Towards partial fulfillment for Undergraduate Degree Level Programme
Bachelor of Technology in Computer Engineering

A Project Report on:

AUGMENTED REALITY

Prepared by :

Admission No.	Student Name
U15CO002	DHRUMIL JOSHI
U15CO069	SNEHAL SHAEVYA
U15CO070	ARCHIT SHARMA
U15CO072	PUSHKAR KUMAR

Class : B.TECH. IV (Computer Engineering) 8th Semester

Year : 2018-2019

Guided by : Dr. K.N. Jariwala
Mr. S.R. Goyal

DEPARTMENT OF COMPUTER ENGINEERING
SARDAR VALLABHBHAI NATIONAL INSTITUTE OF TECHNOLOGY,
SURAT - 395 007 (GUJARAT, INDIA)

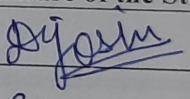
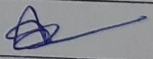
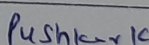
Student Declaration

This is to certify that the work described in this project report has been actually carried out and implemented by our project team consisting of

Sr.	Admission No.	Student Name
1	U15CO002	Dhrumil Joshi
2	U15CO069	Snehal Shaevya
3	U15CO070	Archit Sharma
4	U15CO072	Pushkar Kumar

Neither the source code there in, nor the content of the project report have been copied or downloaded from any other source. We understand that our result grades would be revoked if later it is found to be so.

Signature of the Students:

Sr.	Student Name	Signature of the Student
1	Dhrumil Joshi	
2	Snehal Shaevya	
3	Archit Sharma	
4	Pushkar Kumar	

Certificate

This is to certify that the project report entitled Augmented Reality is prepared and presented by

Sr.	Admission No.	Student Name
1	U15CO002	Dhrumil Joshi
2	U15CO069	Snehal Shaevya
3	U15CO070	Archit Sharma
4	U15CO072	Pushkar Kumar

Final Year of Computer Engineering and their work is satisfactory.

SIGNATURE :

14/3
GUIDE

14/3
JURY

HEAD OF DEPT.

ABSTRACT

The emergence of virtual reality and augmented reality is constantly providing new ways of interaction with digital information.However, virtual reality indulges the users in simulated world or artificial world with very little to no connection with the real world.Augmented reality adds or integrates virtual objects to real ones. It refers to computer technology that adds information to a user's sensory perceptions. AR research majorly focus on see-through devices, such as electronic head gears,cell phones.It projects graphics over the world environment in real time.This interface minimizes the effort that a user has to put when switching his/her attention between real tasks and a computer screen. In AR, the user's point of view and the computer interface gets integrated together.

The AR shop is an interactive shop where users can see the live demonstration of the object that they wish to purchase,choosing amongst a number of different kind of objects by looking through their phone, enabling easier accessibility and a better understandability of the actual worth of the object that the user wishes to purchase.

Table of Contents

List of Figures

Abbreviations

JDK Java Development Kit

AR Augmented Reality

VR Virtual Reality

IDE Integerated Development Environment

SLAM Simultaneous localization and mapping

PTAM Parallel Tracking and mapping

RANSAC Random Sampling Consensus

Chapter 1

Introduction

1.1 Motivation

Augmented Reality is an innovation in the field of development which superimposes virtual objects into reality by means of devices such as smart phones, optical devices, tablet screen,etc. It makes excellent use of geolocation such as GPS data of a user's device.This data helps techonology to accumulate information about user's immediate surroundings, so as to superimpose information more personally.AR offers an interactive and dynamic way to experience and imagine new things. It has a high potential in many applications for businesses and brands. One such application is AR shop or Augmented Reality shop. In today's world, online shoping is an ongoing and rapidly growing trend. But often it happens that we buy something looking at static pictures and images of the product only to later realize that the product wasn't worth the buy.AR shop has the capability to overcome this by allowing users to have a live demo of the object they are going to buy. This can be done by adding virtual objects (products) to the real environment of the user. AR shop can solve a major real world problem effectively and hence we are highly motivated to work and contribute in this field.

1.2 Objectives

To obtain a fully functional augmented reality application that is user interactive. AR shop is the application that shall provide an online shopping environment for convinient user experince using virtualization of products to be bought in the real world. The application will simply place a product into the real world environment and change the position or scale based on tracking user's immediate environment and sensing the depth of the physical world.The application shall help people to visualize the product in their home before they actually purchase the product. The user can select the product from product catalog, insert/move product within a photo or a video and customize the products based on availabe options. Interaction with digital objects,personalization and customization, and AR visualization from any angle are the key features of this app.This fully functional app is going to use Unity 3D development IDE along with Wikitude engine for making AR based android AR shop application. The main objective of this application is to engage end users.It would allow the customers to make a more informed purchase decision.When this objective is achieved, the business growth from retailers point of view will also be significant.Retailers would have a high profit rate and the rate of return of products from the users will also decrease.Also it would save time for the customers and accuracy related to purchase will be more.

1.3 Contribution

A detailed study on augmented reality and how it is useful to the society by making an application which eases user's choice making for buying online stuff. AR shop places a 3D scaled product into environment and changes the position and scale based on tracking of the environment and sensing the depth of the physical world. The shopping app would give the users more information about the worth of the product by dynamic demonstration of the product onto live environment.Using Unity 3D IDE and wikitude engine, an AR based android application is to be developed.The AR technology works for shopping as follows:-

a. *Content Management* done with the help of data provided and also with the help of catalogue.

b. *Model*-It refers to preparing scaled virtual 3D objects.

c. *Service Adaption* - Refers to desigining of user experience and intergrating it with the app.

d. *Engagement* -It refers to launching the app and view the product in the immediate environment. Rotate it, swap colors and also replace it.

1.4 Applications

Benefits and applications of AR shop are as follows:-

Online Retailers Oriented:

a. Reduce the rate of returns of the delivered product as the AR shop app provides a better idea about the product.

b. Improve profitability

End User Oriented:

a. Save time of the users by not involving them into returning the product and then reordering.

b. Save money of the users as there might be some penalty when a user returns an item.

c. Accuracy and precision related to purchase.

Chapter 2

Theoritical Background and Literature Survey

2.1 What is Augmented Reality?

"Augmented Reality" blurs the line of distinction between the two environments "The Real" and "The Virtual". Supplementing the human perception,It merges the real world with the virtual.
In comparision with virtual reality, which creates immersible and computer-generated environments, Augmented reality is closer to the natural or real world. Augmented reality further adds graphics, sounds, haptics, smell and tactile quality to the real world. With augmented-reality displays, informative graphics appear in your field of view, and audio coincides with what you see. Hence the display looks much like a normal pair of glasses. These enhancements are auto-refreshed in continuation to reflect the movement of your head.

Augmented reality truly changes the way we view and perceive the real world.In this section we will explore the scope of this "middle path" between real world and virtual world, its components and how it will be used.

One of the most powerful uses of virtual world isn't to replace the real world, rather augment the user's view of the real world with additional information. This idea, introduced by Ivan Sutherland's pioneering work on head-mounted displays, is referred to as *augmented reality*.
It is necessary to make a distinction between the concept of *real* and *virtual*.

The operational definitions that we adopt are as follows:

a. A Real object is an object that has an actual objective existence.

b. A Virtual object is an object that exists in essence or effect, but has no formal or actual existence.

Fundamentally, Augmented Reality is about augmentation or enhancement of human perception.It supplies information that is not otherwise detectable by human senses. The augmented reality presented to a user enhances that user's performance in the world and perception of the world.
An augmented reality system generates and provides a composite view for the user using the system. It combines the real scene viewed by the user and a virtual scene generated by the computer.Hence augmenting and enhancing the scene with additional user interpretable information.

For example, graphics and text overlaid on the surrounding world explains how to operate, maintain, and repair equipment, without requirement of a separate user reference to a separate paper or manual. Similarly, participants in a business meeting can interact with a dynamic shared financial or organizational model in 3D, selectively aided with each user's personal (and also private) annotations.However, Generating such material by hand, will require a tremendous amount of expertise and effort, far greater than that currently needed to design "hand-crafted" hypermedia and multimedia presentations. [1]

Figure 2.1: AR displays overlaying computer generated graphics onto real world [4]

2.2 Difference between Augmented Reality (AR) and Virtual Reality (VR)

Virtual Reality strives to immerse a user inside an imaginary, computer-generated ”virtual world”.In its different technologies, the user is cut off from any view of the real world giving the viewer an absolute virtual view.Though it's potential is at least as much as Virtual Reality, very less attention has been paid to Augmented Reality. In Augmented Reality, the user can see the real world around him/her, with graphics superimposed with the real world. In place of replacing the real world, we supplement and enhance it.Hence it seems to the user that the real and virtual objects coexisted.
A visible difference between the two systems is immersiveness. Virtual reality has a totally immersive environment. The visual senses are under the control of the system.Whereas, an augmented reality system is augmenting the real world scene necessitating that the user maintains a sense of presence in that world and the new view is a superset of the real view that the user experiences. The virtual objects are merged with the real view to obtain the augmented display.
Milgram [2] describes a taxonomy that identifies how augmented reality and virtual reality work are related. He defines the Reality-Virtuality continuum shown as Figure 2.2.

The real world and a totally virtual environment are at the two ends of this continuum with the middle region called Mixed Reality. Augmented reality lies near the real world end of the line with the predominate perception being the real world augmented by computer generated data. Augmented reality does not simply mean the superimposition of a graphic object over a real world scene. This is technically an easy task. One difficulty in augmenting reality, as defined here, is the need to maintain accurate registration of the virtual objects with the real world image.

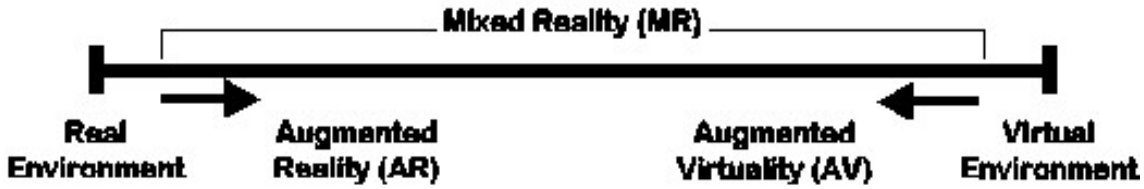


Figure 2.2: Miligram's Reality [1]

2.3 Working of AR System

A virtual reality system completely immerses the user in a computer generated environment. This computer generated environment is maintained by the system in a frame of reference registered with the graphic system creating the rendering of the virtual world. For effective immersion, the frame of reference maintained by the user's body and brain must be registered with the virtual world reference.Hence it is required that the motions or changes made by the user should result in appropriate changes in the perceived world.Also when the user is viewing a virtual world there is no connection between these two reference frames and a connection must be created.

An augmented reality system is considered as the ultimate immersive system. The user can't become more immersed in the real world as augmented view is an addition to the real view of the user. The task is to register virtual frame of reference with what the user is able to see and perceive. Registration is more critical in an augmented reality system as we are much more sensitive to visual mis alignments than to the type of vision-kinesthetic errors that may result in a virtual reality system. Figure 2.3 shows the reference frames that are and must be related in an AR system. [1]

2.3.1 AR components, working and their interrelation

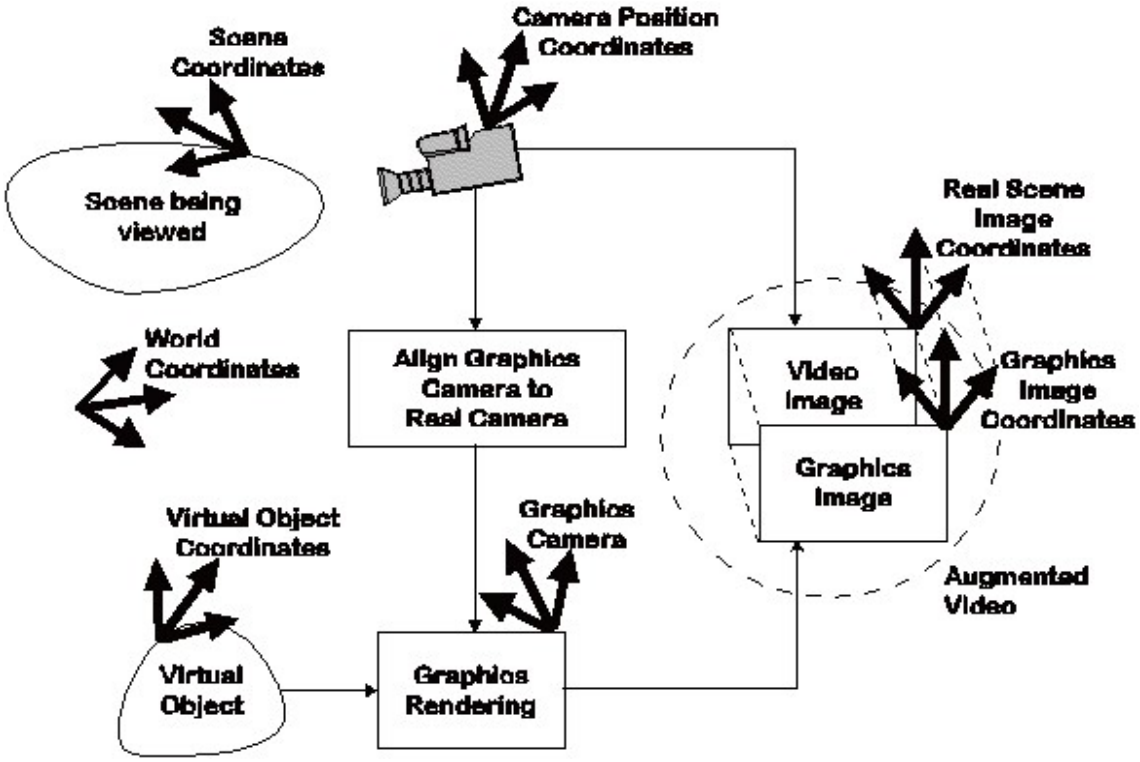


Figure 2.3: AR system components [1]

The scene is viewed by an imaging device(camera in this case).It(camera) performs a perspective projection of the 3D world onto a 2D image plane. The intrinsic (focal length and lens distortion) and extrinsic (position and pose) parameters of the device determine image plane projection.Standard Computer graphics system generates the virtual image.The graphics system requires information about the imaging of the real scene for accurate and precise rendering these objects. This data controls the synthetic camera used to generate the virtual object image which is merged with real scene image to form the AR image. [1]

2.3.2 Performance Issues in AR System

AR systems are expected to be precise and accurate to run in real-time so that the user can move freely within the scene at the same time also to able to see a properly rendered augmented image. This creates two performance criteria on AR system. They are :

a. Update rate for augmented image generation

b. Criteria of accuracy and precision of registration of the virtual and real image.

c. Issues may arise in case of peak use by the users as advanced techniques for handling large traffic have not been introduced since this is still a new area of discovery.

d. Performance can be largely hindered depending upon the amount of physical memory of the server and client. Less physical memory leads to lesser utilization of the virtual memory to be used. This can be improved by allocating more resources to the underyling database which depends on the configuartion of machine.

e. Multithreaded server functionality allows efficient usage of DBMS as settings of numerous threads of AR system server rapidly increases scalibility of the device. If there is no multithreaded server, each client's request is passed via a single connection to the database which cannot be scaled.

2.4 Display Technologies in Augmented Reality

The combination of real and virtual images has technical challenges for system designers. The term monitor-based (non-immersive) or "window-on-the-world" (WoW), AR is used to refer to display systems where computer generated images are analogically or digitally overlaid onto live or stored video images.Figure 2.4 shows monitor based Augmented Reality.Though the technology for achieveing this has been well known,mostly by means of chroma keying, a heavy quantity of useful and critical applications present themselves when this concept is *stereoscopically* implemented.Other display technologies are required to increase the sense of presence. [1]

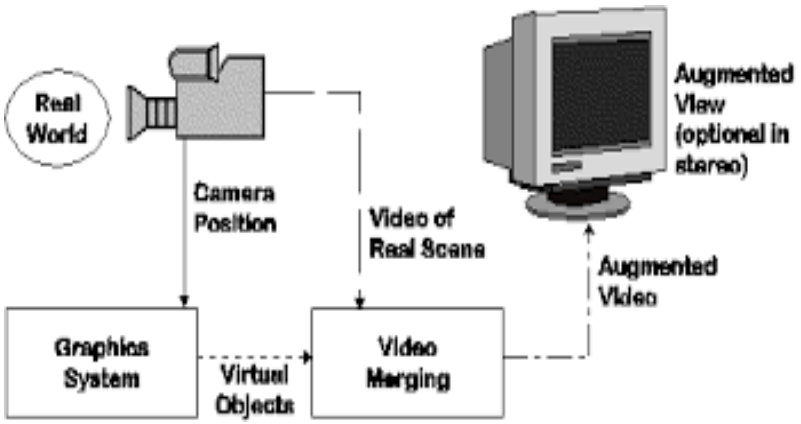


Figure 2.4: Monitor Based Augmented Reality [1]

Head-mounted displays (HMD) are widely used in VR systems. AR researchers have been working with two types of HMD. They are video see-through and optical see-through. The ”see-through” designation comes from the need for the user to be able to see the real world view that is in front of him even while wearing the HMD. The standard HMD used in virtual reality work completely isolates the user from the surrounding environment. As the display is visually isolating the system must use video cameras that are aligned with the display to obtain the real world view. A diagram of a video see-through system is shown in Figure 2.5. [1]

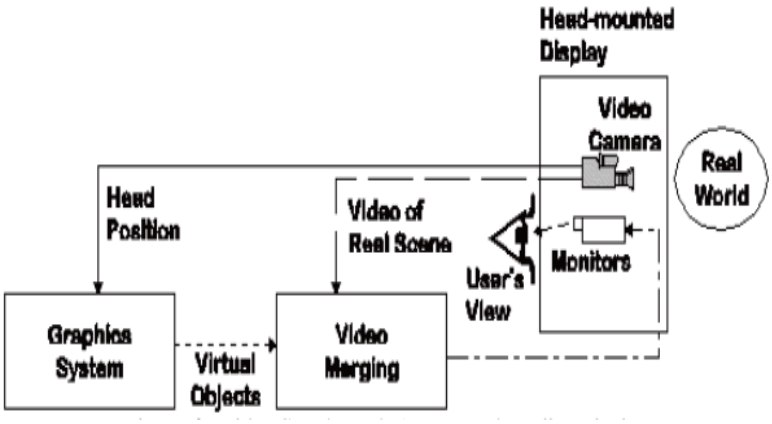


Figure 2.5: Video-see through AR display [1]

The optical see-through HMD eliminates the video channel that looks at the real scene. Instead, the merging of real world and virtual augmentation is optically done in front of the user,as shown in Figure 2.6. This technology is similar to heads up displays (HUD) that appears in military airplane cockpits and recently some experimental automobiles also. In this case, the two images are optically merged on the head mounted display, rather than the cockpit window or auto windshield. [1]

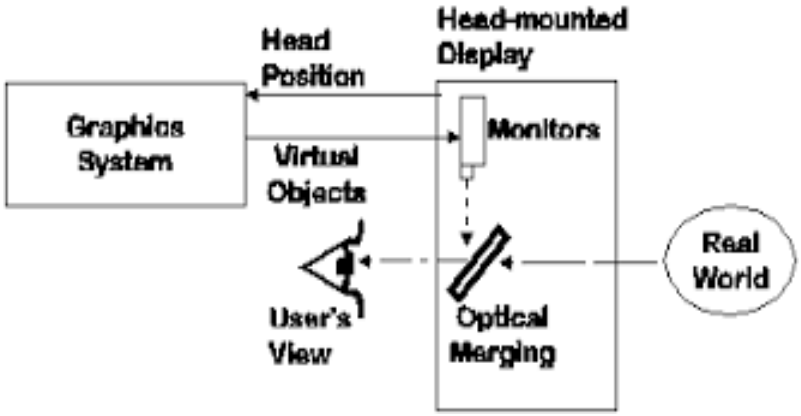


Figure 2.6: Optical-see through AR display [1]

2.5 Wikitude

Wikitude is a technology provider who largely focuses on Augmented Reality.Wikitude initially focused on providing location-based augmented reality experiences through the Wikitude World Browser App. In 2012, the company restructured its proposition by launching the Wikitude SDK, a development framework utilizing image recognition and tracking, and geolocation technologies. The primary product of company is Wikitude SDK, launched in October 2008,new changes included 3D model rendering, video overlay,image recognition and tracking, location based AR.With the invention of SLAM, new technologies like object recognition and tracking its position,markerless instant tracking were also added.The SDK works on Android,Windows and iOS operating systems and optimisation is currently being done for eyewear devices.It was the first app which was released for public that used a location-based approach to augmented reality.

2.5.1 Working of Wikitude

Location based augmented reality wikitude at first entered the market with its geo area AR app. The fundamental rule was, the situation of articles on the screen of the cell phone is determined utilizing the client's situation (by GPS or Wifi), the course in which the client is confronting (by utilizing the compass) and accelerometer. Augmentations can be placed at specific points of interest and afterwards viewed through the devices' screen or lenses.

One of the best known examples of Geo based AR is a game named Pokemon Go. Image Recognition Image recognition techologies have also been included in order to trigger augmented reality technology within the app.

Important features of the target image or marker. This allows to overlay and stick augmentations in fixed position on top or around the image.

SLAM: Instant Tracking, Object and Scene Recognition

In 2017 Wikitude launched its SLAM technology. Instant Tracking, the first feature using SLAM,removing the mandatory requirement for target image,it allowed developers to easily map environments and display augmented reality content. Object Recognition is the latest addition based on SLAM, with the launch of SDK 7. The fundamental difference between Object Recognition and Tracking is that instead of recognizing images and planar surfaces, the Object Tracker can work with three-dimensional structures and objects (tools, toys, machinery. . .).

2.6 Simultaneous localization and mapping

Simultaneous localization and mapping,also known as SLAM is the core component and logic of Augmented reality products,it is the computational problem of constructing or updating a map of an unknown environment while simultaneously keeping track of an agent's location within it. For this problem,several algorithms have been suggested,of which includes particle filter, extended Kalman filter, and GraphSLAM. This technology plays a huge role in latest products like self-driving cars, unmanned aerial vehicles, autonomous

underwater vehicles and even inside the human body. [5]

2.6.1 Problem definition

Given a series of sensor observations o_t, over discrete time steps [5] $\boldsymbol{t}$,and a map of the environment m_t.All quantities are usually probabilistic, so the objective is to compute:

$$P(m_t, x_t | o_t)$$

Applying Bayes' rule gives a framework for sequentially updating the location posteriors, given a map and a transition function

:

$$P(x_t, o_t | m_t) = \sum_{m_t - 1}^{\infty} P(o_t | x_t, m_t) \sum_{x_t - 1}^{\infty} P(x_t | x_t - 1) P(x_t - 1 | m_t, o_t - 1) / Z$$

EM algorithm can be used for solving inference problems.

2.6.2 Parallel Tracking and Mapping for Small AR Workspaces

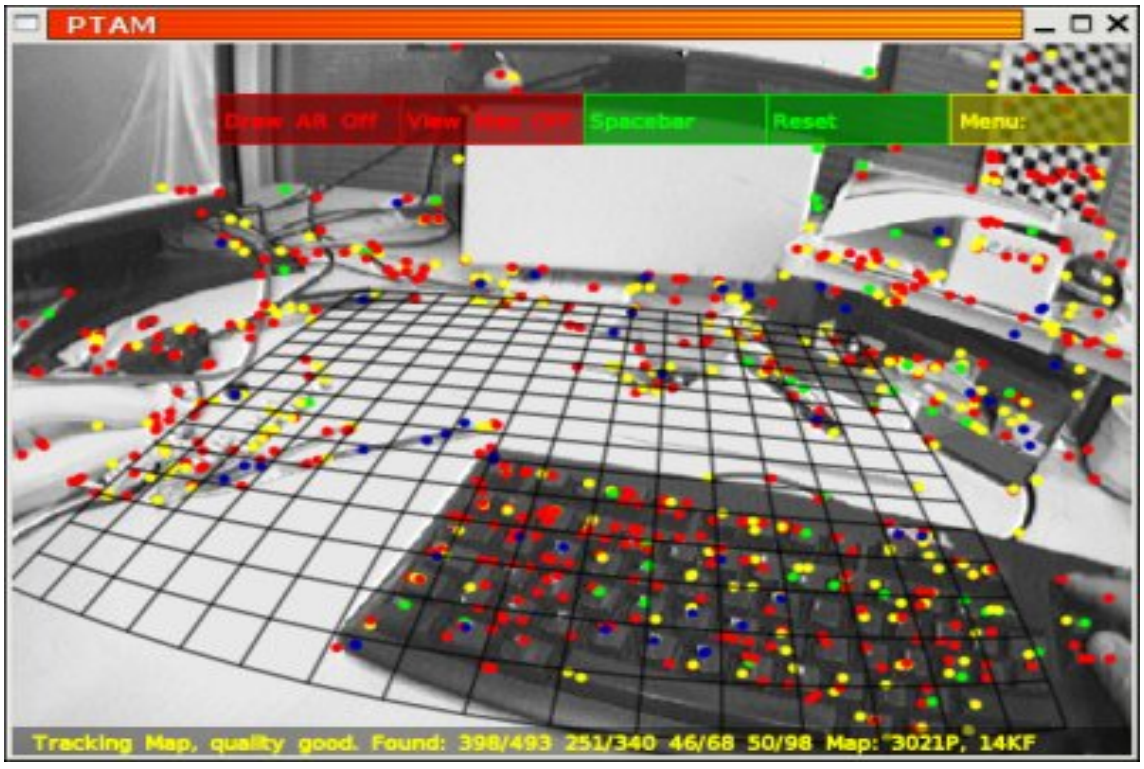


Figure 2.7: PTAM [6]

PTAM (Parallel Tracking and Mapping) is a camera tracking system for augmented reality. It requires no markers, pre-made maps, known templates, or inertial sensors.This software is aimed at AR/vision/SLAM researchers! It does not come with pre-made AR games like “Ewok Rampage.” [6] Topological maps are a method of environment representation which capture the connectivity (i.e., topology) of the environment rather than creating a geometrically accurate map. Topological SLAM approaches have been used to enforce global consistency in metric SLAM algorithms. In contrast, grid maps use arrays (typically square or hexagonal) of discretized cells to represent a topological world, and make inferences about which cells are occupied. Typically the cells are assumed to be statistically independent in order to simplify computation.

Chapter 3

Proposed Algorithm/Work/Application

3.1 Using the Device Tracker in Unity

Interactive AR/VR experiences are provided by Vuforia engine in Unity through by using the Rotational and Positional Device Trackers. Coordinates relative to the world position are described by this Device Tracker and provide rotational/positional coordinates. The Rotational Device Tracker leverages 3 degrees of freedom for optimal AR/VR experience. The Positional Device Tracker is great for AR applications that need to put content in the environment or want to provide additional robustness to Target tracking with 6 degrees-of-freedom.

3.2 Working with Vuforia in Unity to create image target and image target database

Vuforia is an AR based SDK (Software Development Kit) that is used in making AR apps. Vuforia uses computer vision's principles to identify planar images like boxes,chairs ,etc helping in positioning and orientation of objects. The following subsections will provide an idea about the fundamentals of Vuforia.

3.2.1 Creating an Image Target

An image target is required for the camera to recognize a reference and also track it. Orientation and actual size of the targeted image has a direct effect on the same attributes of the superimposed images.
Any image can be assigned as a target image keeping in mind that the features of the target image determines how well the target is tracked. In this tutorial, we will use an online tool to generate feature-rich target images.Augmented Reality Marker Generator online tool generates an image target, and saves the image on the computer. [3]

3.2.2 Image Target

Go to the Vuforia Developer Portal,log in to your account.You will see the Target Manager, under the Develop tab. First add a database. Use the designated button to add a database.
Name your database, and select Device as the type. [3]

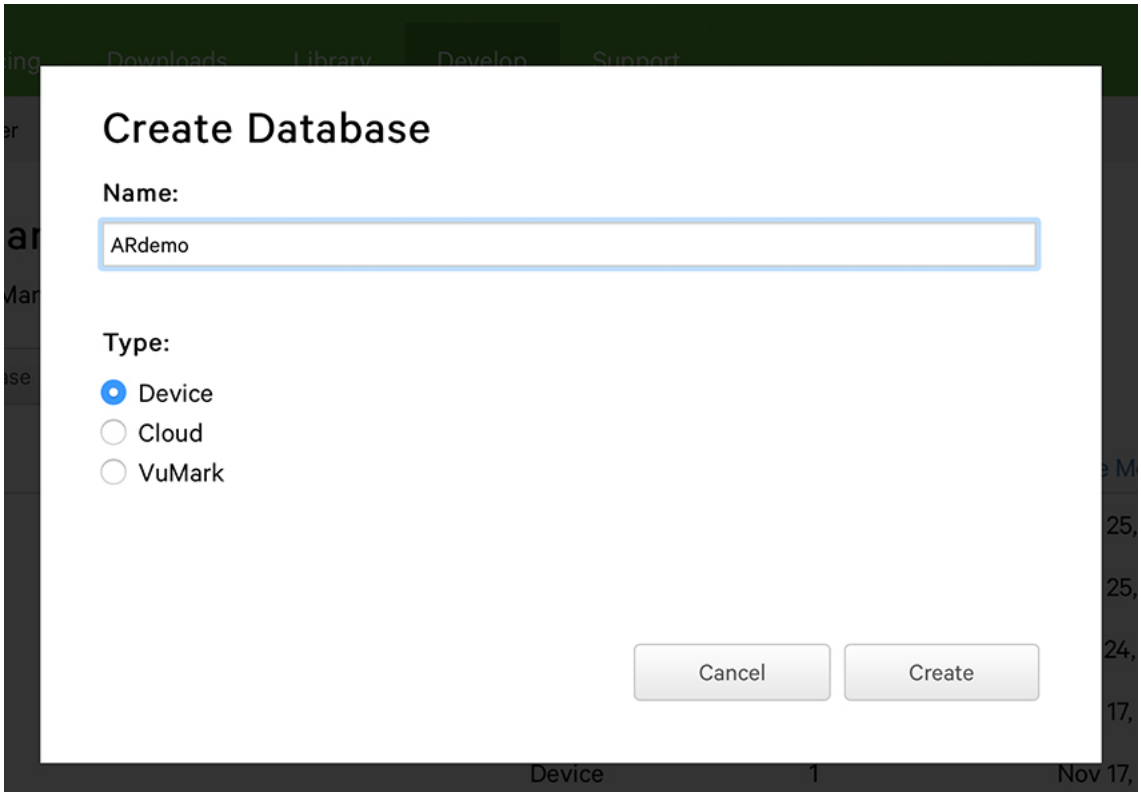


Figure 3.1: Create database [3]

We can now add a target in this database. Click on the Add Target button, and the following window shall appear in front of you.In our case select single image as type. Select the image generated target using the online tool. If you face trouble uploading the file, convert it to .jpg file format and upload it again. [3]

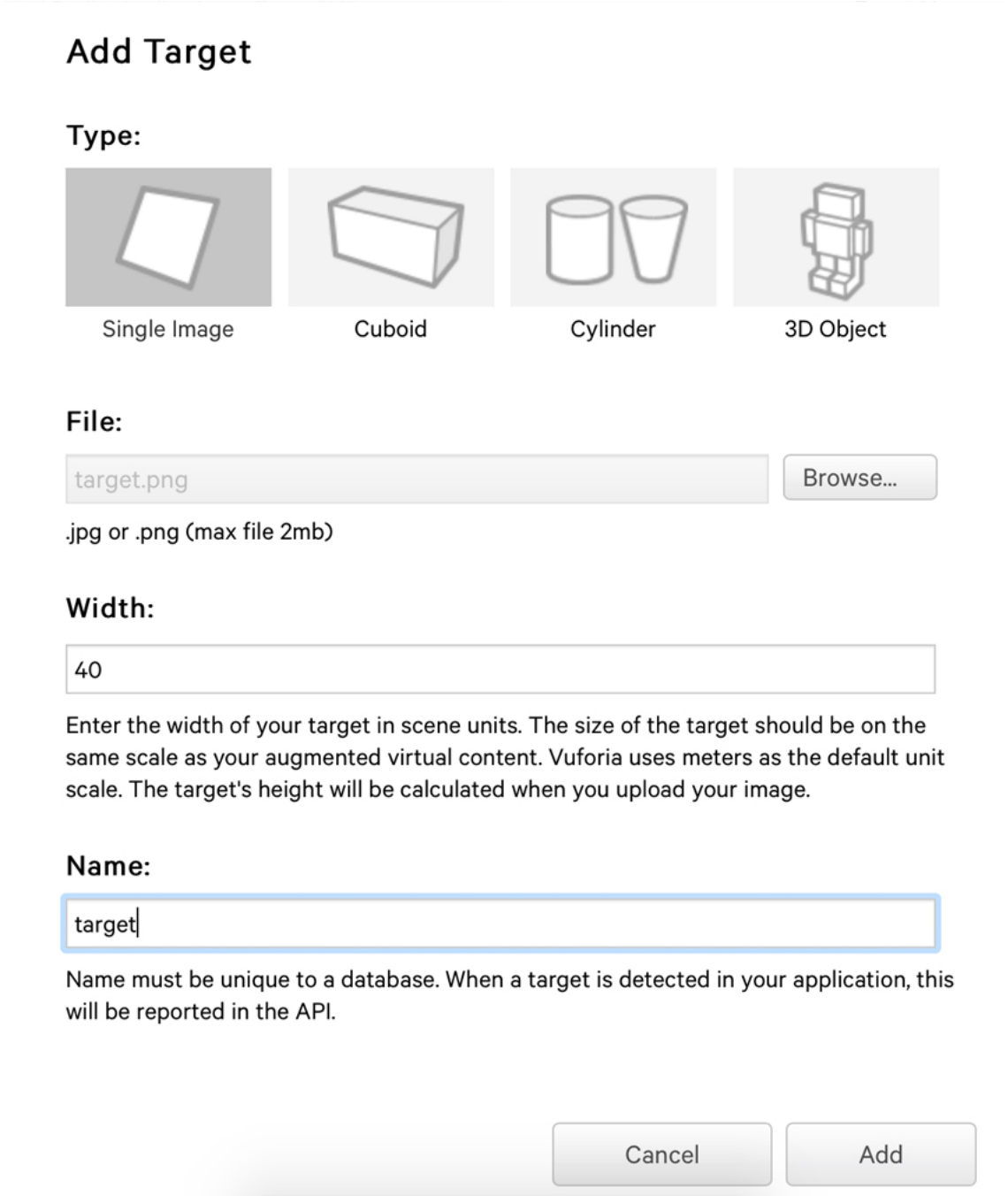


Figure 3.2: Add a target in database [3]

Crucial parameter here is width. This should match the real size of the target

image that you will be finally printing on paper. Set the width to 40. There is no unit since it matched the unit of your scene.
Once you add the target into your database, Vuforia rates your target. With the target image generator we used, features are high and therefore it gets 5 stars, which means it's easy for Vuforia to recognize and track this target.

3.2.3 ARCamera Object

Add the ARCamera object of Vuforia to our scene. To do so, follow the 1) Assets –2)Vuforia –3) Prefabs directory and add the ARCamera object by dragging and dropping to the scene.
Select the ARCamera object and under the Inspector tab, you will observe the App License Key section. This license key shall be acquired from the Vuforia developer portal.
You need to log in to your Vuforia account on the Developer Portal and under the Develop tab, you will find the License Manager section. Add License Key button should be clicked next. On the following page, select the project type as development and define an application name for your project. Name can be altered later on. [3]

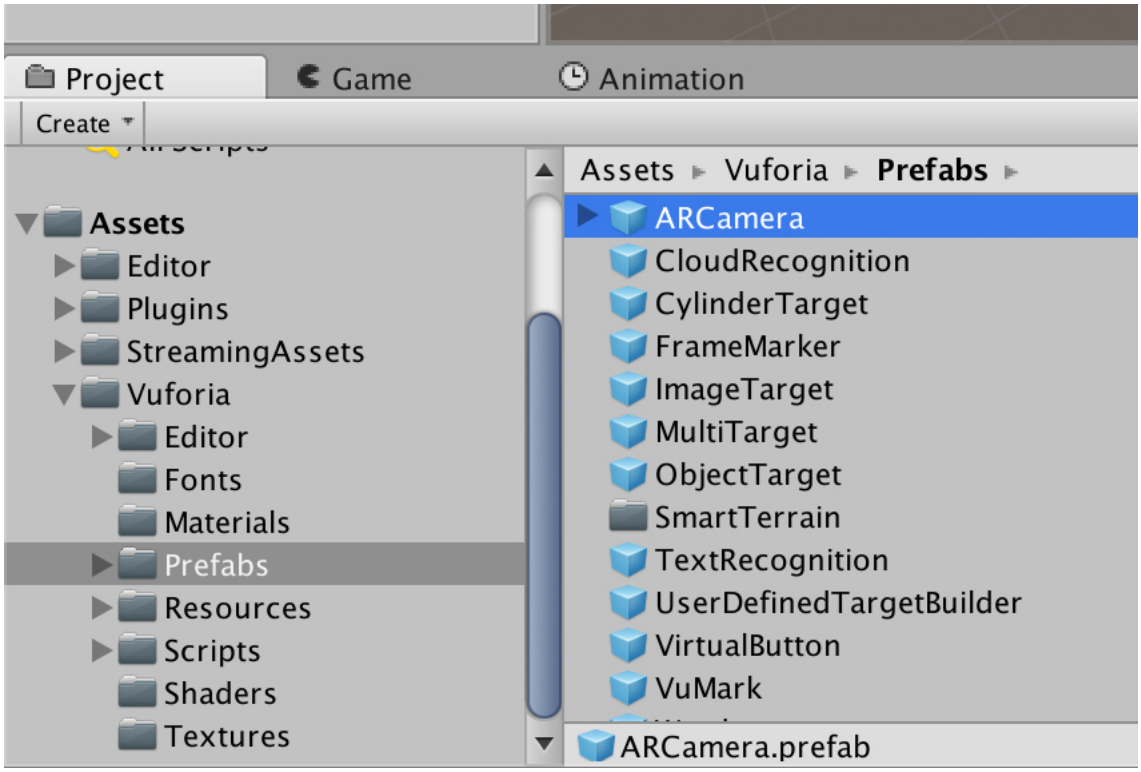


Figure 3.3: Drag and drop ARCamera to the scene [3]

3.2.4 Augmented Graphics

Next is to create the graphics, tie them to the image target. You can either create a GameObject or import your own 3D model into Unity and then use it. In this tutorial we wiill be using a simple 3D cube object for simplicity.

Set it's x, y and z parameters for the Scale option under Transform to 40, so that it matches the size of generated image target. If you set another width value for your image target when generating it in the developer portal, use the value that you selected so as to match the full size of the image target. The last step for getting our AR app working is to set the object as the child of the image target. To do so, under the hierarchy menu, drag the object and drop it on the imageTarget object.

Hit the Play button to run your application. It shall use your capturing device. Get the target image printed or open it from your phone so that Vuforia detects it through your capturing device. [3]

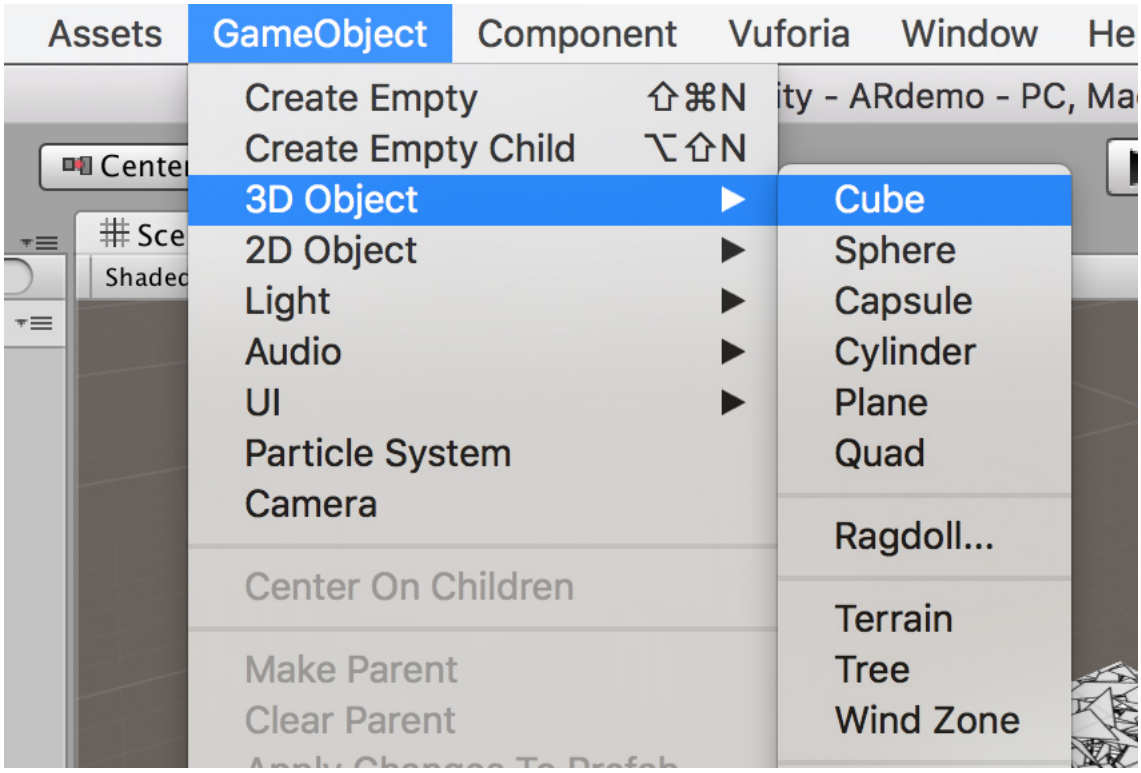


Figure 3.4: Steps to create game object:a [3]

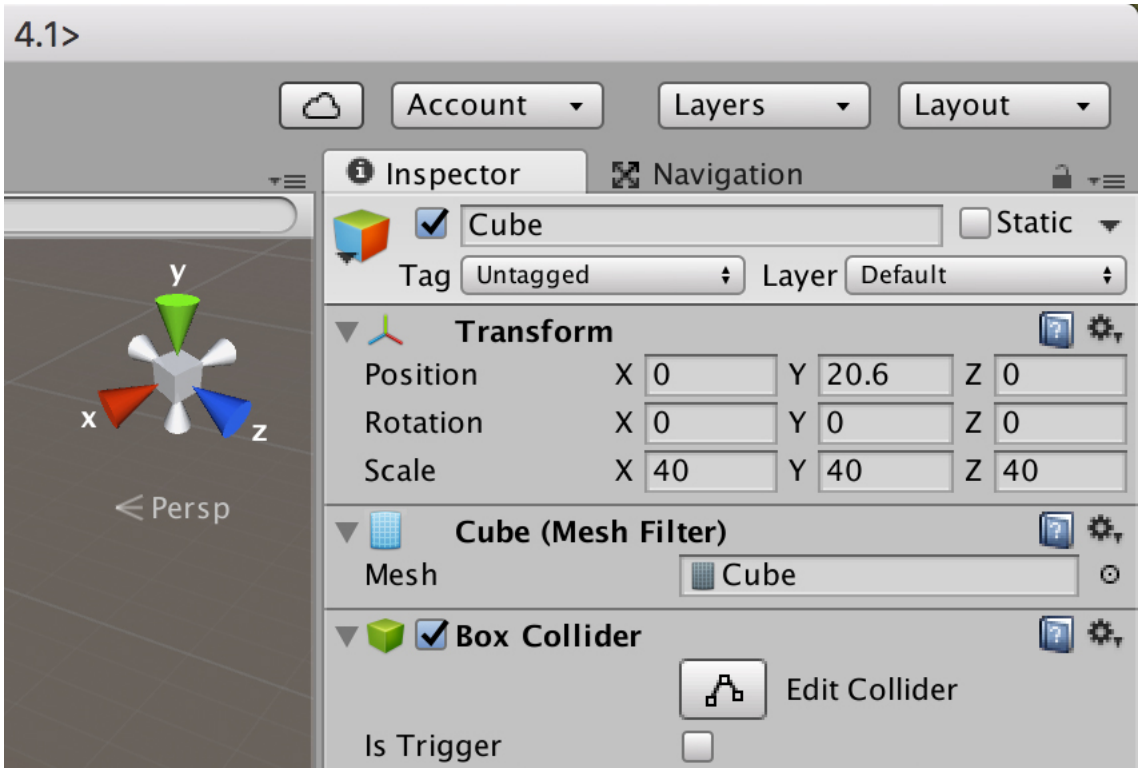


Figure 3.5: Steps to create game object:b [3]

	Vuforia	Wikitude
Type	Free+Commercial SDK Option	Free+Commercial SDK Option
iOS	Supported	Supported
Android	Supported	Supported
Marker	Advanced+VUMark	Advanced
3D Object Tracking	Only on box and cylinder and small size 3D objects too	Supported
GPS	Supported	Supported
Visual Search	Limited Support	Cloud Recognition and Offline as well
Face Tracking	Not Supported	Face Detection only
Web Support	Not Supported	Supported
Unity3D	Present	Present including 3D Tracking
Plugins API	No APIs	Integration with 3D parties available

Table 3.1: Comparison between Vuforia and Wikitude

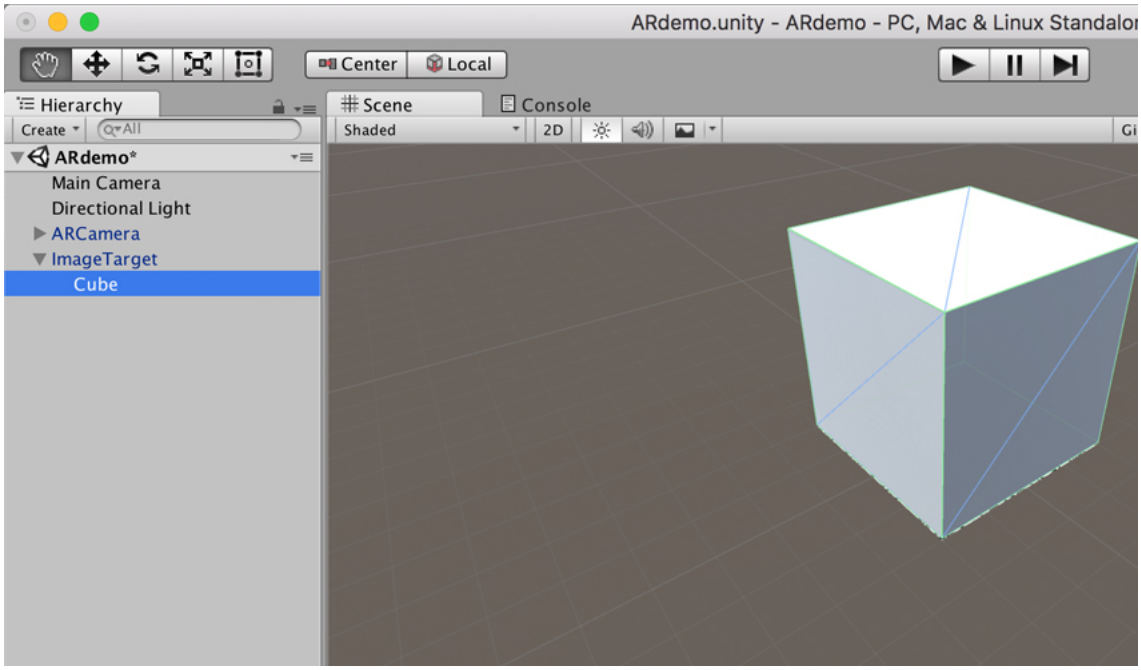


Figure 3.6: Steps to create game object:c [3]

3.2.5 Interaction Scripts

The steps show how a basic Augmented Reality application is developed that recognizes and tracks our target image and also displays the designated 3D graphics.But for a complete AR application, we also need to be able to interact

with the objects, augmenting the reality.For this, we need to detect where we clicked—or touched, in the case of a mobile device. We can do this by implementing a ray-tracer and can be done by creating a Script file in this folder.It should have same name as the file name. Naming is important because the code should match this specific file name.
Once we've created our rayTracer script,we can activate it by assigning it to one of the objects in the scene.We can select the ARCamera object and add the rayTracer scripts as a component using the Add Component button.

3.3 SLAM

New SLAM algorithms are still in developing stages,their requirements are driven by changing requirements and different kinds of maps or sensors or models. Most of the SLAM algorithms are derived from the following .

3.3.1 Mapping

Topological maps capture the connectivity (i.e., topology) of the scenario rather than creating a geometrically accurate map. Topological SLAM approaches have been used to enforce global consistency in metric SLAM algorithms.

In contrast, grid maps use arrays (typically square or hexagonal) of discretized cells to represent a topological world, and make inferences about which cells are occupied. Typically the cells are assumed to be statistically independent in order to simplify computation. Under such assumption,)$P(x_t - 1|m_t, o_t)$ are set to 1 if the new map's cells are consistent with the observation o_t at location x_t and 0 if inconsistent.Modern self driving cars mostly simplify the problem of mapping, by using the details of map which is collected beforehand. Details can include map annotations to the extent of marking locations of individual white line segments and curbs on the road. Location-tagged visual data such as Google's StreetView may also be used as part of maps. Essentially such systems simplify the SLAM problem to a simpler localisation only task, perhaps allowing for moving objects such as cars and people only to be updated

in the map at runtime.

3.3.2 Sensing

SLAM will always use several different types of sensors, and the powers and limits of various sensor types have been a major driver of new algorithms. Statistical independence is the mandatory requirement to cope with metric bias and with noise in measures. Different types of sensors give rise to different SLAM algorithms whose assumptions are which are most appropriate to the sensors. At one extreme, laser scans or visual features provide details of a great many points within an area, sometimes rendering SLAM inference unnecessary because shapes in these point clouds can be easily and unambiguously aligned at each step via image registration. At the opposite extreme, tactile sensors are extremely sparse as they contain only information about points very close to the agent, so they require strong prior models to compensate in purely tactile SLAM. Most practical SLAM tasks fall somewhere between these visual and tactile extremes.

Sensor models divide broadly into landmark-based and raw-data approaches. Landmarks are uniquely identifiable objects in the world whose location can be estimated by a sensor—such as wifi access points or radio beacons. Raw-data approaches make no assumption that landmarks can be identified, and instead model $P(o_t|x_t)$ directly as a function of the location.

3.3.3 Kinematics

The $P((x_t)$ defines the kinematic specifications of the model,represeting details about commands given to a robot. Kinematics is included because it gives us a correct estimation of sensing under conditions of inherent and ambient noise. The dynamic model adjusts the commitments from different sensors, different halfway mistake models lastly includes in a sharp virtual portrayal as a guide with the area and heading of the robot as some billow of likelihood. Mapping is the last delineating of such model, the guide is either such portrayal or the unique term for the model. For 2D robots, the commands are usually

given by a mixture of rotation and ”move forward” commands,by the addition of motor noise. The distribution formed by independent noise in angular and linear directions is non-Gaussian, but is often approximated by a Gaussian. An alternative approach is to ignore the kinematic term and read odometry data from robot wheels after each command—such data may then be treated as one of the sensors rather than as kinematics.

3.3.4 Multiple Objects

The related issues of data association and computational multifaceted nature are among the issues yet to be completely settled, for instance the distinguishing proof of various confusable tourist spots. A late development in the component based SLAM writing included the re-evaluation of the probabilistic establishment for Simultaneous Localisation and Mapping (SLAM) where it was presented as far as multi-question Bayesian sifting with arbitrary limited sets that give better execution than driving element based SLAM calculations in testing estimation situations with high false alert rates and high missed identification rates without the requirement for information affiliation

3.3.5 Moving objects

Non-static environments, like vehicles or pedestrians other than the subject material, still persists to present research challenges. SLAM with DATMO is a model which tracks moving objects in a similar way to the agent itself.

3.4 Landmark Extraction Algortihms

The two basic landmark extraction algorithms are:-

a. SPIKE

b. RANSAC

3.5 SPIKE

Spike deals with hybrid parallel solution for linear systems which is concerned with landmark extraction from laser or sonar scan range data.In scanning systems where scans yield multiple values within a certain angle of scanning, this algorithm tries to find extreme differences in the values read by the scanners. This happens when the distance measured at one angle is different to the distance measured at the next angle. Which in turn indicates that there is a geometric change between the angles, which can be interpreted as a landmark.
The process is bifurgated into two steps, algorithm being of three stages:
1. Diagnoal blocks factorization
2. Spikes computation
3. Providing a solution to the reduced matrix [7]

Figure 3.7: SPIKE [7]

3.6 RANSAC (Random Sampling Consensus)

3.6.1 Introduction to RANSAC

This method can be used to extract lines from a laser scan that can in turn be used as landmarks. RANSAC finds these line landmarks by randomly taking a sample of the laser readings and then using a least squares approximation to find the best fit line that runs through these readings. [7]
A basic assumption is that the data consists of ”inliers”, i.e., distributon of data can be determined by a model parameter, which may contain errors/noise,

and ”outliers” which are exceptions for the model. The outliers may be, from random values of the noise or from faulty measurements or wrong hypotheses about the elucidation of data. RANSAC follows that for the data that fit into the model, a procedure is developed which can estimate the model parameters that explains or fits this data.Random sample consensus,uses an iterative model for guessing a mathematical model from a set of data which consists of exceptions. It works by identifying those exceptions in a data set and determine the new model using the same information which is free of the outliers.
RANSAC is achieved by the following process:
1. Arbitarily select a subset of data
2. Fit a model to the selected subset
3. Define the number of exceptions
4. Keep repeating the above steps for a limited number of iterations.

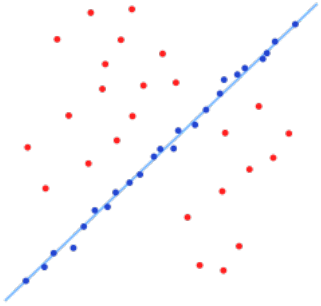


Figure 3.8: RANSAC [8]

3.6.2 Data Association

The problem of data association is that of matching observed landmarks from different (laser) scans with each other. [7] Problems in Data Association:

a. You might not re-observe landmarks every time.

b. You might observe something as being a landmark but fail to ever see it again.

c. You might wrongly associate a landmark to a previously seen landmark.

3.6.3 Nearest Neighbor Approach

a. When you get a new laser scan use landmark extraction to extract all visible landmarks.

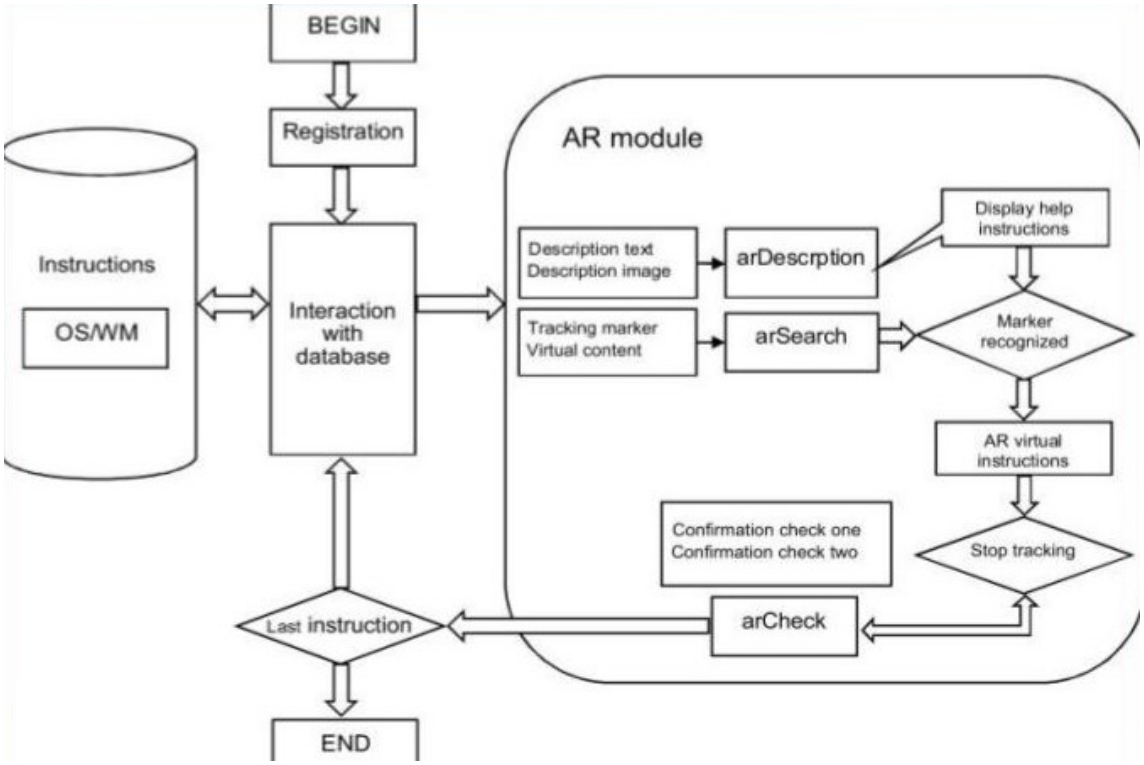
b. Associate each extracted landmark to the closest landmark we have seen more than N times in the database.

c. Pass each of these pairs of associations (extracted landmark, landmark in database) through a validation gate.

d. If the pair passes the validation gate it must be the same landmark we have re-observed so increment the number of times we have seen it in the database.

e. If the pair fails the validation gate add this landmark as a new landmark in the database and set the number of times we have seen it to 1.

3.6.4 Working Model of AR Shopping App



3.7 Rotation of an object

Use Transform.Rotate to rotate GameObjects in a variety of ways. The rotation is often provided as a Euler angle and not a Quaternion.

You can specify a rotation in world axes or local axes.

World axis rotation uses the coordinate system of the Scene, so when you start rotate a GameObject, its x, y, and z axes are aligned with the x, y, and z world axes. So if you randomly rotate a cube in world space, its axes align with the world. When you select a cube in the Unity Editor's Scene view, rotation Gizmos appear for the left/right, up/down and forward/back rotation axes. Moving these Gizmos rotates the cube around the axes. If you de-select and then re-select the cube, the axes restart in world alignment.

Local rotation uses the coordinate system of the GameObject itself. So, a newly created cube uses its x, y, and z axis set to zero rotation. Rotating the cube updates the rotation axes. If you de-select and the re-select the cube, the axes will be shown in the orientation they were before.

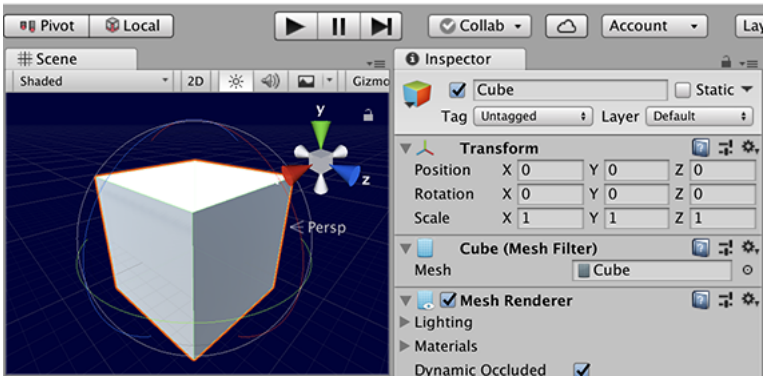


Figure 3.9: A cube not rotated in Local Gizmo Toggle [9]

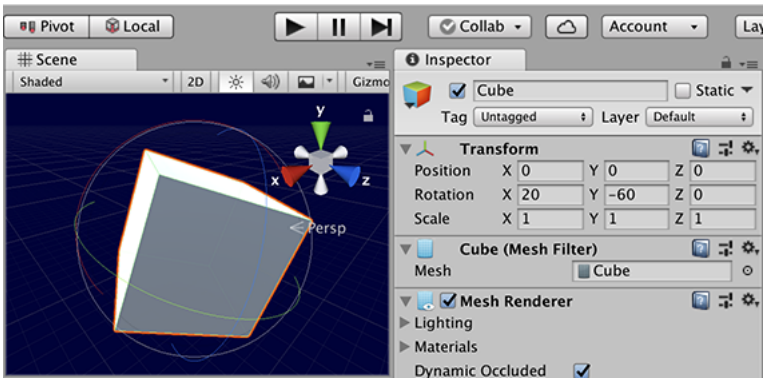


Figure 3.10: A cube rotated in Local Gizmo Toggle [9]

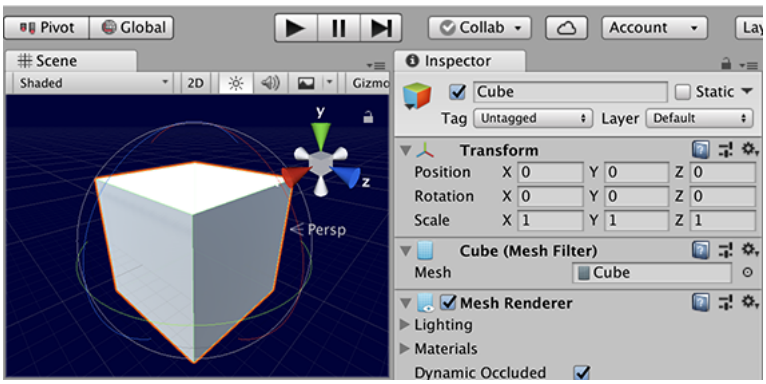


Figure 3.11: A cube not rotated in Global Gizmo Toggle [9]

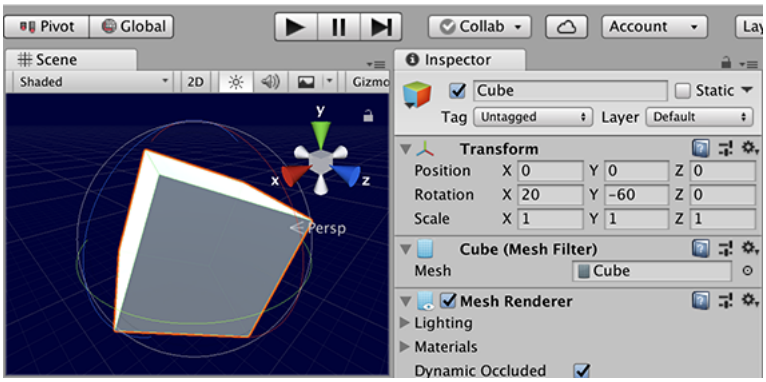


Figure 3.12: A cube rotated in Global Gizmo Toggle [9]

Chapter 4

Simulation and Result

4.1 Working prototype model of AR app

```
using UnityEngine;
using UnityEngine.UI;
using System.Collections.Generic;
using Wikitude;

public class InstantTrackerController : SampleController
{
      public GameObject ButtonDock;
      public GameObject InitializationControls;
      public Text HeightLabel;
      public Text ScaleLabel;

      public InstantTracker Tracker;

      public List<Button> Buttons;
      public List<GameObject> Models;

      public Image ActivityIndicator;

      public Color EnabledColor = new Color(0.2f, 0.75f, 0.2f,
0.8f);
      public Color DisabledColor = new Color(1.0f, 0.2f, 0.2f,
0.8f);

      private float _currentDeviceHeightAboveGround = 1.0f;

      private MoveController _moveController;
      public GridRenderer _gridRenderer;

      private HashSet<GameObject> _activeModels = new
HashSet<GameObject>();
      private InstantTrackingState _currentState =
InstantTrackingState.Initializing;
      private bool _isTracking = false;
```

Figure 4.1: Code Snippet : Instant Tracker Controller

```
public HashSet<GameObject> ActiveModels {
            get {
                  return _activeModels;
            }
      }

      private void Awake() {
            Application.targetFrameRate = 60;

            _moveController = GetComponent<MoveController>();
            _gridRenderer = GetComponent<GridRenderer>();
      }

      #region UI Events
      public void OnInitializeButtonClicked() {
            Tracker.SetState(InstantTrackingState.Tracking);
      }

      public void OnHeightValueChanged(float newHeightValue) {
            _currentDeviceHeightAboveGround = newHeightValue;
            HeightLabel.text = string.Format("{0:0.##} m",
_currentDeviceHeightAboveGround);
            Tracker.DeviceHeightAboveGround =
_currentDeviceHeightAboveGround;
      }

      public void OnBeginDrag (int modelIndex) {
            if (_isTracking) {
                  // Create object
                  GameObject modelPrefab = Models[modelIndex];
                  Transform model =
Instantiate(modelPrefab).transform;
                  _activeModels.Add(model.gameObject);
                  // Set model position at touch position
                  var cameraRay =
Camera.main.ScreenPointToRay(Input.mousePosition);
                  Plane p = new Plane(Vector3.up, Vector3.zero);
                  float enter;
                  if (p.Raycast(cameraRay, out enter)) {
                        model.position =
cameraRay.GetPoint(enter);
                  }

                  // Set model orientation to face toward the
camera
                  Quaternion modelRotation =
Quaternion.LookRotation(Vector3.ProjectOnPlane(-
Camera.main.transform.forward, Vector3.up), Vector3.up);
                  model.rotation = modelRotation;

                  _moveController.SetMoveObject(model);
            }
      }
```

Figure 4.2: Code Snippet : Instant Tracker Controller

```
public override void OnBackButtonClicked() {
            if (_currentState ==
InstantTrackingState.Initializing) {
                  base.OnBackButtonClicked();
            } else {

      Tracker.SetState(InstantTrackingState.Initializing);
            }
      }
      #endregion

      #region Tracker Events
      public void OnEnterFieldOfVision(string target) {
            SetSceneActive(true);
      }

      public void OnExitFieldOfVision(string target) {
            SetSceneActive(false);
      }

      private void SetSceneActive(bool active) {


            foreach (var button in Buttons) {
                  button.interactable = active;
            }

            foreach (var model in _activeModels) {
                  model.SetActive(active);
            }

            ActivityIndicator.color = active ? EnabledColor :
DisabledColor;

            _gridRenderer.enabled = active;

            _isTracking = active;
      }

      public void OnStateChanged(InstantTrackingState newState)
{
            Tracker.DeviceHeightAboveGround =
_currentDeviceHeightAboveGround;
            _currentState = newState;
            if (newState == InstantTrackingState.Tracking) {
                  InitializationControls.SetActive(false);
                  ButtonDock.SetActive(true);
            } else {
                  foreach (var model in _activeModels) {
                        Destroy(model);
                  }
                  _activeModels.Clear();

                  InitializationControls.SetActive(true);
                  ButtonDock.SetActive(false);
            }
            _gridRenderer.enabled = true;
      }
      #endregion
}
```

Figure 4.3: Code Snippet : Instant Tracker Controller

```
using UnityEngine;

public class ScaleController : MonoBehaviour
{
      public float MinScale = 0.1f;
      public float MaxScale = 0.5f;

      private InstantTrackerController _controller;
      private Transform _activeObject = null;

      private Vector3 _touch1StartGroundPosition;
      private Vector3 _touch2StartGroundPosition;
      private Vector3 _startObjectScale;

      private void Start () {
            _controller =
GetComponent<InstantTrackerController>();
      }

      private void Update () {
            if (Input.touchCount >= 2) {
                  Touch touch1 = Input.GetTouch(0);
                  Touch touch2 = Input.GetTouch(1);
                  Transform hitTransform;

                  if (_activeObject == null) {
                        if (GetTouchObject(touch1.position, out
hitTransform)) {
                              SetTouchObject(hitTransform);
                        } else if
(GetTouchObject(touch2.position, out hitTransform)) {
                              SetTouchObject(hitTransform);
                        } else if
(GetTouchObject((touch1.position + touch2.position) / 2, out
hitTransform)) {
                              SetTouchObject(hitTransform);
                        }

                        if (_activeObject != null) {
                              _touch1StartGroundPosition =
GetGroundPosition(touch1.position);
                              _touch2StartGroundPosition =
GetGroundPosition(touch2.position);
                              _startObjectScale =
_activeObject.localScale;
                        }
                  }
```

Figure 4.4: Code Snippet : Scale Controller

```
                  if (_activeObject != null) {
                        var touch1GroundPosition =
GetGroundPosition(touch1.position);
                        var touch2GroundPosition =
GetGroundPosition(touch2.position);

                        float startMagnitude =
(_touch1StartGroundPosition -
_touch2StartGroundPosition).magnitude;
                        float currentMagnitude =
(touch1GroundPosition - touch2GroundPosition).magnitude;

                        _activeObject.localScale =
_startObjectScale * (currentMagnitude / startMagnitude);

                        if (_activeObject.localScale.x
< MinScale) {
                              _activeObject.localScale = new
Vector3(MinScale, MinScale, MinScale);
                        }

                        if (_activeObject.localScale.x >
MaxScale) {
                              _activeObject.localScale = new
Vector3(MaxScale, MaxScale, MaxScale);
                        }
                  }
            } else {
                  _activeObject = null;
            }
      }

      private bool GetTouchObject(Vector2 touchPosition, out
Transform hitTransform) {
            var touchRay =
Camera.main.ScreenPointToRay(touchPosition);
            touchRay.origin -= touchRay.direction * 100.0f;

            RaycastHit hit;
            if (Physics.Raycast(touchRay, out hit)) {
                  hitTransform = hit.transform;
                  return true;
            }

            hitTransform = null;
            return false;
      }

      private Vector3 GetGroundPosition(Vector2 touchPosition)
{
            var groundPlane = new Plane(Vector3.up,
Vector3.zero);
            var touchRay =
Camera.main.ScreenPointToRay(touchPosition);
            float enter;
            if (groundPlane.Raycast(touchRay, out enter)) {
                  return touchRay.GetPoint(enter);
            }
            return Vector3.zero;
      }

      private void SetTouchObject(Transform newObject) {
            if
(_controller.ActiveModels.Contains(newObject.gameObject)) {
                  _activeObject = newObject;
            }
      }
}
```

Figure 4.5: Code Snippet : Scale Controller

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class table_1 : MonoBehaviour {
      private InstantTrackerController trackerScript;
      private GameObject ButtonsParent;


      // Use this for initialization
      void Start () {
            trackerScript = GameObject.Find
("Controller").gameObject.GetComponent<InstantTrackerControlle
r> ();
            ButtonsParent = GameObject.Find ("Buttons Parent");

            trackerScript._gridRenderer.enabled = false;
            ButtonsParent.SetActive (false);

      }

      void OnEnable(){
                trackerScript._gridRenderer.enabled = false;
                 ButtonsParent.SetActive (false);

      }

      void OnDisable(){
            ButtonsParent.SetActive (true);
      }
}
```

Figure 4.6: Code Snippet : Table Object

```
using UnityEngine;

public class DockUI : MonoBehaviour
{
     public RectTransform ButtonsBackground;
     public RectTransform ButtonsParent;

     private void Awake() {
          Update();
     }

     private void Update() {
          float currentRatio = (float)Screen.width /
(float)Screen.height;
          if (currentRatio > 1.0f) {
               ButtonsBackground.sizeDelta = new
Vector2(0f, -200f);
               ButtonsParent.sizeDelta = new
Vector2(-600f, 0f);
          } else {
               ButtonsBackground.sizeDelta = new
Vector2(0f, -206f);
               ButtonsParent.sizeDelta = new Vector2(0f,
0f);
          }
     }
}
```

Figure 4.7: Code Snippet : DockUI

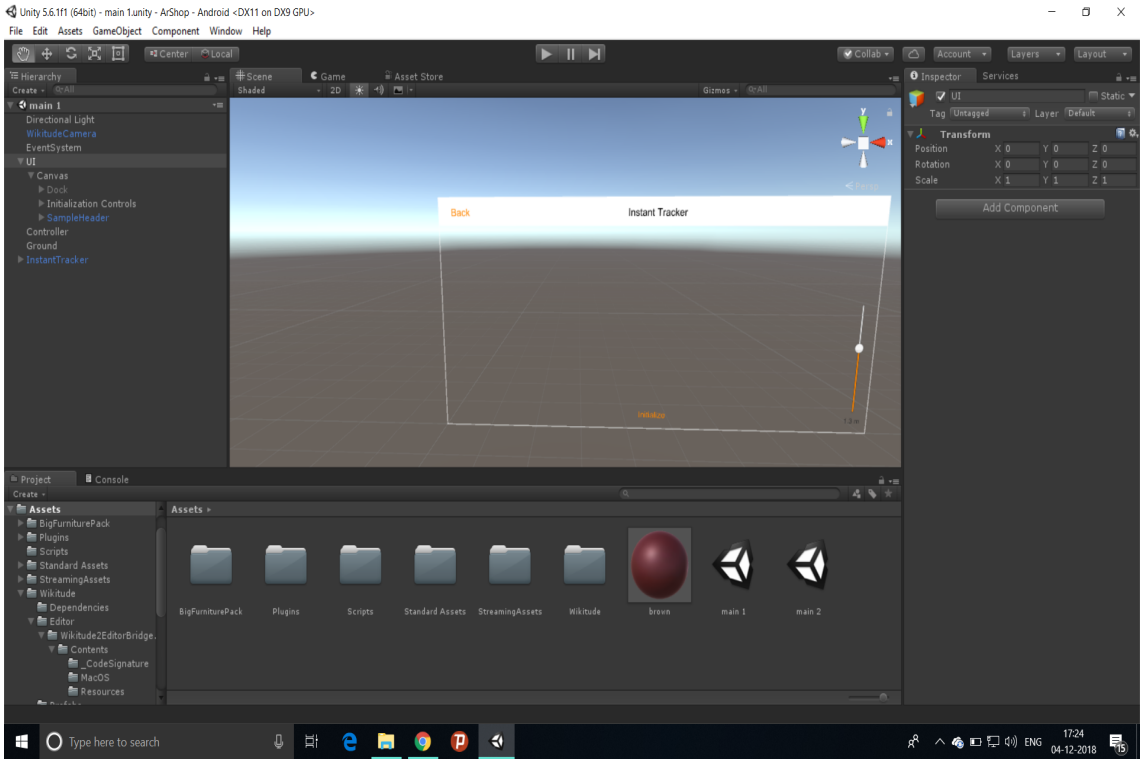


Figure 4.8: Scene window

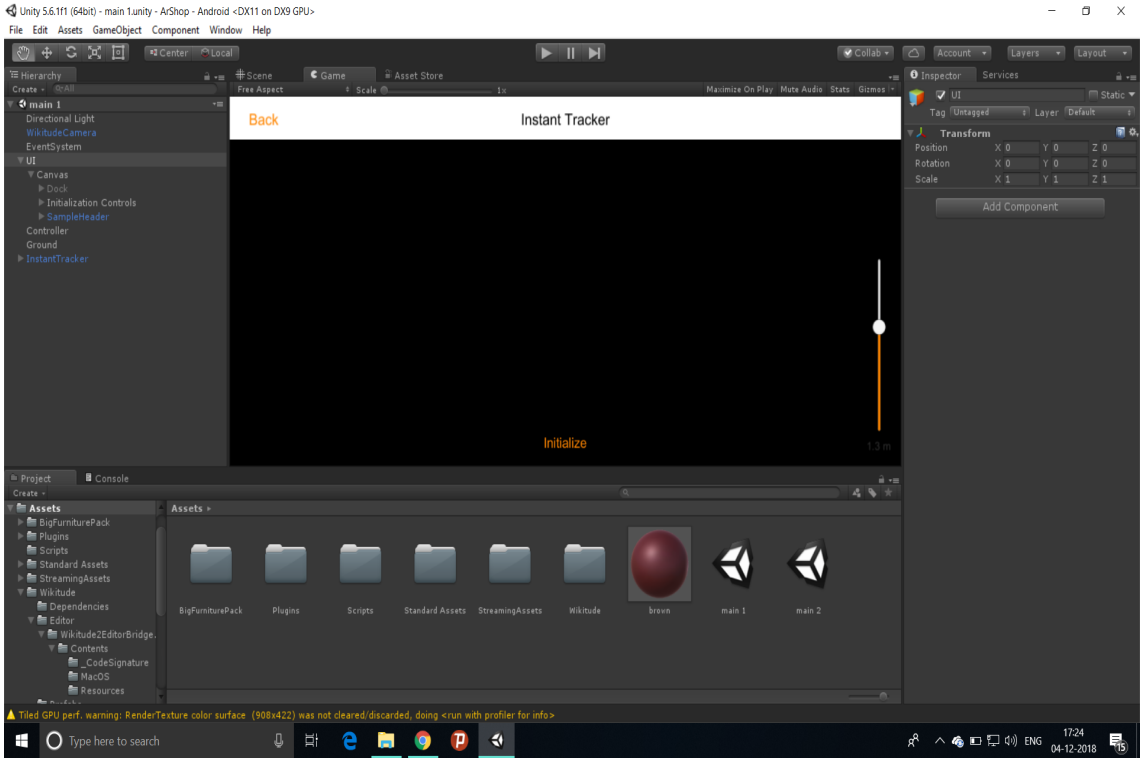


Figure 4.9: Game Window

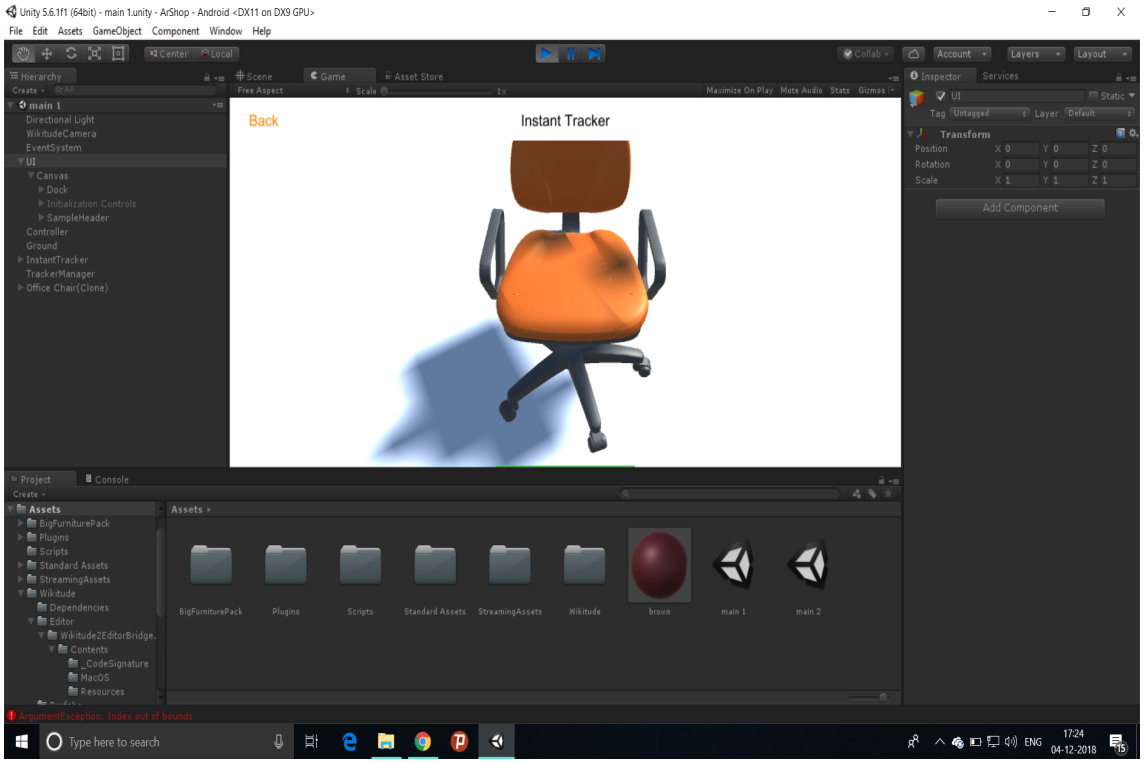
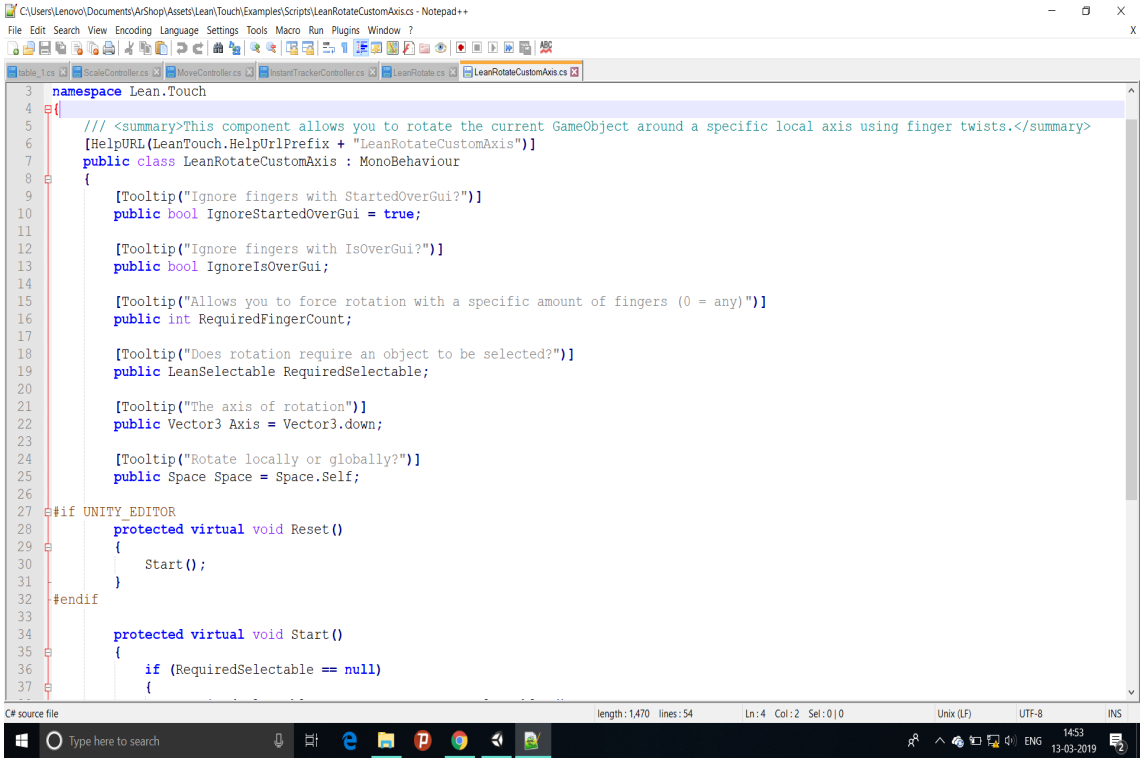
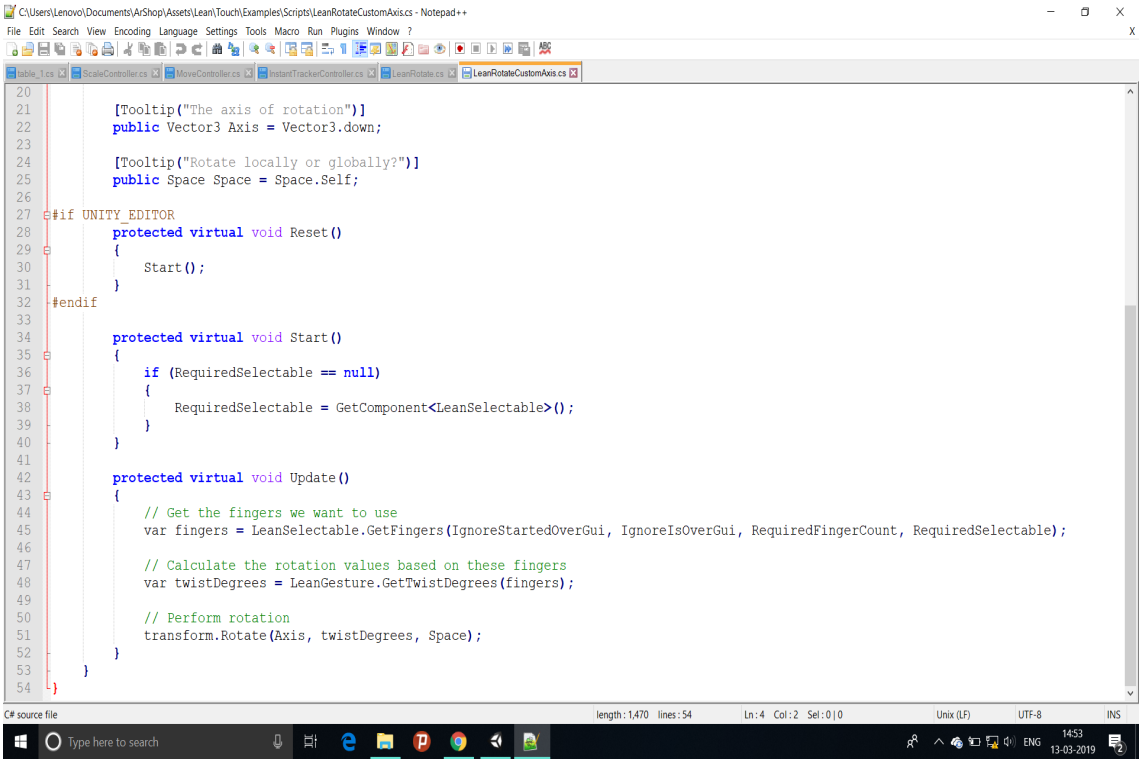


Figure 4.10: Output Result



```
namespace Lean.Touch
{
    /// <summary>This component allows you to rotate the current GameObject around a specific local axis using finger twists.</summary>
    [HelpURL(LeanTouch.HelpUrlPrefix + "LeanRotateCustomAxis")]
    public class LeanRotateCustomAxis : MonoBehaviour
    {
        [Tooltip("Ignore fingers with StartedOverGui?")]
        public bool IgnoreStartedOverGui = true;

        [Tooltip("Ignore fingers with IsOverGui?")]
        public bool IgnoreIsOverGui;

        [Tooltip("Allows you to force rotation with a specific amount of fingers (0 = any)")]
        public int RequiredFingerCount;

        [Tooltip("Does rotation require an object to be selected?")]
        public LeanSelectable RequiredSelectable;

        [Tooltip("The axis of rotation")]
        public Vector3 Axis = Vector3.down;

        [Tooltip("Rotate locally or globally?")]
        public Space Space = Space.Self;

#if UNITY_EDITOR
        protected virtual void Reset()
        {
            Start();
        }
#endif

        protected virtual void Start()
        {
            if (RequiredSelectable == null)
            {
```

Figure 4.11: Code Snippet : Rotating object (a)



```
        [Tooltip("The axis of rotation")]
        public Vector3 Axis = Vector3.down;

        [Tooltip("Rotate locally or globally?")]
        public Space Space = Space.Self;

#if UNITY_EDITOR
        protected virtual void Reset()
        {
            Start();
        }
#endif

        protected virtual void Start()
        {
            if (RequiredSelectable == null)
            {
                RequiredSelectable = GetComponent<LeanSelectable>();
            }
        }

        protected virtual void Update()
        {
            // Get the fingers we want to use
            var fingers = LeanSelectable.GetFingers(IgnoreStartedOverGui, IgnoreIsOverGui, RequiredFingerCount, RequiredSelectable);

            // Calculate the rotation values based on these fingers
            var twistDegrees = LeanGesture.GetTwistDegrees(fingers);

            // Perform rotation
            transform.Rotate(Axis, twistDegrees, Space);
        }
    }
}
```

Figure 4.12: Code Snippet : Rotating object (b)

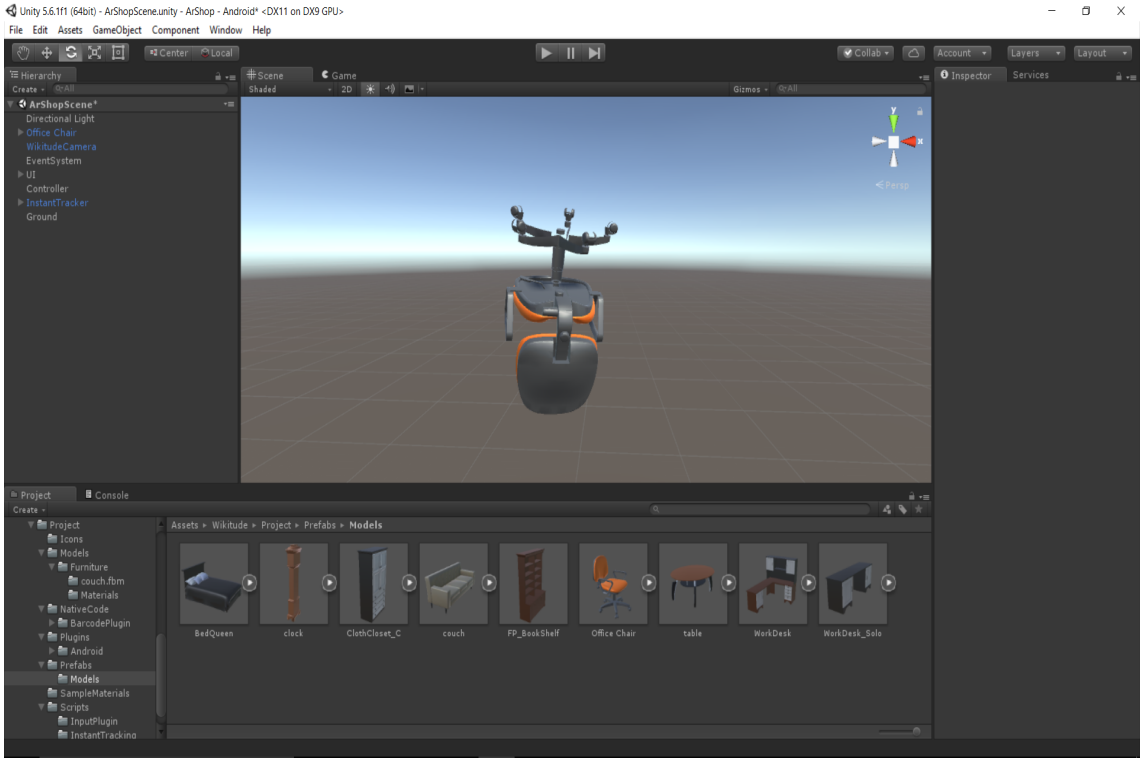


Figure 4.13: Vertically movable object(a)

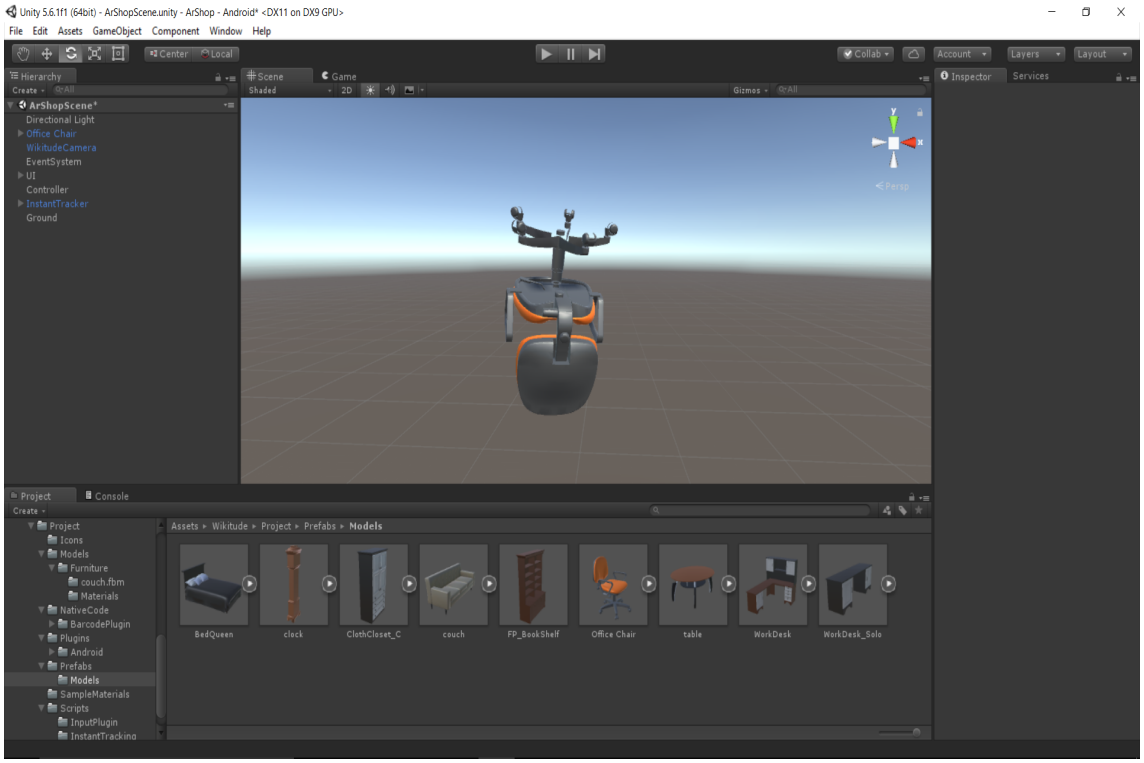


Figure 4.14: Vertically movable object (b)

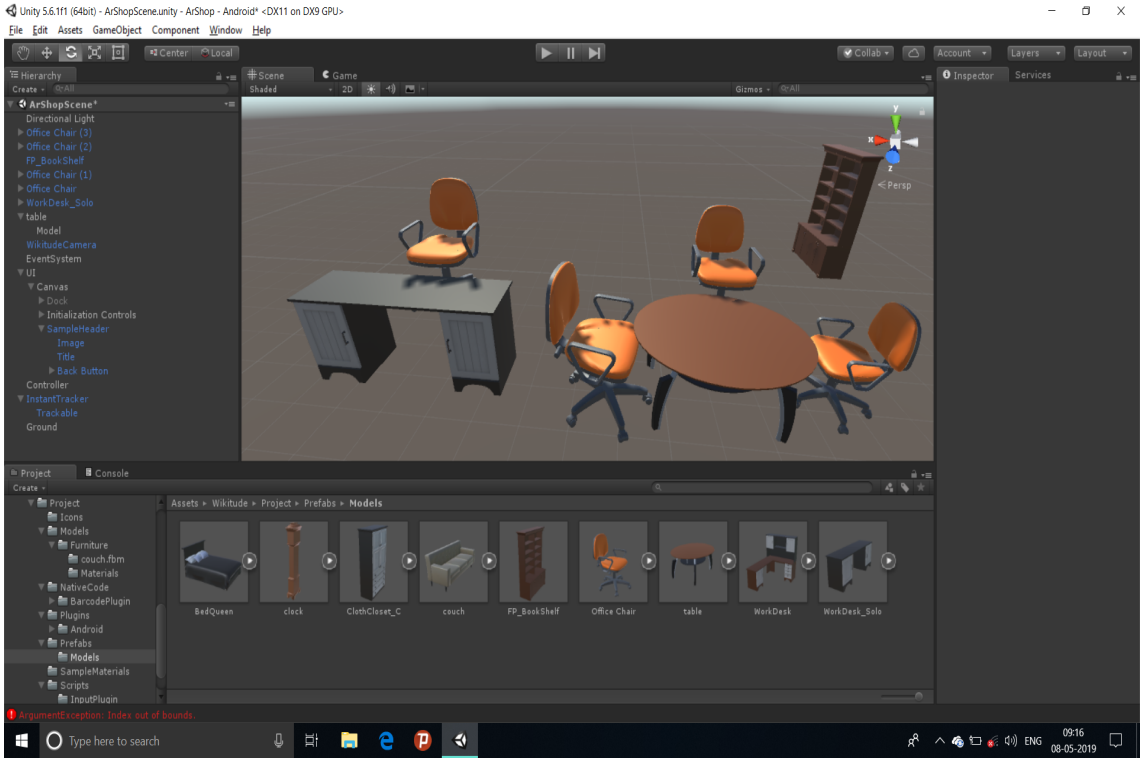


Figure 4.15: Multiple object scenario

Chapter 5

Conclusion and Future Works

5.1 Conclusion

Hence, we conclude our proposal for making an effective platform for online shopping by making a user interactive as well as dynamic shopping application.It is an Unity Engine based application. We made the application so that user can check the demonstration of the products that they are going to buy lively by adding the virtual objects of the products to the real immediate environment of the user. The main objective behind the project is to make end users as well as retailers more comfortable with online shopping by decreasing the return rate of the products to a large extent which would in turn add to growth of the business of the retailers as well as the customers would be more satisfied.

5.2 Future Works

We plan to build, test , integrate the modules of the whole system and again perform system testing so that we are ready with the augmented reality user interactive application which will benefit end users as well as retailers. For this, we would start by building small, high priority modules first then integrate them and test them. We then move on to next phase of building other modules which are of less priority and doing so repeatedly we would finally arrive at making an application that is in a production ready environment.

References

[1] Se.rit.edu.(n.d.).AugmentedRealityHomePages-Introduction. [online]Availableat:http://www.se.rit.edu/~jrv/research/ar/introduction.html[Accessed3Dec.2018].

[2] Etclab.mie.utoronto.ca.(n.d.).[online]Availableat: http://etclab.mie.utoronto.ca/publication/1994/Milgram_ Takemura_SPIE1994.pdf[Accessed3Dec.2018].

[3] Anon,(n.d.).[online]Availableat:https:// gamedevelopment.tutsplus.com/tutorials/ introduction-to-vuforia-on-unity-for-creating-augmented-reality\ -applications--cms-27693[Accessed3Dec.2018].

[4] Anon,(2006).FoundationsofLocalbasedServices.[online] Availableat:https://www.researchgate.net/figure/ d-Augmented-Reality-How-Stuff-Works-2005_fig8_ 238664871[Accessed3Dec.2018].

[5] Dspace.mit.edu.(n.d.).[online]Availableat:https://dspace. mit.edu/bitstream/handle/1721.1/36832/16-412JSpring2004/ NR/rdonlyres/Aeronautics-and-Astronautics/ 16-412JSpring2004/A3C5517F-C092-4554-AA43-232DC74609B3/0/ 1Aslam_blas_report.pdf[Accessed3Dec.2018].

[6] Robots.ox.ac.uk.(n.d.).ParallelTrackingandMappingforSmallARWorkspace .[online]Availableat:http://www.robots.ox.ac.uk/~gk/PTAM/ [Accessed3Dec.2018].

[7] Eng.auburn.edu.(n.d.).[online]Availableat:http://www.eng.auburn.edu/~agrawvd/COURSE/E7950_Spr16/Seminar-Venkata%20satya.pptx[Accessed3Dec.2018].

[8] En.wikipedia.org.(n.d.).Randomsampleconsensus.[online] Availableat:https://en.wikipedia.org/wiki/Random_sample_consensus[Accessed3Dec.2018].

[9] U.Technologies,"Unity-ScriptingAPI:Transform.Rotate", Docs.unity3d.com.[Online].Available:https://docs.unity3d.com/ScriptReference/Transform.Rotate.html.[Accessed: 13-Mar-2019]

group21_AugmentedReality

ORIGINALITY REPORT

7%	4%	1%	5%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	en.m.wikipedia.org Internet Source	2%
2	Submitted to University of Queensland Student Paper	1%
3	Ren, Fu, Qingyun Du, Xueling Wu, Jianya Gong, and Huayi Wu. "", International Conference on Earth Observation Data Processing and Analysis (ICEODPA), 2008. Publication	1%
4	Submitted to Universiti Teknologi Malaysia Student Paper	1%
5	socialcompare.com Internet Source	<1%
6	Submitted to Bridgepoint Education Student Paper	<1%
7	Submitted to University of Oxford Student Paper	<1%
8	Submitted to UT, Dallas Student Paper	<1%

9	Submitted to Tshwane University of Technology Student Paper	<1%
10	Submitted to Rochester Institute of Technology Student Paper	<1%
11	wma.my Internet Source	<1%
12	Submitted to University of Northumbria at Newcastle Student Paper	<1%
13	Submitted to Universiti Malaysia Pahang Student Paper	<1%
14	fedetd.mis.nsysu.edu.tw Internet Source	<1%

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	On		