and software requirements:

Hardware-

ARKit support was coming to all iOS devices that will be getting the iOS 11 update. However, that is not the case as only selected iPhones will be supporting ARKit when iOS 11 drops later this year. This includes all iOS devices that are powered by Apple’s A9 or A10 chip. This means that ARKit will only work on the following devices:

- iPhone SE
- iPhone 6s
- iPhone 6s Plus
- iPhone 7
- iPhone 7 Plus
- iPad Pro (All three variants and models)
- New 9.7-inch iPad (2017)

Software-

Xcode: To Code the Application

SKVideo node: To play the Videos

SpriteKit: is 2D   framework.

Language: swift 4.0

How Our Application Works:

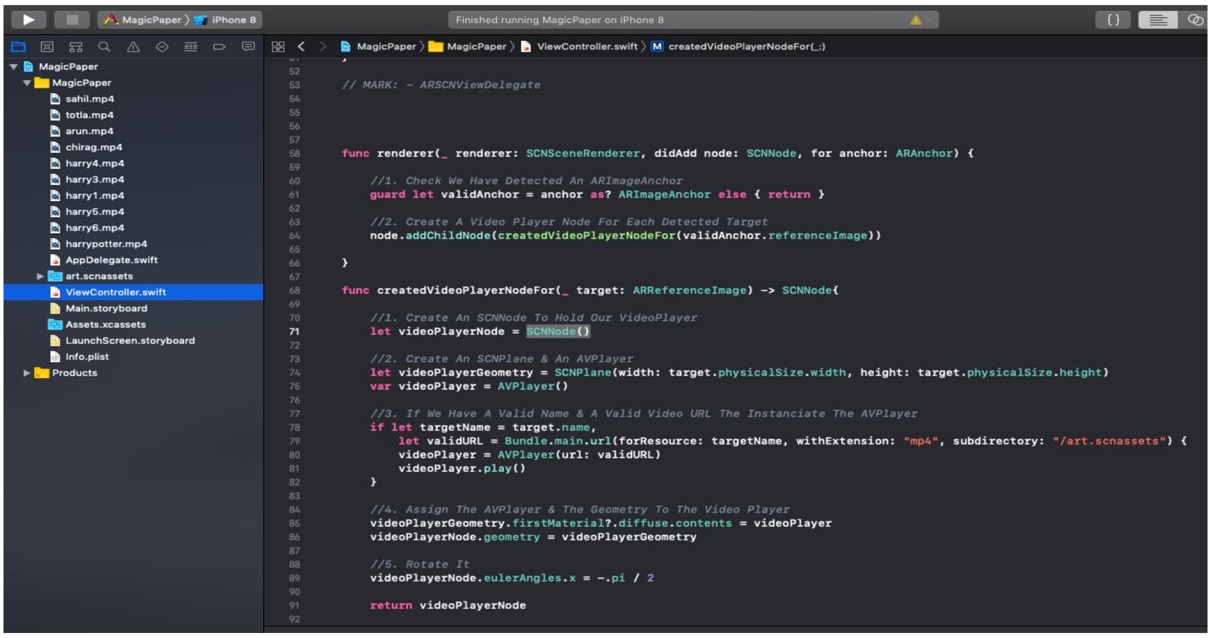
Basically there are two components

Anchor : The Object to be detected. In this Case they are the pages of the Magazine.

NODE : The item that will get displayed on the place of anchor.

In Our Case it is a video File

CODE SNIPPETS:



```swift
    // MARK: - ARSCNViewDelegate



    func renderer(_ renderer: SCNSceneRenderer, didAdd node: SCNNode, for anchor: ARAnchor) {

        //1. Check We Have Detected An ARImageAnchor
        guard let validAnchor = anchor as? ARImageAnchor else { return }

        //2. Create A Video Player Node For Each Detected Target
        node.addChildNode(createdVideoPlayerNodeFor(validAnchor.referenceImage))

    }

    func createdVideoPlayerNodeFor(_ target: ARReferenceImage) -> SCNNode{

        //1. Create An SCNNode To Hold Our VideoPlayer
        let videoPlayerNode = SCNNode()

        //2. Create An SCNPlane & An AVPlayer
        let videoPlayerGeometry = SCNPlane(width: target.physicalSize.width, height: target.physicalSize.height)
        var videoPlayer = AVPlayer()

        //3. If We Have A Valid Name & A Valid Video URL The Instanciate The AVPlayer
        if let targetName = target.name,
            let validURL = Bundle.main.url(forResource: targetName, withExtension: "mp4", subdirectory: "/art.scnassets") {
            videoPlayer = AVPlayer(url: validURL)
            videoPlayer.play()
        }

        //4. Assign The AVPlayer & The Geometry To The Video Player
        videoPlayerGeometry.firstMaterial?.diffuse.contents = videoPlayer
        videoPlayerNode.geometry = videoPlayerGeometry

        //5. Rotate It
        videoPlayerNode.eulerAngles.x = -.pi / 2

        return videoPlayerNode

```

Improvements:-

Augmented reality in newspapers is potentially a very useful feature and could be a great enhancement to them in the future. I enjoy looking at digital content linked to newspaper articles. Based on my experience, however, the technology needs to be improved. I consider the following features to be very important in order for AR in newspapers to be effective.

- The augmented reality feature should be easy to use, reliable, and as foolproof as possible.
- The scanning and content loading process should be rapid. Instant or very nearly instant gratification is necessary in order for the technology to appeal to people.
- The digital content should offer added value to the print article.
- Ideally, the digital content should be obtainable only through the AR program and shouldn't be something that a person could find on the Internet on their own.
- If the content is available on the Internet and can be accessed via a web browser, it's very important that obtaining it through an AR program is a rapid and convenient process.
- Some people feel that the new content should contain movement or interactivity, since otherwise static material in the newspaper is simply being replaced with more static material. I don't mind seeing static digital content as long as it's rich in new information.

CONCLUSION:

The potential for Augmented Reality applications in newspaper sector particular in connection with the press, are enormous. Besides facing the unavoidable challenges AR Applications are chasing whatever is coming in the path.
AR provides significant added value, both on the cultural and on the commercial levels. From analysis what is emerged is that newspaper publishing can work parallel without the need to completely replace them.

These levels of next-generation mobile devices are looking forward with a possibility of adoption for AR solutions, which is a mainstream technology that is now unavoidable.

APPENDIX-SOURCE CODE

SWIFT CODE:

```
import UIKit
import SceneKit
import ARKit

class ViewController: UIViewController, ARSCNViewDelegate {

@IBOutlet var sceneView: ARSCNView!   //usually we use an UI View
but here we are using ARSCN view

override func viewDidLoad() {
super.viewDidLoad()

//Set the view's delegate
sceneView.delegate = self

//Show statistics such as fps and timing information
sceneView.showsStatistics = true

}

override func viewWillAppear(_ animated: Bool) {
super.viewWillAppear(animated)

//Create a session configuration
let configuration = ARImageTrackingConfiguration() // to track the
anchors
```

```
if let trackedImages =
ARReferenceImage.referenceImages(inGroupNamed:
"NewsPaperImages", bundle: Bundle.main)
{ // tell what images are to be tracked where are they to be found

configuration.trackingImages = trackedImages

configuration.maximumNumberOfTrackedImages = 10  // how many
images can be tracked at a single point

}

//Run the view's session
sceneView.session.run(configuration)
}

override func viewWillDisappear(_ animated: Bool) {
super.viewWillDisappear(animated)

//Pause the view's session
sceneView.session.pause()
}

//MARK: - ARSCNViewDelegate




func renderer(_ renderer: SCNSceneRenderer, didAdd node:
SCNNode, for anchor: ARAnchor) {

//1. Check We Have Detected An ARImageAnchor
guard let validAnchor = anchor as? ARImageAnchor else { return }
```

```
//2. Create A Video Player Node For Each Detected Target
node.addChildNode(createdVideoPlayerNodeFor(validAnchor.refere
nceImage))

}

func createdVideoPlayerNodeFor(_ target: ARReferenceImage) ->
SCNNode{

//1. Create An SCNNode To Hold Our VideoPlayer
let videoPlayerNode = SCNNode()

//2. Create An SCNPlane & An AVPlayer
let videoPlayerGeometry = SCNPlane(width:
target.physicalSize.width, height: target.physicalSize.height)
var videoPlayer = AVPlayer()

//3. If We Have A Valid anchor Name & A Valid Video URL The
Instanciate The AVPlayer
if let targetName = target.name,
let validURL = Bundle.main.url(forResource: targetName,
withExtension: "mp4", subdirectory: "/art.scnassets") {
videoPlayer = AVPlayer(url: validURL)
videoPlayer.play()
}

//4. Assign The AVPlayer & The Geometry To The Video Player
videoPlayerGeometry.firstMaterial?.diffuse.contents = videoPlayer
videoPlayerNode.geometry = videoPlayerGeometry

//5. Rotate It by 90 degrees
videoPlayerNode.eulerAngles.x = -.pi / 2

return videoPlayerNode
```

```
}

}
```

XML CODE:

```
<?xml version="1.0" encoding="UTF-8"?>
<document
type="com.apple.InterfaceBuilder3.CocoaTouch.Storyboard.XIB"
version="3.0"                                        toolsVersion="13122.16"
targetRuntime="iOS.CocoaTouch"       propertyAccessControl="none"
useAutolayout="YES"  useTraitCollections="YES"  useSafeAreas="YES"
colorMatched="YES" initialViewController="BV1-FR-VrT">
   <dependencies>
      <plugIn
identifier="com.apple.InterfaceBuilder.IBCocoaTouchPlugin"
version="13104.12"/>
      <capability name="documents saved in the Xcode 8 format"
minToolsVersion="8.0"/>
   </dependencies>
   <scenes>
      <!--View Controller-->
      <scene sceneID="tXr-a1-R10">
         <objects>
            <viewController                                    id="BV1-FR-VrT"
customClass="ViewController"          customModuleProvider="target"
sceneMemberID="viewController">
               <view  key="view"  contentMode="scaleToFill"  id="U0K-
SW-4ec">
                  <rect   key="frame"   x="0.0"   y="0.0"   width="375"
height="667"/>
                  <autoresizingMask                  key="autoresizingMask"
flexibleMaxX="YES" flexibleMaxY="YES"/>
```

```
                <subviews>
                  <arscnView clipsSubviews="YES" multipleTouchEnabled="YES" contentMode="scaleToFill" translatesAutoresizingMaskIntoConstraints="NO" id="BrB-h1-WRS">
                    <rect key="frame" x="0.0" y="0.0" width="375" height="667"/>
                  </arscnView>
                </subviews>
                <color key="backgroundColor" white="0" alpha="1" colorSpace="custom" customColorSpace="genericGamma22GrayColorSpace"/>
                <constraints>
                  <constraint firstItem="BrB-h1-WRS" firstAttribute="leading" secondItem="fQZ-KI-GVf" secondAttribute="leading" id="GsS-dJ-CKf"/>
                  <constraint firstItem="BrB-h1-WRS" firstAttribute="bottom" secondItem="fQZ-KI-GVf" secondAttribute="bottom" id="VpT-BR-CcM"/>
                  <constraint firstItem="BrB-h1-WRS" firstAttribute="trailing" secondItem="fQZ-KI-GVf" secondAttribute="trailing" id="XyZ-9z-H8e"/>
                  <constraint firstItem="BrB-h1-WRS" firstAttribute="top" secondItem="U0K-SW-4ec" secondAttribute="top" id="rJc-2c-zQA"/>
                </constraints>
                <viewLayoutGuide key="safeArea" id="fQZ-KI-GVf"/>
              </view>
              <connections>
                <outlet property="sceneView" destination="BrB-h1-WRS" id="5nT-qQ-ynl"/>
              </connections>
            </viewController>
            <placeholder placeholderIdentifier="IBFirstResponder" id="SZV-WD-TEh" sceneMemberID="firstResponder"/>
          </objects>
```

```
    </scene>
  </scenes>
</document>
```