American University of Ras Al Khaimah
School of Engineering
SENIOR DESIGN PROECT-(II)

Educational lab in VR

Omar Irfan Khan (2013000987)

Mohammad Eyad Masoud (2013001165)

Supervised by: Dr. Abdul Halim Jallad

Department of Computer Science and Computer Engineering
Submitted in partial fulfillment of the requirements of B.Sc. Degree in Computer Engineering

June 9, 2020

June 9, 2020

Students Statement

We, the undersigned students, certify and confirm that the work submitted in this project report is entirely our own and has not been copied from any other source. Any material that has been used from other sources has been properly cited and acknowledged in the report.

We are fully aware that any copying or improper citation of references/sources used in this report will be considered plagiarism, which is a clear violation of the Code of Ethics of the American University of Ras Al Khaimah.

Omar Irfan Khan (2013000987)

Mohammad Eyad Masoud (2013001165)

June 9, 2020

Supervisor Certification

This to certify that the work presented in this senior year project manuscript was carried out under my supervision, which is entitled:

“**Educational lab in VR**”

Omar Irfan Khan (2013000987)

Mohammad Eyad Masoud (2013001165)

I hereby that the aforementioned students have successfully finished their senior year project and by submitting this report they have fulfilled in partial the requirements of B.Sc. Degree in ____ Engineering.

I also, hereby that I have **read, reviewed and corrected the technical content** of this report and I believe that it is adequate in scope, quality and content and it is in alignment with the ABET requirements and the department guidelines.

Dr. Abdul Halim Jallad,

June 9, 2020

ACKNOWLEDGMENT

Firstly we would like to express our deepest gratitude to our senior design project advisor Dr. Abdul Halim Jallad who guided us and gave his invaluable advice throughout the project. Without his guidance we may have not been able to go down the right path of development and are indebted to him.

Also, this project would not have been initiated/accomplished without the commitment, resiliency and endeavor of each member. Moreover, we would like to acknowledge the diligent senior design project committee for providing us with the project templates and a step by step explanation of how important each phase is.

Furthermore, we would like to thank our parents, friends, and people who helped us out. They supported and encouraged us throughout this project which kept us persevering and patient during the development of this project.

June 9, 2020

ABSTRACT

Health and safety play an important role in our daily lives. It is often overlooked by students especially in educational institutions. Chemistry laboratories are the first place where safety is not one of the top priorities while making the lab. Although, the professors do inform the students on what precautions to take while handling chemicals or while using a burner. We aim to show each hazard that can happen if proper precautions are not taken. Moreover, we tend to show all this in an environment where the user would not be harmed in real life and yet carry out dangerous experiments in virtual reality. The definition of virtual reality (VR) is derived from both the words 'virtual' and 'reality'. The definition of virtual is near and reality is what we experience as human beings. In technical terms, VR is the terminology used to describe a three-dimensional, computer-generated environment which can be explored and interacted by a person. The person becomes a part of this virtual world or is immersed within this environment and is allowed to perform several actions.

Keywords:

Virtual Reality (VR), Education System, Health and safety, Hazards, Precautions, Experiments.

TABLE OF CONTENTS

LIST OF FIGURES

Figure *Page*

LIST OF TABLES

LIST OF ABBREVIATIONS

LIST OF ABBREVIATIONS	
CPU	Central processing unit
FPS	Frames per second
GPU	Graphical processing unit
HMD	Head mounted device
PC	Personal computer
PTSD	Post-traumatic stress disorder
VR	Virtual reality

1 Chapter 1: Introduction

1.1 Problem Statement and Purpose

Health and safety regulation play a major role in our society especially for students. Since they are more vulnerable to accidents as they are not aware of what consequences may occur. Since virtual reality (VR) is a new area and implementing pipelines and control systems to demonstrate health and safety is pretty advanced. We decided to take a step back to build the basics of health and safety training. This project aims to demonstrate how students can be safe in a chemistry lab by implementing the things that could possibly go wrong in a lab and what a student should if such a scenario arises by incorporating virtual reality into health and safety training for students.

1.2 Project and Design Objectives

The objectives of this project is as follows:

To design and construct a laboratory in VR which provides the user with accessibility to use various apparatus such apparatus such as Bunsen burner, conical flasks. This simulation can be utilized by academic institutions to make their students more aware about the health and safety issues.

Design objectives:

- Develop the relevant apparatus for the laboratory so it can mimic and show the desired results as if the user was using this equipment in real life.
- Construct various sprites and lightning models to create the perfect atmosphere.

- Maintain the memory and graphics usage to optimal levels.
- Develop rigidbodies and appropriate colliders so that the object functions like how it should in real life.
- Retain a solid 60-75 frame per seconds (FPS) as well as consistently outputs at the desired resolution of the head mounted display (HMD) refresh rate.
- Create scripts that will allow the user to interact with objects around the lab.
- Allow the user to walk around the room and access various areas of the lab given, that it is within the play space area defined by the chaperone system.

1.3 Intended Outcomes and Deliverables

The intended outcomes of the project:

This project aims to deliver a working chemistry laboratory which will allow users to explore various situations where they can freely test and learn more about safety guidelines. It will enable students to comprehend the relevance of maintaining a safe workplace. Moreover, they will demonstrate an understanding of how to prevent injuries in the laboratory, and to adequately assess and manage it accordingly. Furthermore, it will allow students to easily decipher information regarding types of exposure, policies, procedures and equipment needed to deal with hazardous materials.

The deliverables are as follows:

- Chemistry lab that demonstrates education and training concepts in VR.
- Assets that could be used in the future for further development of educational labs.

1.4 Summary of Report Structure

The report is divided up into six sections:

Chapter one of the report covers the existing problem that has negative effects on academia. Then, stating the purpose of constructing such a system for the problem. Afterwards, discussing the intended deliverables that we have produced in our project followed by the project and design objectives that need to be satisfied.

Chapter two provides an overview on how relevant is our project as compared to the research that is being done, to the preferred market and how this project can relate to ethics and environmental issues.

Chapter three discusses the introductory design of the project in details which consists of the project stipulation, the design process, legal aspects, design constraints, design standards, design alternatives, simulation tests and safety considerations.

Chapter four comprises of the results produced from our project and how closely do they resemble a real life chemistry laboratory. Also, there are discussions about which software was used and how several assets were created.

Chapter five contains the management procedure of how the project has been partitioned into various phases the project has gone through, the estimated required budget, quality and risk management.

Lastly, chapter six summarizes the deliverables produced along with future plans on how we can build upon this scenario. Additionally, the references and appendices are also mentioned here.

2 Chapter 2: Background

There are three categories of reality:

1. Virtual Reality (VR)
2. Augmented Reality (AR)
3. Mixed Reality (MR)

VR had existed since the 1960s but during the 1980s the excitement of VR had died down. Mainly due to the limited amount of computing power and display technology at that time. Up until 2012 an entrepreneur by the name of Palmer Luckey made a VR headset called the Oculus Rift that had solved some of the problems that proved unsolvable back then. In 2016 HTC Vive was released. Since it's a relatively new technology, VR is being amalgamated in gaming, films, construction, sports and museums. VR creates a virtual environment for the user.

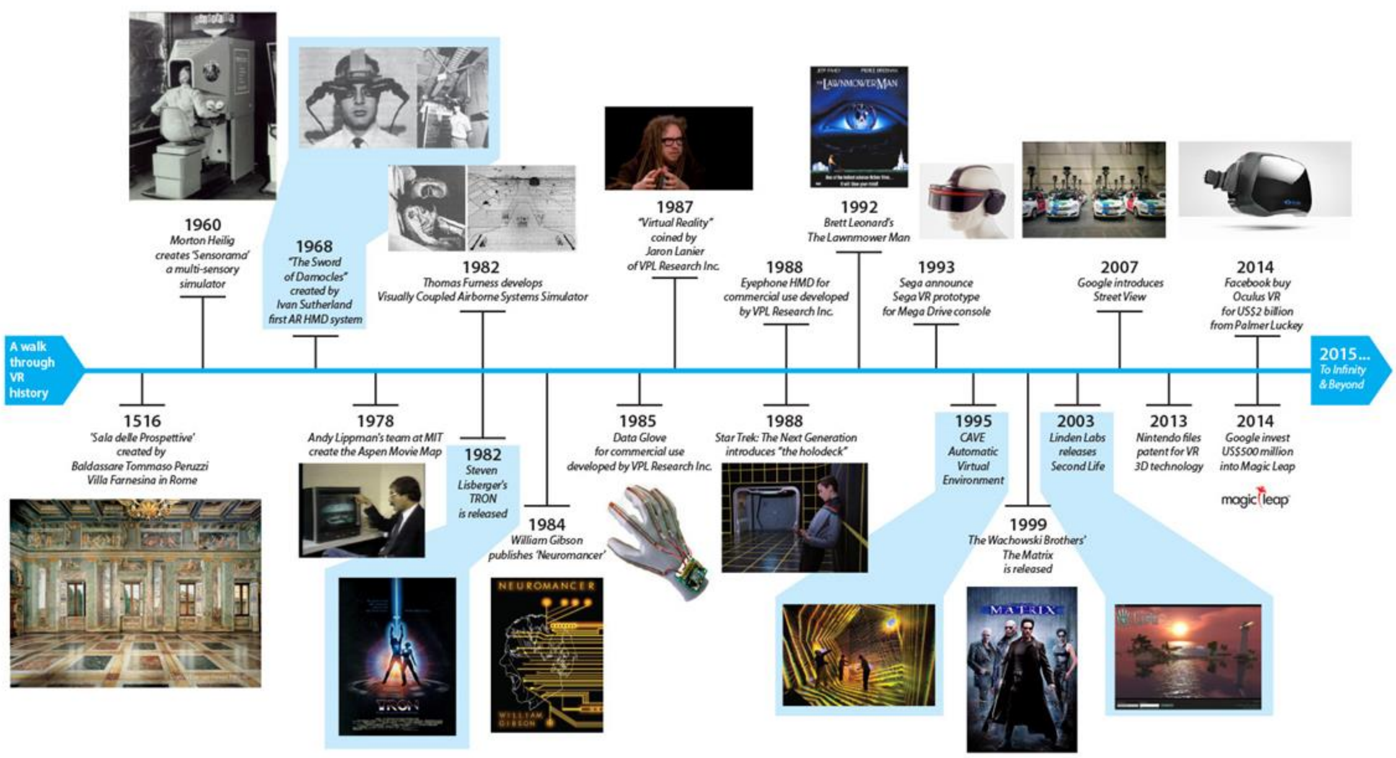


Fig. 1 VR history

AR on the other hand is not as old as VR but it existed before in the form of Google glass, Daqri smart helmet or Epsons moverio. AR takes the view of the real world and adds data over it in the form of either numbers displaying your walking or notifications from social media. Unlike VR, AR can be easily implemented in our daily lives for example in our mobile phones. The potential market for both AR and VR is much diverse but, AR has the upper hand rather than VR since there are more applications that can be made with AR which provide convenience and ease to the user in the same reality. The most commonly known application of AR is Pokemon Go.

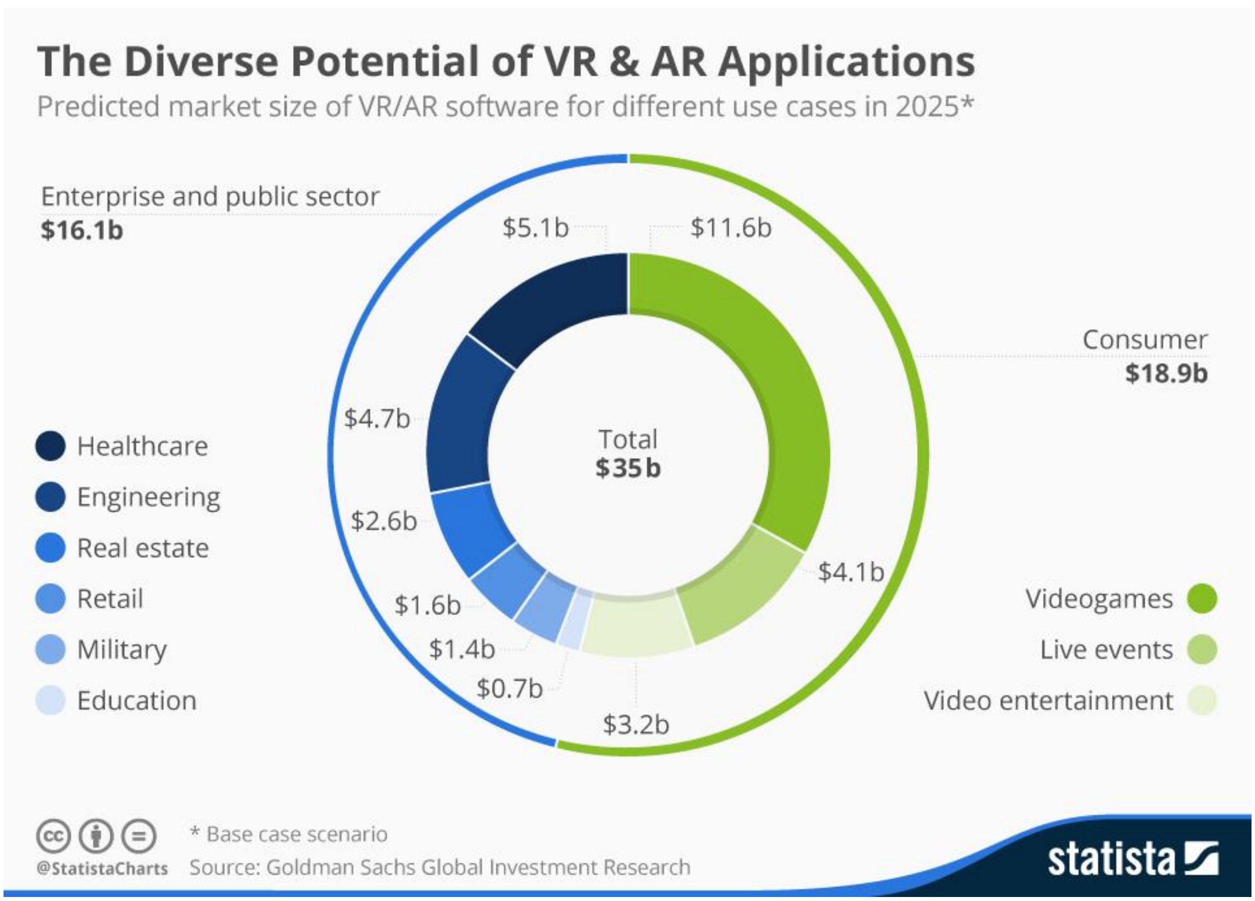


Fig. 2 AR & VR potential

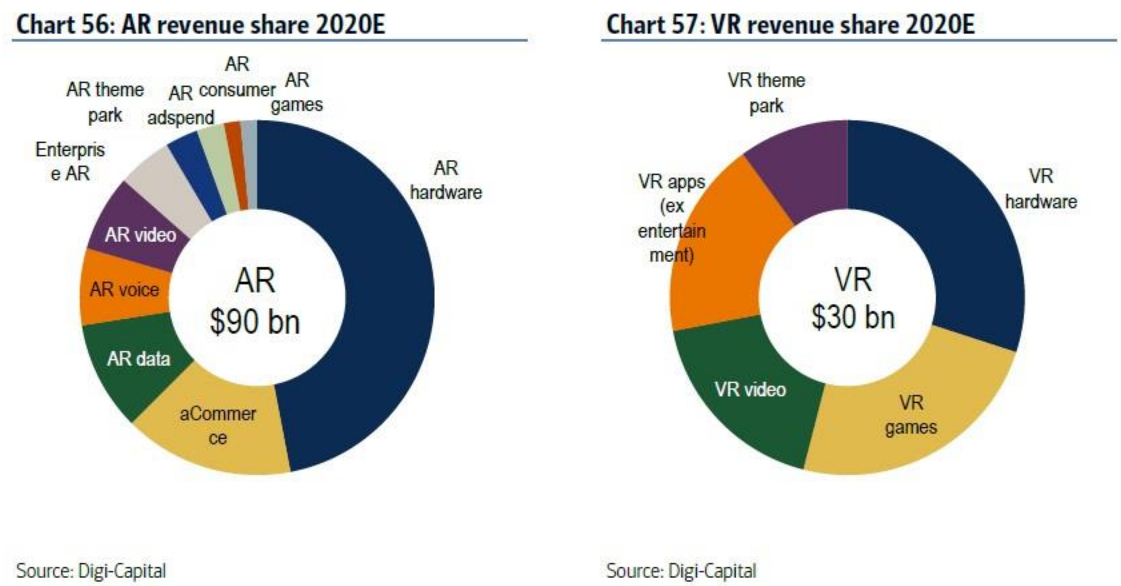


Fig. 3 AR versus VR revenue

Fig. 4 AR in the commercial industry

MR is the least commonly known between the two since there has not been much news about it. MR uses the best of both worlds between the two realities. It tries to combine certain characteristics of VR and AR together. MR allows the user to see the real world but it also adds other virtual objects and fixes them to a point in space. Consequently, whenever the user moves around the distance to the virtual object decreases or increases depending on the movement of the user [1]. Recently, Microsoft announced HoloLens which is based on MR.

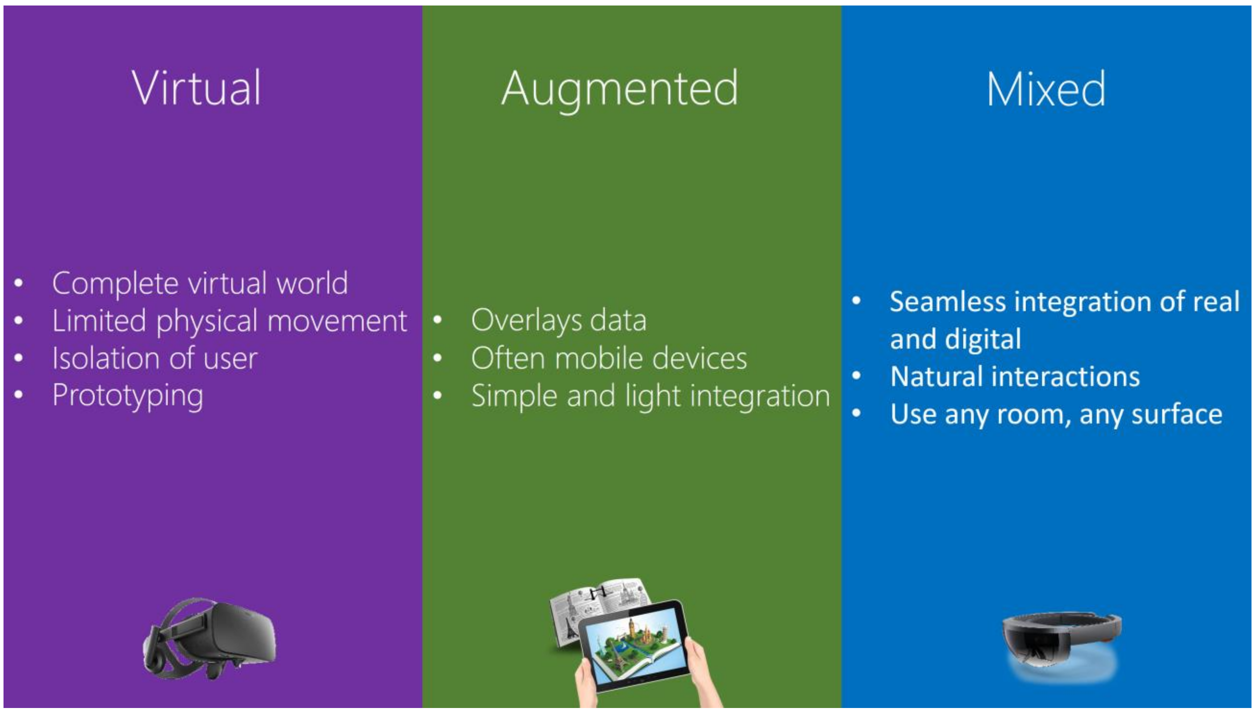


Fig. 5 Summary of VR, AR and MR

2.1 Relevant Literature Research

Since this topic is fairly new the amount of research papers are very limited. VR has been used for education by various organizations such as Gamar, Google Expeditions Pioneer, Program, Unimersiv, Immersive VR education and many more. Immersive VR education claim that students of all ages usually retain 10% of what they read and 30% of what they see. Using VR in education enables students to store more information as more senses are involved in the learning experience [2].

Additionally, one of the applications the organization has created is Lecture VR which allows us to recreate any lecture or record any live lecture to be replayed infinitely [3]. NearPod is another one of these tools that allow teachers to engage students with interactive lessons [4]. Furthermore, Google Expeditions Pioneer Program provides the necessary tools for a teacher to take their students on a field trip to virtually anywhere. It could be to the coral reefs or the surface of Mars in the afternoon [5].

3 Chapter 3: Methods and Materials

This chapter elaborates more on the design and approaches that were used for the project as well as how the prototype was assembled including specifications, alternatives that could have been used, how we could optimize the application, standards and constraints.

Methodology:

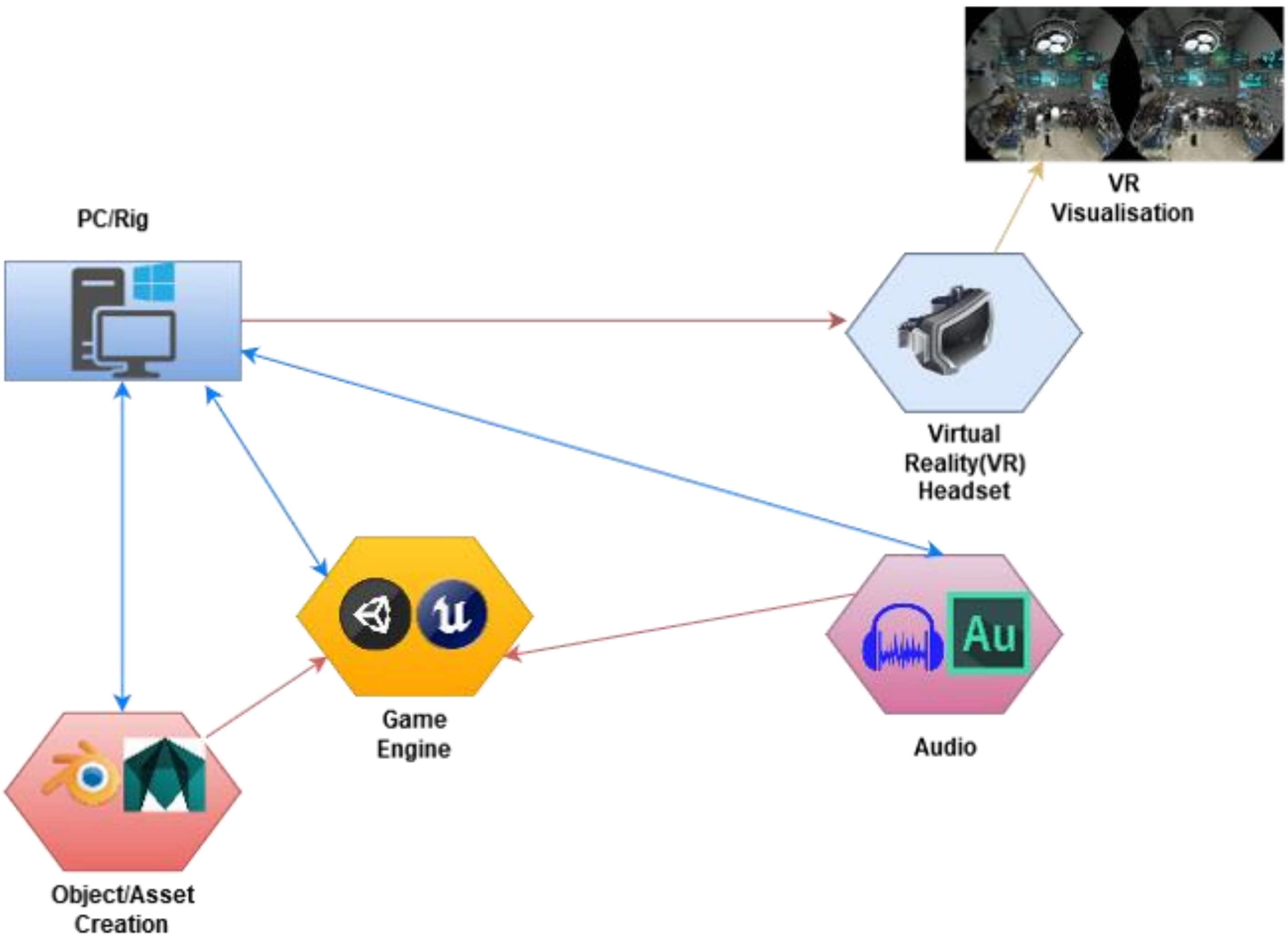


Fig. 6 Block Diagram

3.1 System Design and Components

We used the following components to make the laboratory plus the work possible in the laboratory:

- HTC Vive
- A compatible PC/laptop that is VR ready.
- Unity to make the laboratory more interactive
- Blender to make the laboratory
- Audio files from the internet to add sound

3.2 Design Specifications, Standards and Constraints

HTC vive uses two screens, each for one eye. The device uses more than 70 sensors and one of them is a MEMS (microelectromechanical system). It allows precision tracking by the lighthouse which uses simple photosensors on any object that needs to be tracked. On the other hand, the pc has nvidia gtx 1070, 32 gb ram and 1 tb hard drive space. All these specifications make the pc more than ready to handle VR headsets.

We modelled the laboratory in blender to allow easy rigging and placement of objects. Moreover, unity does not allow the freedom to model objects as freely as blender allows.

The standards that we are going to follow are American Section of the International Association for Testing Materials (ASTM International). This standard clearly states on how to perform each test and what the guidelines for each procedure are.

The main constraints are that the HTC Vive has to be tethered to the PC and its cable length is only about 5-6 meters. The newer model of vive is a wireless version and gives more freedom and convenience for the user to move around. Moreover, VR headsets are too expensive right now and not many people are willing to spend $2000 on such devices. In addition, people do not always have a big open space available to play in. Most VR headsets offer a seated experience but it is not worth it as you cannot interact with object or move around.

On the other hand, blender has unintuitive layers system and a sluggish view port performance with high poly scenes. Rigging is also a bit clunky but doable.

Furthermore, Unity has most of the features locked since you have to pay for it. Also, performance problems may be hard to locate, address and fix since you are usually dealing with black box (no code or very less code is used) and there is no multi-threading. Also, mono which is a development framework but it is so ancient that it hardly gets updated. This causes problems in garbage collection (GC) once your project reaches a certain size as GC consumes a significant amount of your central processing unit (CPU) time.

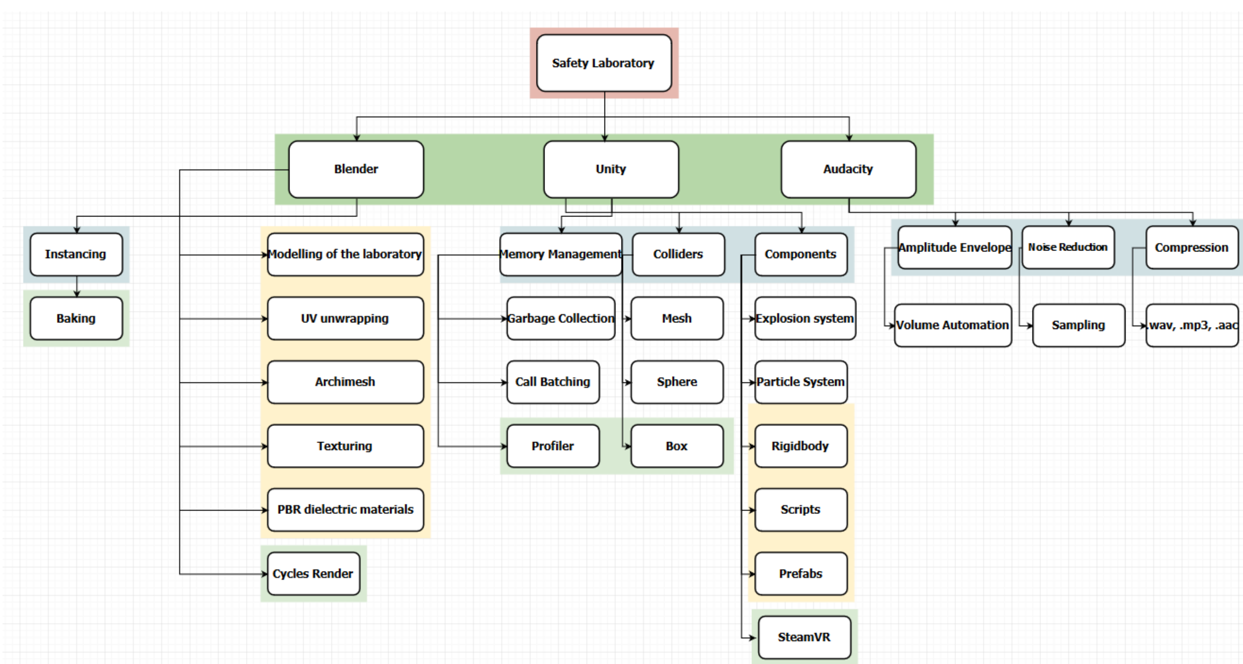


Fig. 7 Components flowchart

	HTC Vive	Oculus Rift	PlayStation VR	Samsung Gear	Google Cardboard
MSRP	$799.00	$599.00	$399.99	$99.99	$15.00
Platform	Windows PC	Windows PC	PlayStation 4	Samsung Smartphone	Smartphone
Experience	Seated VR, Standing VR, Room-scale VR (up to 5 m diagonally), Positional tracking	Positional tracking, Standing VR, Seated VR	Positional tracking, Standing VR, Seated VR	Seated VR	Seated VR
Field of View	110 degrees	110 degrees	100 degrees	96 degrees	Varies
Refresh Rate	90 Hz	90 Hz	90 - 120 Hz	60 Hz	60 Hz
Display Resolution per Eye	1080 x 1200	1080 x 1200	1080 x 960	1440 x 1280	Varies
Headset Weight	1.2 lbs	1.0 lbs	1.3 lbs	0.7 lbs (without phone)	0.2 lbs (without phone)
Fit	Adjustable headset strap, fits most glasses, 2 eye relief adjustments: lens distance from eye & interpupilliary distance	Fits glasses, linterpupilliary distance adjustment	Fits glasses, lens distance from eye can be adjusted, linterpupilliary distance adjustment	Fits glasses, focus adjustment	Varies
Included in the Box:	Headset, 2 x Controllers, 2 x Base Stations, Earbuds, Link Box, Link Box power adapter, Link Box mounting pad, 2 x Base Station power adapter, 2 x Micro USB chargers, Sync cable, HDMI cable, USB cable, Cleaning Cloth, Alternate face cushion	Oculus Rift Headset, Camera Sensor, Remote, Xbox One Wireless Controller, Xbox USB Wireless Adapter, 2xAA Batteries, Integrated Headphone Removal Tool, 2 x Oculus Logo Stickers, Oculus Lens Wipe Cloth	Headset, Processor Unit, connection cable, HDMI cable, USB cable, Stereo headphones, AC power cord, AC adaptor, demo disc	Headset	Headset

Fig. 8 VR headset comparison

Unity	Unreal
Supports 21+ platforms (PC, Console, Mobile, Web)	A visual scripting system for non-coders enables quick prototyping
Elaborate documentation is provided	Supports approximately 6 platforms (Mainly PC and Consoles)
Loads of official & community tutorials out there	Has a history of being one of the nicest looking engines
Free if you make <$100K from it	Mediocre documentation
Easy for programmers, Easy for designers	Lots of tutorials if you are a designer, very little for coders
Great for building any kind of game	Free if you make $<$50K from it
Flexibility is provided by a strong component programming model	Built from a FPS, which makes it a lot harder to do non FPS games
Works with 3rd party IDEs	Developers have full control of the engine and source code
Offers choice of scripting languages	Quick release-cycle
Has a great animation system	Slow and very high build size

Table. 1 Unity versus Unreal Engine

3.3 Design Alternatives

- VR Headset

It has been a difficult choice to choose between Oculus Rift or the HTC Vive. Oculus tries to make the setup process as simple as possible but it is rather drawn out since Oculus requires two USB 3.0 ports; one for tracking the camera and another for the headset [5].Unfortunately, neither of the headsets listed above provide any way of interaction with the virtual world which limits your virtual reality experience to only two senses that is sight and sound. They all have to use a controller. Fortunately, the Vive provides motion controllers to interact with the virtual world while you traverse through it making you use three of your five senses: sight, sound and haptics/touch. Recently, Facebook announced Oculus Touch which gives controllers and allows interaction with the environment just like Vive's motion controller does. But it will be released in december 2016. The oculus touch is an extra $200 plus you need another sensor to track the movement accurately. This makes the Rift price around $880.

- Object/Asset Creation

There are various software's to use but the ones that most stand out in the inexpensive category are blender and sculptris. Since blender has more similarities to maya and more robust than sculptris we decided to go with blender.

- Game Engine

The alternatives are Unreal, CryEngine and Torque3D. We decided to go with Unity since it has a more user friendly interface and more simple programming languages to use such as boo which is very similar to python and C#. Although, Unreal supports C++ but unfortunately making a script is a bit more complex in Unreal since you need to create a blueprint first and assign a set of parameters to the variables.

- Sound

There are many software's available but mainly the ones that are free are Audacity, OcenAudio, and Wavosaur. Audiowwise and Adobe Audition are both for professionals and very expensive. Whilst, both software's provide a lot of tools to use unlike Audacity or OcenAudio. But in the end, what matters is the ease of use and Audacity is really easy to use.

3.4 System Analysis and Optimization

After further observation we saw that there are a lot of optimizations to be done to ensure the game runs smoothly as possible without monopolizing all the memory. The areas that needed to be specifically optimized were, graphical processing unit (GPU), shaders and textures, CPU, and audio.

❖ GPU

The graphical aspects of your game can affect both the CPU and the GPU. Both components play an important part in optimization since the GPU does more work for the CPU while optimizing. The GPU is often plagued by bottlenecks such as the fillrate (number of pixels a GPU can render to screen) or memory bandwidth consumption and the calculation of the number of vertices for each object along with the complexity of the vertex shader. We solved the fillrate problem by disabling some components of the GPU and lowering the resolution slightly. As for the vertices, we limited calculations to only a several million vertices to improve performance.

❖ Shaders and Textures

Texture packing was used to reduce the build size and increase performance of the game. Texture packing combines all textures into one large image instead of storing them all as separate textures.

❖ CPU

The CPU has a lot of processing to do to render objects on the screen such as, light reflections of an object and which object is the light hitting, sending drawing commands to the GPU and more. So to reduce the amount of work done by the CPU we combined objects together using Unity's batching call which can either be static or dynamic. The static batching provides a significant improve in performance as compared to dynamic batching. Moreover, the objects that caused more reflections/shadows need to be rendered multiple times were used fewer times in the laboratory.

- Audio

Audio can also play a significant hidden role in bringing down the performance of the laboratory. Most of the audio we used were native files for both short sound effects and for larger files. After further digging through the documentation we found that compressed audio is best for larger files that are usually used for background music.

3.5 Simulation and/or Experimental Test

Firstly, to start making such a laboratory possible we had to either model it or buy an existing one from turbosquid or any other model sites. We took the inexpensive route since there were hardly any laboratories available that allowed you to manipulate objects. Everything was just combined into one big object. After a painstaking process of mastering blender we were able to model our laboratory in accordance with the specifications provided by the ASTM.

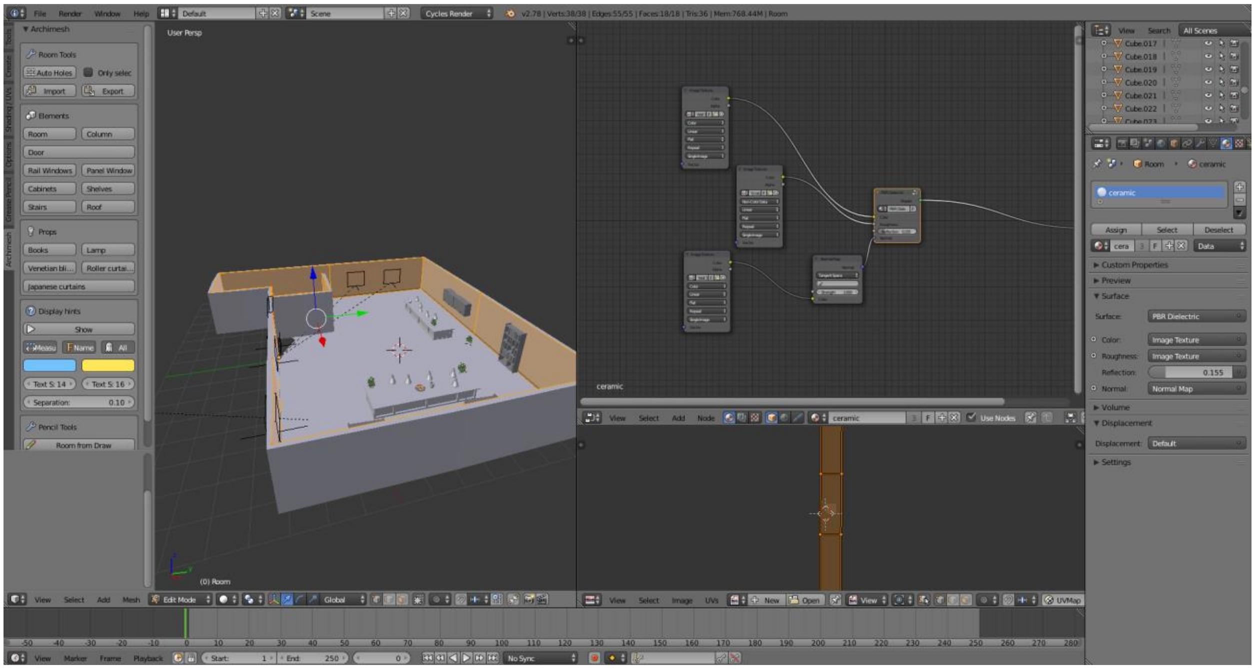


Fig. 9 Laboratory modelled in blender

Each object present in the laboratory will have a certain amount of reflection depending if they are either diffuse, glossy, glass or anisotropic bidirectional scattering distribution function (BSDF). Some objects may need a Fresnel value as real life objects do not fully reflect light.

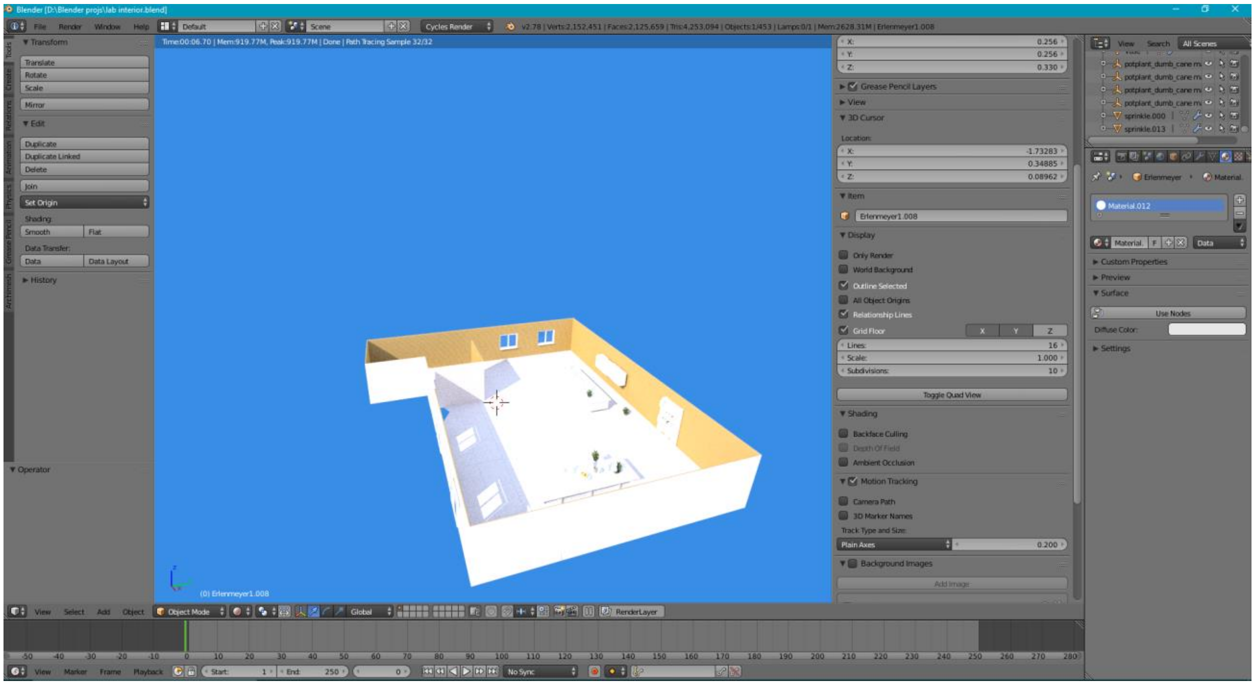


Fig. 10 Rendered model

After adding the appropriate textures for the floor, wall and the objects this is the rendered version of the laboratory. To import this into unity we need to convert it into .fbx. Sadly, blender does not save the textures while saving the file as .fbx. This in turn causes unity to just give a default white texture to each object in the scene. So we have to import the textures separately.

After importing the textures and laboratory we proceeded to add the appropriate scripts, prefabs, libraries and more to the laboratory using Unity. To use the game in VR we need to add the appropriate library depending on what VR kit we bought. Since we are using the HTC Vive we will have to add the SteamVR library to allow development and debugging in VR. Additionally, we need to set up the chaperone system that defines that boundaries of the play space area. Some scripts had to be written so that the water can stay in the beaker and the beaker would not overflow if it reaches the maximum amount of volume the beaker can hold.

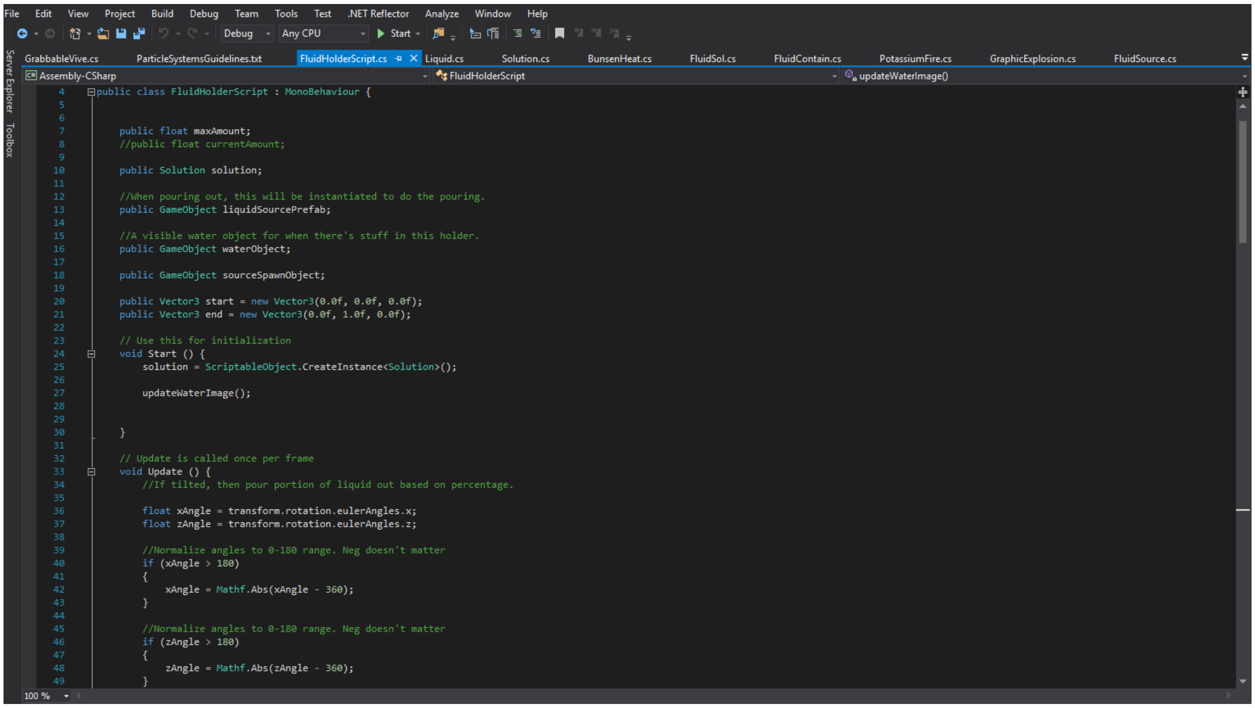


Fig. 11 Fluid holder script

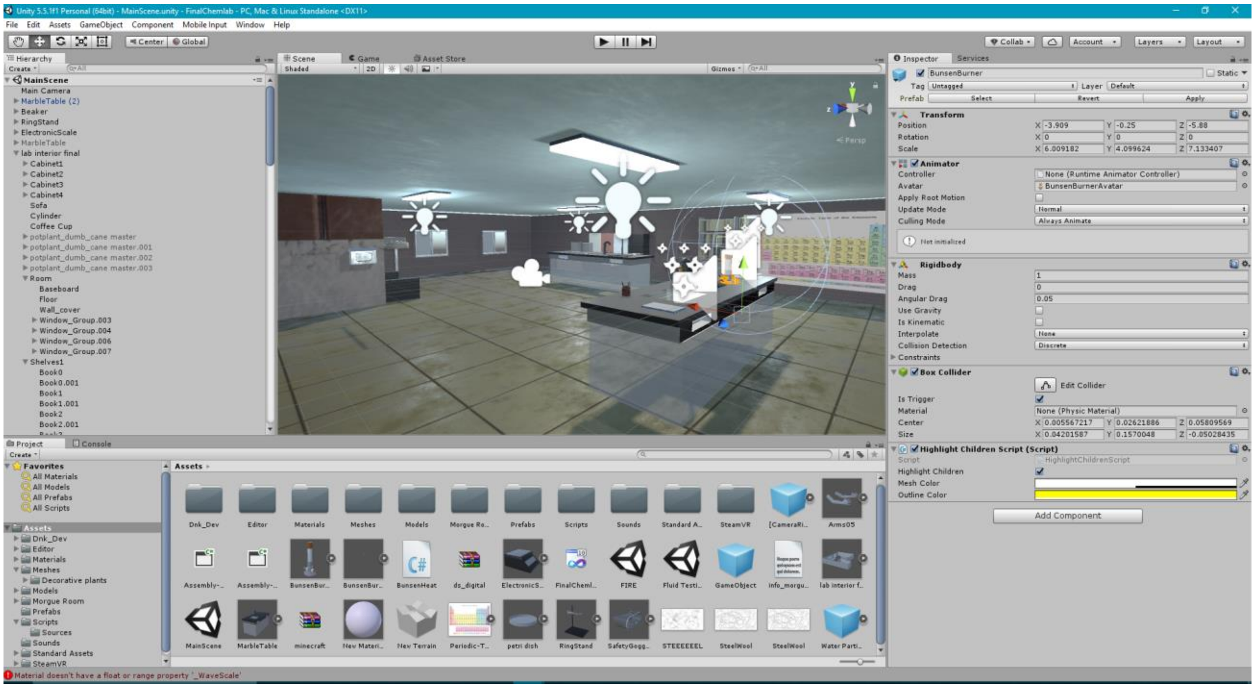


Fig. 12 Final Scene

4 Chapter 4: Results and discussion

4.1 Results

After importing and setting up everything we were able to make a fully working laboratory that would allow a user to perform experiments and face scenarios that they would normally face in a real life laboratory. Moreover, the user can perform experiments such as burning steel and other objects to check their weight or to see the appropriate flame color. To make the objects accessible by the VR controllers we used a combination of primitive colliders such as box, sphere or cube and mesh colliders on various objects such as the ring stand, beaker. Furthermore, there are faucets available in the scene where the user can fill a beaker or even throw water from the beaker. In addition, the asset store was filled with assets that can simulate fluid like behaviors. Although, most of the fluids were made for graphical purposes instead of using the fluid for logic. Also, most of the fluids available in the asset store are two dimensional (2D).

However, fluvio provided three dimensional (3D) fluids especially catered for VR purposes. Unfortunately, fluvio had some problems in storing the fluid in a beaker as it did not hit the appropriate mesh collider or even if it did the fluid would never accumulate in the object. Alternatively, the particle system that is already incorporated in Unity was used to fill the beaker which could not clearly emulate what we wanted as the particle system was more two dimensional (2D). After tinkering it around for a while we ended up adding a texture map and water materials to make the particle system look and behave like a 3D water molecule. Various assets were used to make the laboratory possible. The main assets that were used:

- Beaker

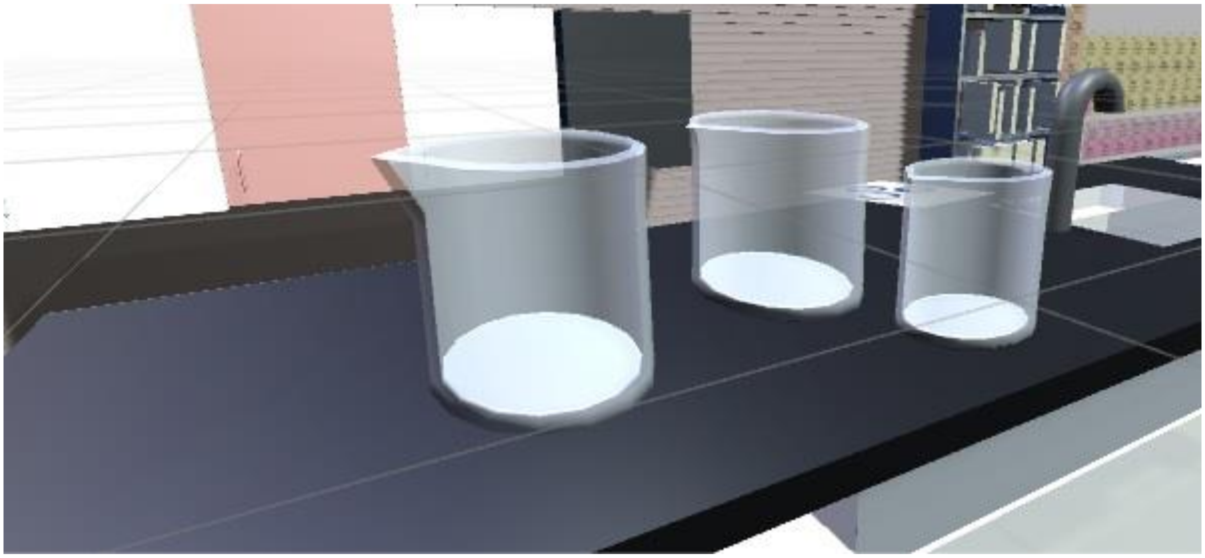


Fig. 13 Several beakers

Made some beakers that consist of a collider, rigidbody and a game object called source spawn point to allow water to fill in the beaker.

- Conical flask

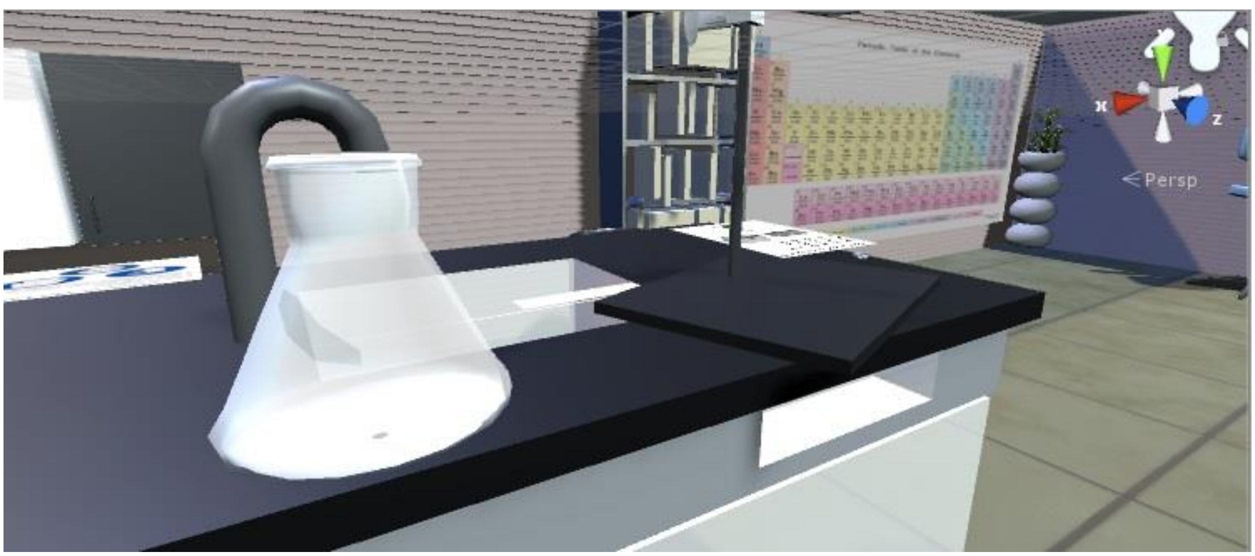


Fig. 14 Conical flask

It has similar properties as the beaker.

- Ring stand

Fig. 15 Ring stand

A ring stand is used to clamp glassware like beakers or conical flask in place so it does not fall down.

- Faucet

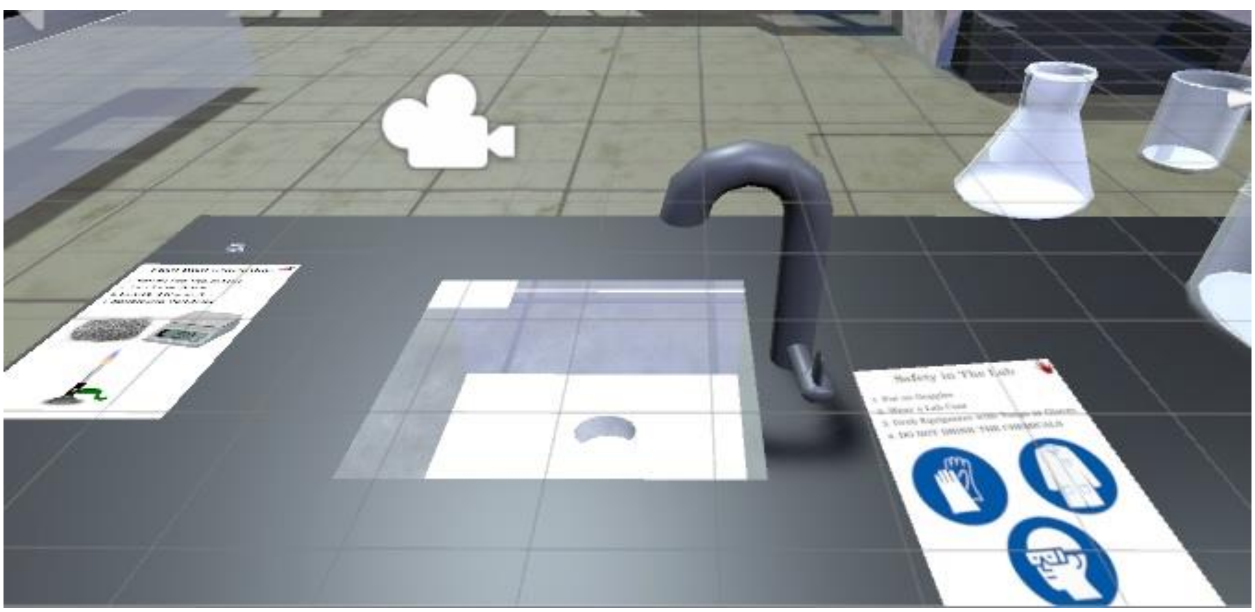


Fig. 16 Faucet attached to the marble table

Allows the user to fill water in the beaker or conical flask.

- Laboratory Sheets(obtained online):

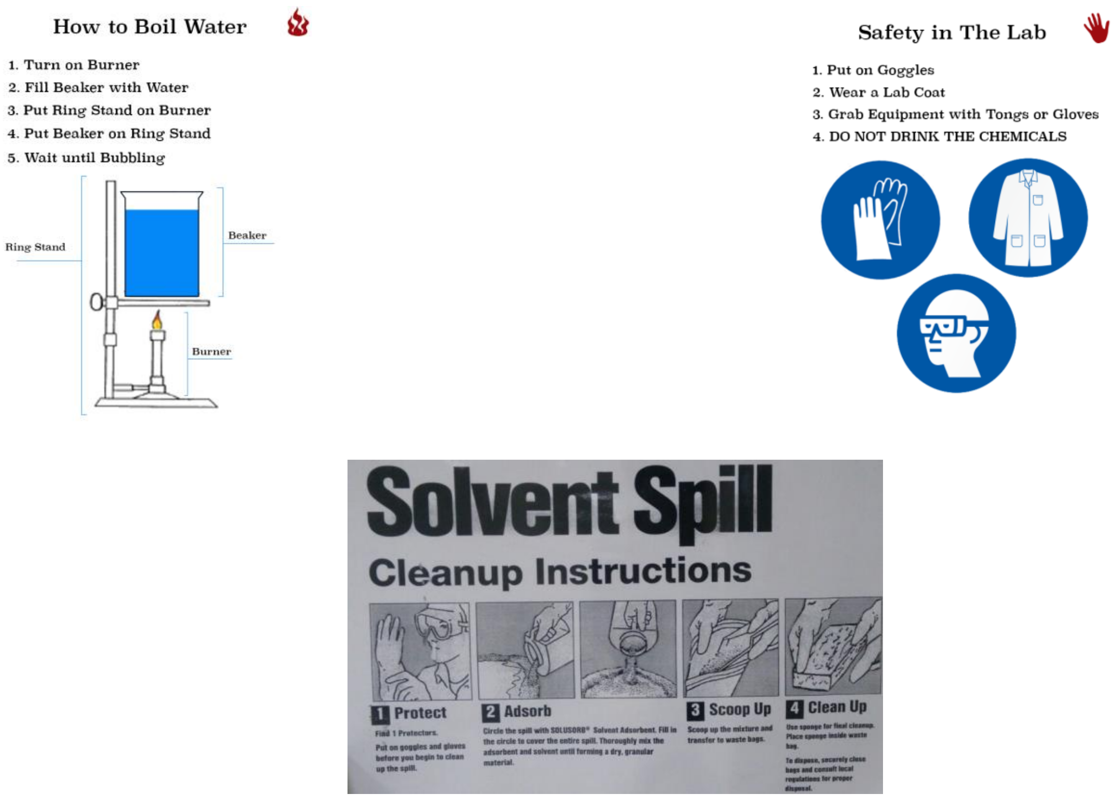


Fig. 17 Experiment and safety guidelines

Laboratory sheets explain the steps required to perform an experiment and how the user should follow safety procedures.

- Fire extinguisher(obtained from Unity asset store):

Fig. 18 Fire extinguisher in action

The fire extinguisher plays an essential role in our demo as it is used to demonstrate what the user should do if a fire breaks out.

- Electronic scale:

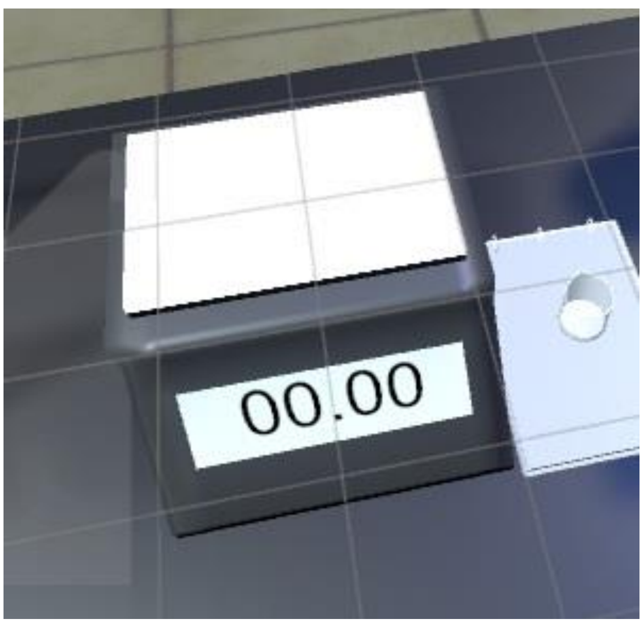


Fig. 19 Electronic Scale to measure weight of objects

- Safety goggles:

Fig. 20 Safety goggles

Safety goggles is a form of protective of eyewear that protects the users' eyes from splashes, acid vapors and particulates from hitting the eyes.

Refer to Fire script in appendix for more details regarding the flow chart on the next page.

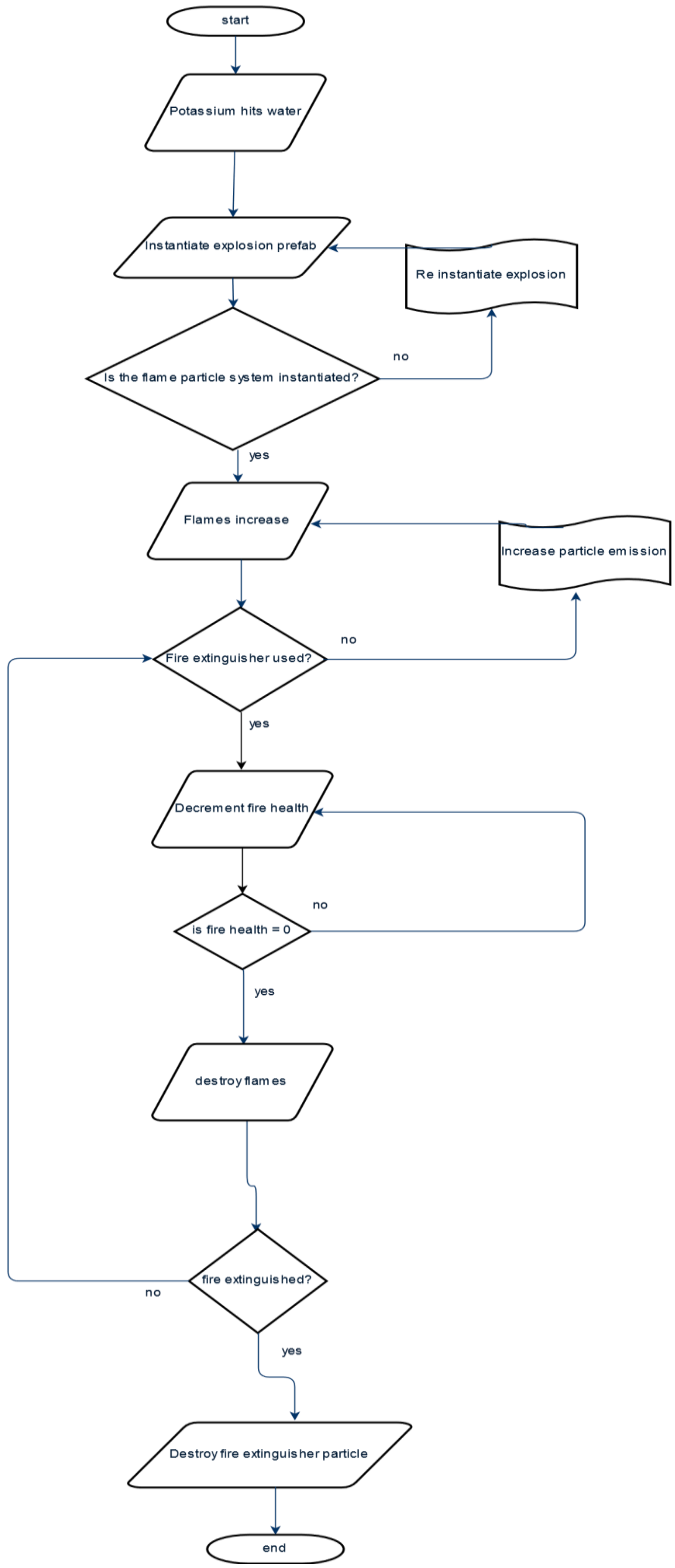


Fig. 21 Fire flowchart

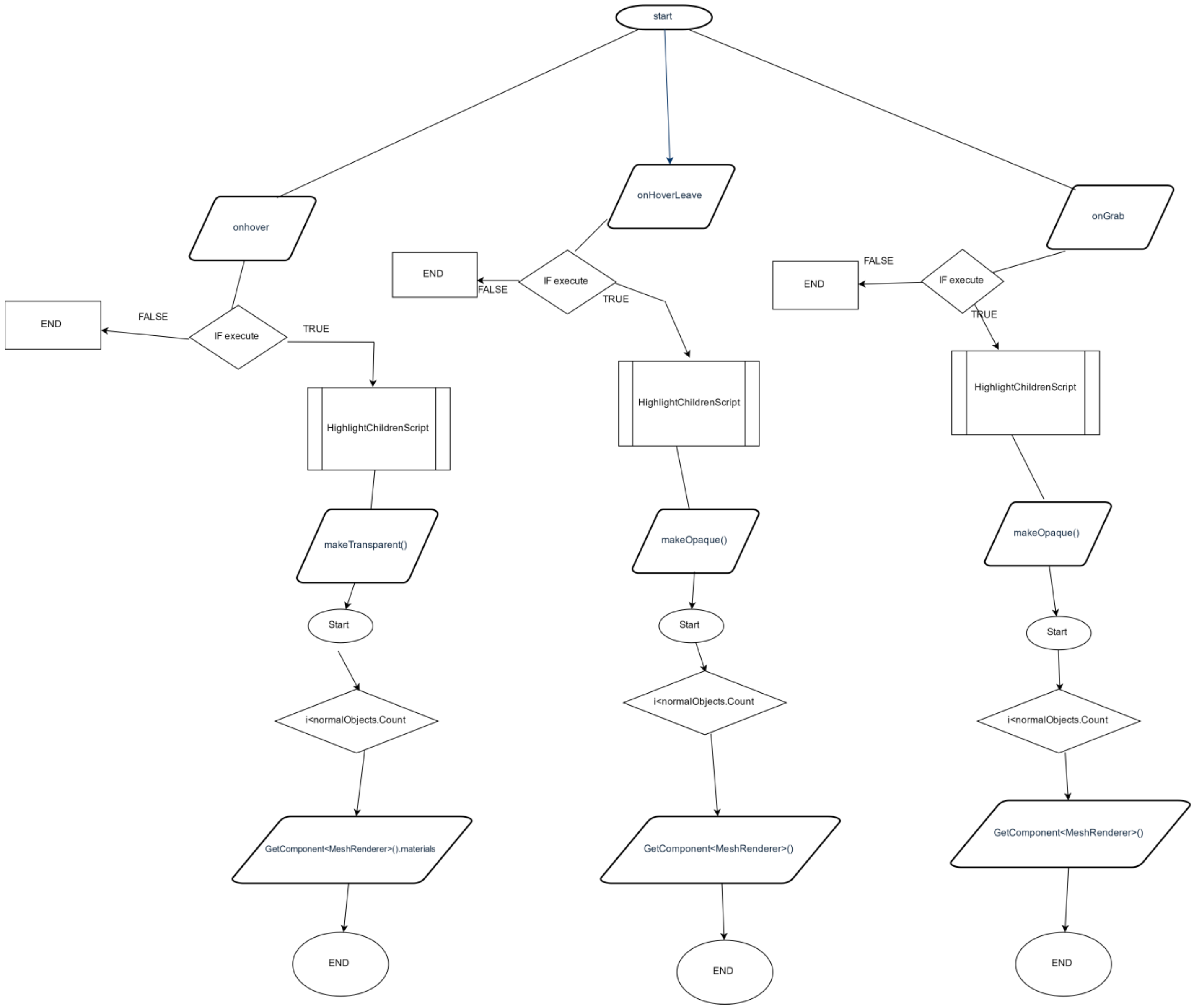


Fig. 22 Grabbable flowchart

Refer to GrabbableVive and Highlightchildren script in the appendix.

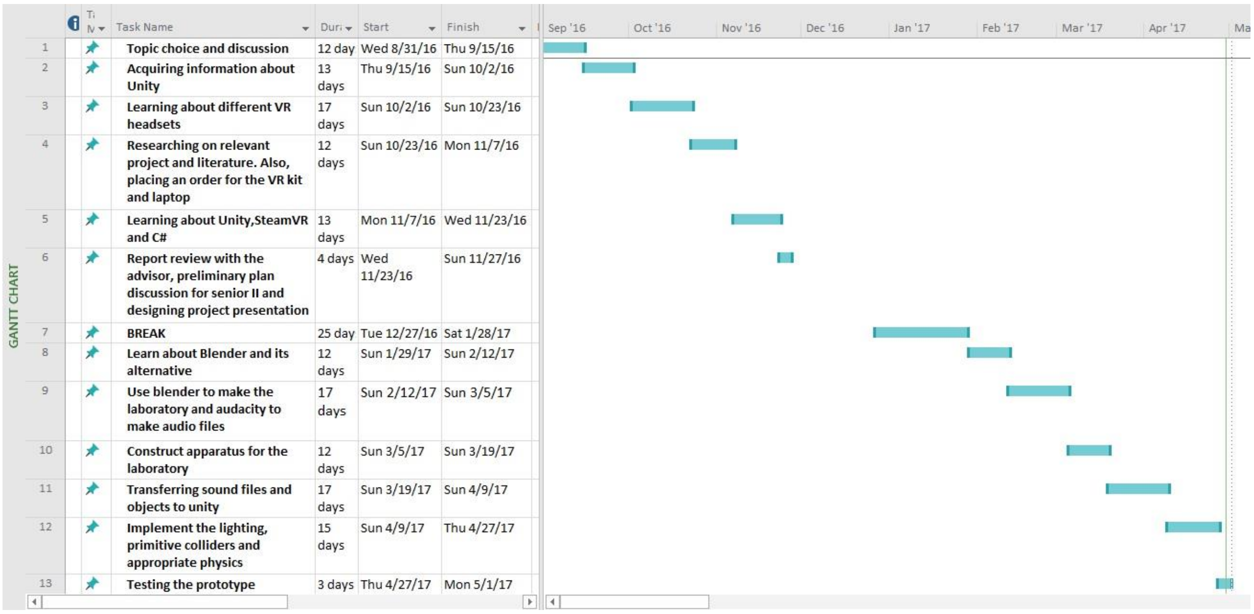
4.2 Discussions

There were various problems in modelling of the laboratory plus in establishing the interactivity of the laboratory. Blender has a very unorthodox interface which at first glance looks extremely complicated. After rigorously learning more about the user interface we were able to model a laboratory that mimics like a real life laboratory. Moreover, while developing in Unity the scene may start acting all strange as it moves your object in another direction. Also, debugging the scene is not currently yet supported in the editor. Therefore, you need to connect the Vive to debug each time you make a change to the scene.

5 Chapter 5: Project Management

5.1 Tasks, Schedule and Milestones

After senior design project I (SDP) steps were taken to rewrite a new schedule and build up upon the objectives set in SDP I. The scheduling included the time frame agreed upon which contained 10 sessions to maintain our time and keep it well managed. Also, at the end of each week, team members inform about the progress update on learning the software or language. At the end of the winter break, the first session commenced to discuss about the viability of using such a software and languages that needed to be learnt. Moreover there was a brainstorming session to find suitable ways to model the laboratory and integrate it in VR. The discussion was fruitful with many brilliant ideas, we discussed the possibility of making the laboratory model in blender rather than using Unity as blender provided more options for customization of the room. The discussion led to various methods as to how we will incorporate our laboratory in VR, whether it will need OpenVR or SteamVR would suffice to run the game. Furthermore, the need for optimization were also discussed as to whether the game will consume a lot of memory or affect performance. At the end of each session we diligently gathered all the probable solutions to the issues we were facing and tried approaching the problem from another perspective. The minutes of each meeting and what each member of the group should work on were kept weekly. Progress updates were notified to the advisor and team members at the end of each week or 2 weeks depending on the task. Then, other issues faced plus updates to the tasks are to be discussed in the next meeting. The minutes of each following meeting were duly noted and discussed in later meetings. The workflow was very successful and our collaboration to succeed in this project was our main concern. The schedule is as follows:



	Task Name	Duration	Start	Finish
1	Topic choice and discussion	12 day	Wed 8/31/16	Thu 9/15/16
2	Acquiring information about Unity	13 days	Thu 9/15/16	Sun 10/2/16
3	Learning about different VR headsets	17 days	Sun 10/2/16	Sun 10/23/16
4	Researching on relevant project and literature. Also, placing an order for the VR kit and laptop	12 days	Sun 10/23/16	Mon 11/7/16
5	Learning about Unity,SteamVR and C#	13 days	Mon 11/7/16	Wed 11/23/16
6	Report review with the advisor, preliminary plan discussion for senior II and designing project presentation	4 days	Wed 11/23/16	Sun 11/27/16
7	BREAK	25 day	Tue 12/27/16	Sat 1/28/17
8	Learn about Blender and its alternative	12 days	Sun 1/29/17	Sun 2/12/17
9	Use blender to make the laboratory and audacity to make audio files	17 days	Sun 2/12/17	Sun 3/5/17
10	Construct apparatus for the laboratory	12 days	Sun 3/5/17	Sun 3/19/17
11	Transferring sound files and objects to unity	17 days	Sun 3/19/17	Sun 4/9/17
12	Implement the lighting, primitive colliders and appropriate physics	15 days	Sun 4/9/17	Thu 4/27/17
13	Testing the prototype	3 days	Thu 4/27/17	Mon 5/1/17

Fig. 23 Scheduling for senior I & II

5.2 Resources and Cost Management

The project resources and materials are vital to our plan. We maintained the limited budget by choosing between different types of VR kits and Computers/laptops. First, the best option that suited our needs for this project while being affordable in our limited budget was the HTC Vive. Next, we had to decide whether to get a PC or a laptop. Although a custom made PC would have been a cheaper option but for the sake of convenience and ease we decided to go with a laptop that is VR ready (please see below for the specifications). The cost estimation for the needed resources were successfully well kept within our allotted project budget.

Final Design Component	Cost
HTC Vive kit	3,800 AED
Laptop : MSI GT62 VR 6RE DOMINATOR PRO, VR READY 32 GB RAM , 1 TB HARD DISK Nvidia GeForce® GTX 1070 8 GB GDDR5 , Intel Core i7-6700HQ Quad Core Processor (2.6-3.5GHz)	8,999 AED
Total Price	12,799 AED

Table. 2 Cost Estimation

5.3 Lessons Learned

Throughout this project we have stumbled at some phases while, other phases have been very productive. First and foremost, researching has made us gain information about the current state of the art VR headsets available or being made. Furthermore, it made us realize how VR is a disruptive technology and will change the face of not only industries, but also academic institutions.

In addition, we realized that combining all the elements together in a game engine is easier said than done. Since the object may get some loss during the conversion to import as an asset or worse yet corrupted. Furthermore, a game engine does not give you the freedom to model something intricate from scratch. So our best bet is using a 3D modelling software such as blender or maya. Another problem we faced was that game engines only accept specific file formats such as .fbx and do not completely import the textures from the file.

One of the most important things we learnt was that time management and teamwork skills play a vital role during this stage and will significantly increase during the onset of the senior design project. We have learnt how to distribute tasks among team members and how to bounce ideas of each other during brainstorm sessions. Additionally, after the brainstorm sessions we provided each other with solutions to the issues once they arise during modelling, audio sampling or in the game engine.

6 Chapter 6: Impact of the engineering solution

6.1 Economical, Societal and Global

Economically, VR will have drastic impacts on lots of industries, ranging from healthcare and education, all the way to entertainment and commerce. An example is the increase in the ease of conceptualization which the real estate industry and remodeling industry are based on. Another example is how experiential marketing will eliminate barriers of entry into commerce.

Society will also reap innumerable benefits from VR not the least of which will be the relative ease of learning new skills. For example surgical students can learn without ever having to cut into a cadaver itself and difficult fields such as rocket science, astrophysics, etc. will be easier to comprehend through virtual hands-on experiences as compared to standard two-dimension teaching practices. Moreover, virtual environments can be tailored to represent everyday challenges that would aid in rehabilitation of veterans with post-traumatic stress disorder (PTSD), enable high-functioning autism patients to be more effective at job interviews etc.

Globally, VR will have a huge impact as we are on the precipice of a revolution that will alter how we live, work and play. Fully-immersive environments and surreal experiences are just some of the ways that virtual reality is going to change the world [6]. This project will be tremendously helpful in the future as it will help uncover the potential of using VR for education, training, modelling and more.

6.2 Environmental and Ethical

Some people may claim that they are placing the companies that provide safety training out of business since every experiment is being done in VR. This in turn increases unemployment of those people working at such organizations.

6.3 Other Issues

VR has other issues such as wearing it for a long time may cause you to feel dizzy or uncomfortable. Since the headset may end up blocking blood flow to your brain. Moreover, some people may experience motion sickness in VR due to fast moving objects. Age also plays a factor as to why motion sickness occurs to the individual. In addition, VR may cause eyestrain and dry eye problems or as optometrists would like to call convergence-accommodation reflex. Our eyes usually converge and focus on the same point in the real world. But in VR, our eyes only focus on the screen from a distance and convergence of objects change as the user moves around.

7 Chapter 7: Conclusions and Recommendations

7.1 Summary of Achievements of the Project Objectives

The following achievements were made:

- Establishing a laboratory with the appropriate apparatuses.
- Created a scenario where the user will take necessary precaution.
- Developed appropriate light maps from beforehand to improve performance.
- Used primitive colliders and mesh collider to add collision to objects.
- Optimized the laboratory so that memory isn't completely used while the game is running.

7.2 New Skills and Experiences Learnt

This project taught us invaluable skills and experiences such as:

- Learning how to create games in a game engine.
- How to write gaming scripts in languages such as Javascript, boo and C#.
- Discovering the problems indie developers or any game developer face while making the game.
- Exploring different VR platforms and how vast VR development is.
- A proper team is required to make a full-fledged game and the individuals must be competent.
- How to model things in the real world in 3D modelling programs.
- The difference between various VR headsets.
- How to filter out noise from sound samples and in the creation of sounds from different objects such as a spoon tapping on a cup.

7.3 Recommendations for Future Work

There are always room for improvements. The project can be improved by adding more scenarios portraying health and safety to students such as if a chemical falls on your hand or if you are in contact with a poisonous gas. Also, we can add appropriate fume hoods and ventilation so the laboratory looks more realistic. Moreover, this project is just a proof of concept to show the infinite possibilities of how VR can be used not only in educational training but also, training in general for a number of jobs.

REFERENCES

[1] Johnson, Eric. "What Are the Differences among Virtual, Augmented and Mixed Reality?" *Recode*. Recode, 27 July 2015. Web. 04 Apr. 2017.

[2] "Virtual Reality Educational Benefits | Immersive VR Education." Immersive VR Education RSS2. N.p., n.d. Web. 29 Nov. 2016.

[3] "Lecture VR | Immersive VR Education." Immersive VR Education RSS2. N.p., n.d. Web. 29 Nov. 2016.

[4] "Take Your Students on a Virtual Reality Adventure from Your Classroom." Nearpod - Take Your Students on a Virtual Reality Adventure. N.p., n.d. Web. 29 Nov. 2016.

[5] Googleforedu. "Google Expeditions." Google. N.p., n.d. Web. 29 Nov. 2016.

[6] Adams, R.L. "Five Ways Virtual Reality Will Change The World." *Forbes*. Forbes Magazine, 17 Oct. 2016. Web. 29 Apr. 2017.

APPENDIX

Fire Script:

```
using System.Collections;

using System.Collections.Generic;

using UnityEngine;


public class fire : MonoBehaviour {

   public float FireHealth = 10;

   public float MaxFireHealth = 50;

   public float HealthRegen = 0;

   public bool IsOnFire = true;

   public ParticleSystem Fireextparticle;


   void Start() { }


   public GameObject explosion;

   public GameObject fireyy; // drag your explosion prefab here

                         //Fireextparticle = GetComponent<ParticleSystem>();


   void OnCollisionEnter(Collision col)

   {   if (col.gameObject.name == "Potassium")

       {

           GameObject expl = Instantiate(explosion, transform.position, Quaternion.identity) as
GameObject;

           Destroy(gameObject); // destroy potassium
```

```
            GameObject firedup = Instantiate(fireyy, transform.position, Quaternion.identity) as
GameObject;

            Destroy(expl, 2); // delete the explosion after 3 seconds

        }

}

    void OnParticleCollision(GameObject other)

    {

        if (IsOnFire)

        {

             FireHealth -= 1.0f;

            if (FireHealth <= 0)

            {

                IsOnFire = false;

                if (Fireextparticle.isPlaying)

                {

                    Destroy(Fireextparticle);

                    Destroy(fireyy);

                    Destroy(other);

                }

                // transform.GetComponent<ParticleEmitter>().emit = false;

                //transform.GetComponent<ParticleSystem>().Clear;

                // other things to do when fire goes out

               //ParticleSystem.Emission.rate //Destroy(Fireextparticle);

            }

        }
```

```
    }


    // Update is called once per frame
    void Update () {
        if (IsOnFire)
        {
            FireHealth += Time.deltaTime * HealthRegen;
            if (FireHealth > MaxFireHealth)
            {
                FireHealth = MaxFireHealth;
            }
        }
    }
}
```

GrabbableVive:

```
using UnityEngine;
using System.Collections;

//All objects that can be grabbed need to have this.
[RequireComponent(typeof(HighlightChildrenScript), typeof(Rigidbody), typeof(Collider))]
public class GrabbableVive: MonoBehaviour {

    public bool createDialogue;
    private int numTimesPickedUp = 0;
    public string firstTouchDialogue;

    public bool isTool;
    public bool isActive;

    public float soundThreshold = .5f;

        public void onHover()
    {
        GetComponent<HighlightChildrenScript>().makeTransparent();
    }

    public void onHoverLeave()
    {
        GetComponent<HighlightChildrenScript>().makeOpaque();
```

```
    }


    public void onGrab()
    {
        numTimesPickedUp++;
        GetComponent<HighlightChildrenScript>().makeOpaque();
    }


    public void OnCollisionEnter(Collision other)
    {
        //Debug.Log(other.gameObject.name);
        /*
        if (Vector3.Magnitude(GetComponent<Rigidbody>().velocity) > soundThreshold)
        {
            GetComponent<AudioSource>().Play();
        }*/
    }
    public void onRelease()
    {
    }
}
```

HighlightChildren:

```
using UnityEngine;

using System.Collections;

using System.Collections.Generic;

public class HighlightChildrenScript : MonoBehaviour

{


  public bool highlightChildren = true;


    public Color meshColor = new Color(1f, 1f, 1f, 0.5f);

    public Color outlineColor = new Color(1f, 1f, 0f, 1f);


    //A list containing arrays of materials. Index of list refers to the gameObject.

    private List<Material[]> normalMaterials;

    private List<GameObject> outlineObjects;

    private List<GameObject> normalObjects;


    // Use this for initialization

    public void Start()

    {

        normalMaterials = new List<Material[]>();

        outlineObjects = new List<GameObject>();

        normalObjects = new List<GameObject>();


    // Set all the original materials into normalMaterials.
```

```
MeshRenderer[] meshRenderers = new MeshRenderer[1];

if (highlightChildren) {

  meshRenderers = GetComponentsInChildren<MeshRenderer>();

}

else {

  meshRenderers[0] = this.GetComponent<MeshRenderer>();

}

    for (int i = 0; i < meshRenderers.Length; i++)
    {
        normalObjects.Add(meshRenderers[i].gameObject);

        Material[] materials = meshRenderers[i].materials;

        normalMaterials.Add(new Material[materials.Length]);

        //For this mesh renderer, store all materials.
        for (int j = 0; j < materials.Length; j++)
        {
```

```
            normalMaterials[i][j] = new Material(materials[j]);
        }

        // Create a copy of this object for every child, this will have the shader that makes the real outline
        GameObject outlineObj = new GameObject();
        outlineObj.transform.parent = meshRenderers[i].gameObject.transform;

        outlineObj.transform.position = meshRenderers[i].gameObject.transform.position;
        outlineObj.transform.rotation = meshRenderers[i].gameObject.transform.rotation;
        outlineObj.transform.localScale = new Vector3(1f, 1f, 1f);
        outlineObj.AddComponent<MeshFilter>();
        outlineObj.AddComponent<MeshRenderer>();
        Mesh mesh;
        mesh = (Mesh)Instantiate(meshRenderers[i].gameObject.GetComponent<MeshFilter>().mesh);
        outlineObj.GetComponent<MeshFilter>().mesh = mesh;

        materials = new Material[materials.Length];
        for (int j = 0; j < materials.Length; j++)
        {
            materials[j] = new Material(Shader.Find("Stencil/Outline"));            }
        outlineObj.GetComponent<MeshRenderer>().materials = materials;

        outlineObj.SetActive(false);
```

```
            outlineObjects.Add(outlineObj);
        }
    }


    public void makeTransparent()
    {
        // Set the transparent material to this object
        for (int i = 0; i < normalObjects.Count; i++)
        {
            Material[] materials = normalObjects[i].GetComponent<MeshRenderer>().materials;
            int materialsNum = materials.Length;
            for (int j = 0; j < materialsNum; j++)
            {
                materials[j] = new Material(Shader.Find("Outline/Transparent"));
                materials[j].SetColor("_color", meshColor);
            }


            //Activate the outline objects
            for (int j = 0; j < outlineObjects.Count; j++)
            {
                outlineObjects[j].SetActive(true);
            }
        }
    }
    public void makeOpaque()
```

```
    {
        for (int i = 0; i < normalObjects.Count; i++)
        {
            //Debug.Log(i);
            normalObjects[i].GetComponent<MeshRenderer>().materials = normalMaterials[i];

            //Activate the outline objects
            for (int j = 0; j < outlineObjects.Count; j++)
            {
                outlineObjects[j].SetActive(false);
            }
        }
    }
}
```