Academic Journal of Nawroz University (AJNU), Vol.9, No.3, 2020
This is an open access article distributed under the Creative Commons Attribution License
Copyright ©2017. e-ISSN: 2520-789X
https://doi.org/10.25007/ajnu.v9n3a839

Educational Augmented Reality Solar System

Ahmed A. H. Alkurdi
Department of Computer Science, Nawroz University, Duhok, Kurdistan Region – Iraq

ABSTRACT

Augmented reality (AR) has been the main focus for several technology corporations in the past few years. AR is the integration of computer generated objects with the users' environment. The advancement of hardware enables devices to create and place virtual objects in our setting. AR concepts have provided great capabilities for the educational, entertainment and health sector.

People, specifically Primary school children comprehend visual material much more efficiently than imaginary or textual subjects. Augmented reality helps in visualizing and picturing different subjects in classrooms whereas traditionally students would have to imagine the subjects at hand.

ARKit is a powerful framework developed for iOS that enables iOS based devices to provide AR capabilities. This framework supports the placement of computer generated object in to our world in a seamless and effortless manner.

The proposed system is an augmented reality mobile application which illustrates the movement of the solar system. It also incorporates functionality to show specific information about planets for educational purposes. the system is a gateway for developing similar educational applications for school children.

Keywords: Augmented Reality, Solar System, iOS, ARKit, Hit-testing.

1. Introduction

Augmented reality (AR) can be defined as a direct or indirect view of the physical world that is augmented by computer generated three dimension (3D) objects. In comparison to virtual reality (VR), AR can be seen closer to the real world whereas VR is closer to a pure computer generated virtual environment. Augmented reality is debatably the most exciting field of computer science nowadays. Arguably the first appearance of AR started in the 1950's. until 1968, where Sutherland created the first augmented reality application using a see through optical device (Carmigniani & Furht, 2011). While AR technology has been around for quite some time. It has only been achievable to consumer grade devices recently. AR has been made available to users through flash based AR algorithms and developed mobile operating systems such as iOS and Android. Since then, AR is being rapidly incorporated into our audio-visual media and integrated to the daily activities of our lives. Researchers believe that AR can significantly improve the concepts of education. It can provide various benefits in teaching and learning environments. For example, AR can make students more engaged, simulated and motivated to study and explore class materials from different aspects. Also, it can be significantly beneficial for teaching subjects that students cannot possibly gain real-world experience of the subject at hand. Moreover, AR can simulate students' imagination and creativity. And can be easily reformed to suit different learning styles (Yuen, Yaoyuneyong, & Johnson, 2011).

This paper attempts to create an educational AR model application of the solar system for Primary school children. The AR model will help Primary school children learn in a more interactive and fun environment.

A study conducted by S. Begum on two schools in India indicate the importance of interactive education for Primary school children. The study shows a significant improvement in students' comprehension of students who were taught using interactive methods to those who were taught using chalk and talk, i.e. the traditional method. On average students of the interactive teaching sample have scored about 20% better than the students who were taught traditionally (Begum, 2016).

The proposed educational application of the solar system is based on iOS and its latest framework ARKit. iOS is one of the leading mobile operating systems nowadays. ARKit is a framework developed by Apple Inc. for its mobile operating system. The framework enables creating 2D and 3D augmented reality experiences using the devices' camera and motion sensors (Apple Inc., 2018).

This project incorporates different technologies and techniques to achieve the required result. Section two provides an explanation of the different technologies to be used. Section three shows the project implementation which include techniques and code required to achieve the AR models and their motion. Section four, five and six includes results and conclusion and future work respectively.

2. Problem Statement

Augmented reality can add a significant improvement to education. Primary school children used to study their subjects on black and white textbooks. Which is a dull experience even for adults. since then, teaching methods have improved to excite Primary school children and motivate them with different subjects. Nowadays, primary school students study in colored textbooks full of colorful images. Teachers try to engage with them in fun educational activities.

That being said. Augmented reality can help Primary school children learn and get a close to real experience with their materials. For example, watching an airplane in 3D instead of just looking at the pictures. Or in this case, walking around the solar system and being able to see how it's actually aligned. How different planets orbit the sun, their size and facts about the planets. Which can all be taught in a fun and stimulating environment. This will definitely improve their learning experience and make teaching an easier job.

3. ARKIT

ARKit is a mobile platform for augmented reality. This platform has been developed by Apple Inc. in 2017 for iOS based devices. ARKit enables developer to create rich augmented reality content and applications for iPhone and iPad. The platform utilizes the device camera and motion sensors to detect and capture the users' environment and place 2D and 3D objects. ARKit works by combining device motion tracking, camera scene capture, advanced scene processing, and display conveniences to ease the process of creating augmented reality experiences (Apple Inc., 2018). ARKit includes three layers that enable augmented reality in iOS based devices which are tracking, scene understanding and rendering.

3.1 Tracking

Tracking is the most important feature of ARKit. It enables tracking the devices location and orientation in 3D space. Tracking relies on the device's motion sensors to track the device's location, it also relies on the gyroscope within the device to track its orientation as well as visual inertial odometry which are the differences in between camera capture frames that enhance the tracking of location and orientation of the device in the real world (Apple Inc., WWDC 2017, 2017).

3.2 Scene Understanding

Scene understanding is the platforms ability to determine and establish attributes or properties of the physical world around. Scene understanding provides the ability to detect features of the real world. It can detect planes such as a table or the floor, or it can detect vertical surfaces like walls or posters. It also can detect points like edges (Apple Inc., WWDC 2017, 2017).

Another capability of scene understanding is hit-testing.

Basically hit-testing is the ability to find a cross section with the real world so that 3D objects can be placed accurately. This enables placing objects like a vase for example on a table surface or on the ground (Apple Inc., WWDC 2017, 2017).

Lastly, scene understanding enables light estimation, which is an important part of the AR experience that detects and estimates light intensity of the real world so that objects that are placed have the correct amount of light put on them. This makes the 3D objects look and feel more natural to the eye such that no object is too shiny or dim to look fake (Apple Inc., WWDC 2017, 2017).

3.3 Rendering

Rendering is drawing the information and objects that are placed by the device. it is the visualization of the information passed by tracking and scene understanding. Also drawing the models into the physical world. ARKit is supported by any iOS renderer such as SpriteKit, SceneKit and Metal (Apple Inc., WWDC 2017, 2017).

4. Design and Implementation

The proposed augmented reality application is designed and implemented using xcode. Xcode is Apple's official integrated development environment (IDE). The information of the solar system is credit of solarsystem.nasa.gov website. And the textures used to create the different planets within the solar system are credit of www.solarsystemscope.com. The textures and maps used are 2K high quality textures which are mapped based on NASA elevation and imagery data.

The implementation of the application can be divided into different parts, where each part is responsible for a specific functionality. Refer to diagram 1 for project workflow.

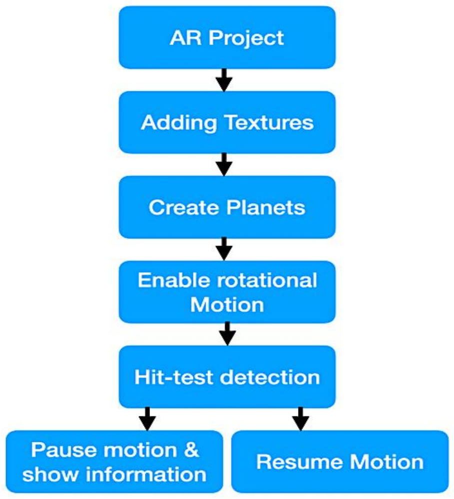


Diagram 1. Project Workflow

4.1 Project

At the beginning, an AR project is created in xcode. AR projects contain a single view AR scene view object that can run the device camera and has the ability to run augmented reality contents.

Then, permission from the user is request through the info.plist file to grant access to the device camera.

Afterwards, a session of AR is run with ARWorldTrackingConfiguration. Which is responsible for establishing a correspondence between the real world and a virtual 3D coordinate space where all contents are placed. ARWorldTrackingConfiguration class tracks the device's location and movement with six degrees of freedom. Namely, the three rotation axis and the three movement axis.

ARWorldTrackingConfiguration has the ability to find, add and track horizontal and vertical planes, 2D image markers, 3D objects.

The session is created and run by creating an instance of ARWorldTrackingConfiguration class and then setting the configuration instance to the scene view session object.

Below is the code responsible for creating the configuration and running the session.

```
// Create a session configuration
```

```
        let configuration =
ARWorldTrackingConfiguration()

        sceneView.debugOptions =
[ARSCNDebugOptions.showFeaturePoints]
        // Run the view's session
        sceneView.session.run(configuration)
```

4.2 Create AR Planets

Our solar system has eight planets, the sun and moon. Thus, a function was created to be called to model, texture and position each planet.

That being said, each planet must be placed at a node in the application 3D space. Nodes are structural element in the scene of the application. Each node represents a position and transform in 3D coordinate space, to which displayable content can be added such as geometry, lights, camera, etc.

The createPlanet function is responsible for creating the geometry of the planets from an SCNSphere object with a different radius for each planet. It is also responsible for setting the texture of each planet. Texture are a combination of diffuse, specular, normal and emission. Which compromise the texture, it's light effect and the elevation levels around planets so that they appear more natural and realistic.

Moreover, the position of each planet is set relative to its position in the solar system.

Note the code below which is responsible for creating a node and calling the createPlanet function.

```
// Create Earth
let earthParent = SCNNode()
earthParent.position = SCNVector3(0,0,-3.5)
earthParent.name = "EARTH"
let earth = createPlanet(geometry: SCNSphere(radius:
0.36), diffuse:  imageLiteral(resourceName: "Earth Texture
Day"), specular:  imageLiteral(resourceName: "Earth
Specular"), emission:  imageLiteral(resourceName: "Earth
Emission"), normal:  imageLiteral(resourceName: "Earth
Normal"), position: SCNVector3(5.5,0,0))
earth.name = "EARTH"
```

Also see the code below for the createPlanet function.

```
func createPlanet(geometry: SCNGeometry, diffuse:
UIImage, specular: UIImage?, emission: UIImage?, normal:
UIImage?, position: SCNVector3) -> SCNNode {
        let planet = SCNNode(geometry: geometry)
planet.geometry?.firstMaterial?.diffuse.contents = diffuse
planet.geometry?.firstMaterial?.specular.contents = specular
planet.geometry?.firstMaterial?.normal.contents = normal
planet.geometry?.firstMaterial?.emission.contents =
emission
planet.position = position
return planet }
```

Afterwards, actions must be added to each planet. An action is an animation that can change the properties of a node that is attached to it. Each planet except the sun must have two actions, one to rotate the planet around its axis including the sun, and another to rotate the planet around the sun. for this a rotation function was defined as shown below.

```
func rotation(time: TimeInterval) -> SCNAction {
        let rotationAction = SCNAction.rotateBy(x: 0, y:
CGFloat(360.degreesToRadians), z: 0, duration: time)
        let forever =
SCNAction.repeatForever(rotationAction)
        return forever }
```

The rotation function was called on each planet with the method runAction and a relative rotation time i.e. speed of rotation relative to the real solar system speeds.

Lastly, each planet was added to the scene view object to be displayed in augmented reality. See figure 1.

Figure 1. AR Planets in a skybox

4.3 Gestures

The class UIGestureRecognizer is responsible for detecting a specific sequence of touches or other input and acting upon them as required. The subclass UITapGestureRecognizer is registered to the scene view and used in the application to enable user tap input. This is done so that users can tap on objects in the scene view. Note below the code required to register and set the gesture recognizer.

```
let tapGestureRecognizer =
UITapGestureRecognizer(target: self, action:
#selector(callNode))
self.sceneView.addGestureRecognizer(tapGestureRecogni
zer)
```

4.4 Hit-testing

hit-testing is the method that searches for objects in the renderer's scene that correspond to a point in the renderer's image. In other words, the renderer renders a 2D image of the augmented reality scene. A point on the rendered 2D image corresponds to any point along a line in the augmented reality scene. That being said, hit-testing is responsible for finding any objects along that line in the scene view.

In the aforementioned project. Hit-testing has been used so that users can tap on a planet in the scene so that they can have an up-close look on the planet and acquire some basic information on the tapped planet.

The process involves getting the scene view location tapped on, then converting that to 3D coordinates. Afterwards, hit-testing the coordinates for available objects. See below.

```
@objc func callNode(sender:UITapGestureRecognizer) {
    let sceneViewTappedOn = sender.view as! SCNView
    let tappedCoordinates = sender.location(in:
sceneViewTappedOn)
    let hitTest =
sceneViewTappedOn.hitTest(tappedCoordinates)
```

4.5 Transactions

SCNTransactions are used to perform animations on a scene graph. Transactions are used to control animatable properties in a scene (Apple Inc., SCNTransaction, 2018). In this project, transactions are used to animate nodes (planets) from their orbits to the camera front and vice versa along with displaying and removing control objects in the device screen. See code below.

```
SCNTransaction.begin()
        SCNTransaction.animationDuration = 0.2
        SCNTransaction.completionBlock = {
            for nodes in
self.sceneView.scene.rootNode.childNodes {
                nodes.isHidden = false
                nodes.isPaused = false
            }
            self.infoTextView.isHidden = true
        }
        let bringForward = CABasicAnimation(keyPath:
"position")
        bringForward.speed = 0.2
        bringForward.fromValue = node.position
        bringForward.toValue = nodePosition
        node.addAnimation(bringForward, forKey:"position")
        SCNTransaction.commit()
```

4.6 Position Animation

The combination of gestures and hit-testing enables detecting a user input and finding the corresponding planet tapped. This provides the ability to call planets forward for close inspection and providing general information about the planets. This is done relying on three functions. callNode, revertNode and animateNode. The callNode function is responsible for hit-testing and finding the corresponding planet. Then a replica of the planet is created and all animation across the system is stopped. The planet replica is called forward and animated to rotate around itself. Moreover, a textview object is used to display general information of the planets in question. See below for the code required to find the corresponding planet from the hit-testing results.

```
if !hitTest.isEmpty {
        let results = hitTest.first!
        let nodeTouched = results.node
        planetName = nodeTouched.name!
        for planetsinf in planetInf {
          if planetsinf.name == nodeTouched.name {
            infoTextView.text = planetsinf.info
            infoTextView.isHidden = false  } }
```

The animateNode function is responsible for the animation of moving the planet from its current position to a position in front of the camera. This involves acquiring the planet position in that frame, getting device location, and calculating the distance required to place the planet since each planet has a different size, they must be placed accordingly. To place each planet correctly in front of the camera, a 4*4 vector matrix is used to be multiplied by the position of the planet, which is also stored as a matrix. This will provide the required animation transform so that the planets can be position accordingly.

```
func animateNode(node:SCNNode) {
    let geometry = node.geometry as! SCNSphere
    if geometry.radius < 1 {
      guard let currentFrame =
sceneView.session.currentFrame else {
            return
        }
        let bringForward = CABasicAnimation(keyPath:
"position")
        bringForward.speed = 0.2
        bringForward.fromValue = node.position
        var toVal = matrix_identity_float4x4
        toVal.columns.3.z = (Float)(geometry.radius * -4)
        let frontViewM =
matrix_multiply(currentFrame.camera.transform, toVal)
        bringForward.toValue =
SCNVector3(frontViewM.columns.3.x,
                          frontViewM.columns.3.y,
                          frontViewM.columns.3.z -
(Float)(geometry.radius * 4))
        node.addAnimation(bringForward, forKey:
"position")
        node.position =
SCNVector3(frontViewM.columns.3.x,
                      frontViewM.columns.3.y,
                      frontViewM.columns.3.z -
(Float)(geometry.radius * 4))
      }
```

Finally, the revertNode function is accountable for returning the node to their original position and restarting the system from where it left off by tapping.

```
let bringForward = CABasicAnimation(keyPath: "position")
     bringForward.speed = 0.2
     bringForward.fromValue = node.position
     bringForward.toValue = nodePosition
     node.addAnimation(bringForward, forKey: "position")
```

See figure 2 for a view of the earth for example when tapped.

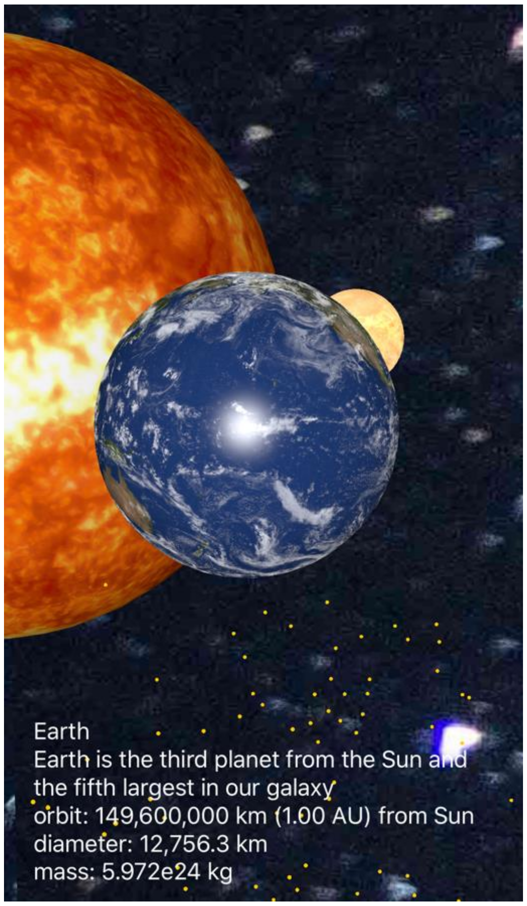

Earth
Earth is the third planet from the Sun and the fifth largest in our galaxy
orbit: 149,600,000 km (1.00 AU) from Sun
diameter: 12,756.3 km
mass: 5.972e24 kg

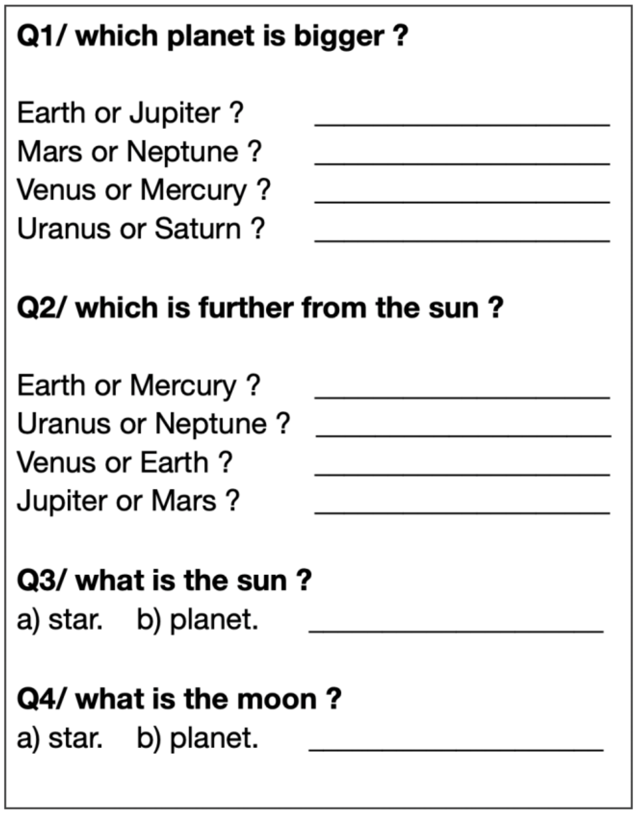
Figure 2. Close up look at earth.

5. Results

Augmented reality has a staggering effect on the learning process when implemented into education. It can dramatically increase comprehension and understanding in record times.

This project has been tested on a group of 10 children with no knowledge of the solar system. These were separated into two groups. Group (a) were taught in the conventional method, with textbooks and oral explanation only, while group (b) were taught using said application with oral explanation. then a questionnaire was presented to them to test their understanding, see figure 3. The results of implementing such a system were evident. The children of group (b) had a much better understanding and have enjoyed the experience much more than group (a). group (b) were also more informed, they averaged 8/10 questions, while group (a) only averaged 5/10.

This shows the importance of implementing such a technology for a better educational system in the future.


Q1/ which planet is bigger ?

Earth or Jupiter ? ____________________
Mars or Neptune ? ____________________
Venus or Mercury ? ____________________
Uranus or Saturn ? ____________________

Q2/ which is further from the sun ?

Earth or Mercury ? ____________________
Uranus or Neptune ? ____________________
Venus or Earth ? ____________________
Jupiter or Mars ? ____________________

Q3/ what is the sun ?
a) star. b) planet. ____________________

Q4/ what is the moon ?
a) star. b) planet. ____________________

Figure 3. Questionnaire sample.

7. Conclusion

In conclusion, technology has significantly improved over the past decade. Hardware capabilities have significantly increased, enabling technologies like virtual reality and augmented reality to be placed in consumer grade devices.

This technological improvement must be used in all aspects of life. Specially in education. In this project, the AR solar system model helps Primary school children get an almost realistic feel of the planets and our sun. ensuring eagerness and enthusiasm in learning experience and helping them be more involved and focused. This approach must be adopted in large scale in schools around the world.

8. Future work

This system is intended to show the movement of the solar system and include some information about our planets. Thus, it mainly achieves its purpose. However, there is endless opportunity and room for developing such systems based on the educational scheme. Most books and subjects can be brought to life using AR technology, this project is a doorway to the large process of visualizing all future books and enabling future generations with the ability of studying in a visually rich environment.

9. Bibliography

1. Apple Inc. (2017, 6). *WWDC 2017*. Retrieved from Apple Developer: https://developer.apple.com/videos/wwdc2017/
2. Apple Inc. (2018). *Apple Developer*. Retrieved from Apple: https://developer.apple.com/documentation/arkit
3. Apple Inc. (2018). *SCNTransaction*. Retrieved from Apple Developer: https://developer.apple.com/documentation/scenekit/scntransaction
4. Begum, S. (2016). TRADITIONAL METHOD OF TEACHING VS MODERN METHOD OF TEACHING- A COMPARATIVE ANALYSIS AT BANGALORE PRIVATE SCHOOLS. *International Education & Research Journal [IERJ]*, 31-35.
5. Carmigniani, J., & Furht, B. (2011). Augmented reality: an overview. In J. Carmigniani, & B. Furht, *Handbook of augmented reality* (pp. pp. 3-46). New York: Springer.
6. Yuen, S. C.-Y., Yaoyuneyong, G., & Johnson, E. (2011). Augmented Reality: An Overview and Five Directions for AR in Education. *Journal of Educational Technology Development and Exchange (JETDE) 4(1)*, 119-140.