Development of USTH interactive virtual world using Unreal Engine

DO Duy Huy Hoang

University of Science and Technology Hanoi

ICT Department

2019-06-18

Declaration

I declare that this thesis is an original report of my research under the guidance of Dr. Tran Giang Son, has been written by me and has not been submitted for any previous degree. The experimental work is almost entirely my own work; the collaborative contributions have been indicated clearly and acknowledged. Due references have been provided on all supporting literatures and resources. I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification.

Acknowledgement

Firstly, I would like to express my thanks to my research supervisor Dr. Tran Giang Son for giving me the opportunity to do research and providing invaluable guidance during this internship. Secondly, my sincere thanks also goes to Dr. Nghiem Thi Phuong for her encouragement, insightful comments and hard questions during the research. Thirdly, I thank my fellow labmates, friends in ICTLab: Mau Nguyen, Linh Loc, An Luu, Duc Pham, Trung Bui for the stimulating discussions and for all the fun we have had in the last three years. I also very appreciate to have been a student at USTH where all professors and staffs are always willing to support their students. Besides, I would like to thank the Unreal Engine and Sketchup community for answering and providing many valuable knowledge I asked during the research. Last but not least, I express my deep and sincere gratitude to my parents who raise me up throughout my life by their patient and their love.

List of Tables

List of Figures

Contents

I. Introduction

1.1 Context and Motivation

A virtual world or a digital world is a computer-based online community environment that is designed and shared by individuals, or may be populated by many users who can create a personal avatar so that they can interact in a custom-built, simulated world. These avatars are graphically rendered using computer graphics imaging (CGI) or any other rendering technology which can make people feel like they are in the real world. Individuals control their avatars using input devices like the keyboard, mouse and other specially designed command and simulation gadgets.

Today's virtual worlds are purpose-built for entertainment, social, educational, training and various other purposes. It is being more better than the real world. So the virtual reality has take over everything. For instance the illustrations showed in a game are more better than the reality we live in. It provides the gamer a whole new experience by taking over his imaginations it's as if we are playing the real game with all those real gaming characters in the real life. Virtual world is bringing out all our imaginations so that we can experience it in reality. Similarly for the automobile industry we can take a virtual ride of the desired vehicle we want to buy which provides a whole new dimension to the world of buying things.

Moreovers, virtual world has been adopted in education for teaching and learning situations. Students are able to interact with each other and within a three dimensional environment. Students can also be taken on virtual field trips, for example, to museums, taking tours of the solar system and going back in time to different eras. In this internship, I'll mainly focus on entertainment-based virtual world as it's closer to real world we are living, bring a pleasant for us, especially with 3D environment.

To create a virtual world, there are some types of engine that allow us to make like Unreal Engine, Unity, Blender, etc…

In one of those engines, Unreal Engine is the most popular choice for many people, from beginer to expert, and also it known as the best engine for 3D architecture virtual world.

The Unreal Engine is a game engine developed by Epic Games, first showcased in the 1998 first-person shooter game Unreal. Although initially developed for first-person shooters, it has been successfully used in a variety of other genres, including platformers, fighting games, MMORPGs, and other RPGs. With its code written in C++, the Unreal Engine features a high degree of portability and is a tool used by many game developers today, with it being source-available. The engine can be used to create three-dimensional, two-dimensional, virtual reality, and augmented reality games, as well as simulations and other experiences.

From my perspective, Unreal Engine has tons of online instructional exercises to be found, archives with clean arrange, so it may be a great step for beginners. You can use both Blueprint and C++ lang so that If you are an artist having no programming experience, then blueprints are a big plus. You can get acquainted with its functioning easily. You can easily implement your game logics to see them running in your game. Also, with great community, when we have a problem, they will help us everywhere and every time. Furthermore, you only need to develop one application and it can be rebuilt in any other platforms.

In this work, I will focus on creating 3D models from scratch and also build a demo of 3D virtual world of USTH using Unreal Engine, whereas a player can walk through, and interact with some objects. This application can be used for celebrating 10 years of USTH, marketing, and also for remote students/parents have a virtual trips before applying to our university or any USTH student who is studying abroad.

1.2 Objectives

Develop an interactive 3D virtual world of USTH building from easily captured 2D images. Specific goals include:

- Collect a dataset of 2D images from cameras for different views of USTH
- Conduct fully 3D models from collected 2D images which will be a 3D model foundation for other researcher or developer to create others world of USTH
- Build a realtime and interactive 3D virtual world of USTH from constructed 3D models.
- Due to time limitation in this internship, the expectation of this project is to conduct totally six USTH floor 3D models and use 2D sample floor to create a 3D virtual world.

1.3 Thesis structures

The thesis will contain all the information necessary to reproduce the result. First, we will introduce briefly the definition, application of virtual world and the aim of my thesis in Chapter I - Introduction. Chapter III is *Materials and Methods*. In the Materials section, we will list all the dependencies/libraries and application we use, as well as the work breakdown description. Then in *Methods*, we dissect carefully everything we do, from creating model and building a fully interactive virtual world. After that, in Chapter *Result and Discussion*, we show some final pictures that previews the USTH virtual world, the dataset of USTH's view and the 3D models. Also we will explain all the small implementation details or resolve problems that are raised in other chapters. Finally is conclusion and future work.

II. State of the art

Intense competitions in computer related industries have lead to the development of new amazing technologies. This drastically improved 3D game engines, which was initially developed for computer games, to become more adaptable and versatile. 3D visualization can provides users with the real spatial information and the 3D models resembles the real world object. Nowaday, Selection of *3D Game Engines* is very important while developing a 3D game using *different art style*. There are a lots of 3D game engines such as *Unreal Engine, Cry Engine, Unity Engine,…*, these are all free to download and also have different specifications and features. This project will focus on *Unreal Engine* because of alot of advantages by using this engine. First of all Unreal Engine is free, and users have fully access to their source code. Therefore, Epic provides multiple official video tutorials, lots of free example projects and content, an extensive wiki and regular streams showing how to use latest features. And the most important part is *A visual scripting system for non-coders enables quick prototyping* or also known as *Blueprint*. Blueprints are authoring tools designed for non programmers so designers and other team members can help tweak and prototype. UE4's Blueprint scripts resemble flowcharts where each box represents a function or value, with connections between them representing program flow. This provides a better at-a-glance indication of game logic than a simple list of events, and makes complex behaviors easier to accomplish and games a lot faster to prototype. Comparing to others engine, for example *Unity*, **Blueprint** is a win for non-developer or a game developer beginer like me. Finnally, **Great Things Happen in Unreal Engine with Unreal Studio and SketchUp** - Sketchup with Unreal Engine are the best toolkit for 3D architecture project.

Figure 1: Sketch and Unreal[1]

III. Programs, Materials, Methodologies

3.1 Programs

3.1.1 Sketchup

SketchUp is a 3D modeling computer program for a wide range of drawing applications such as architectural, interior design, landscape architecture, civil and mechanical engineering, film and video game design. This program was used in this project in order to create and conduct a fully 3D models from scratch. Therefore, they also provide an open library called 3D Warehouse which SketchUp users may upload and download 3D models to share. The models can be downloaded right into the program without anything having to be saved onto your computer's storage and it is free so anyone can download files for use in SketchUp or even other software such as AutoCAD, Revit and ArchiCAD.

3.1.2 3DS Max

Autodesk 3ds Max is a professional 3D computer graphics program for making 3D animations, models, games and images. With this feature, the software will help me to correct the position of the texture and also to edit UV mapping of some models.

3.1.3 Substance Painter

Substance Painter gives me all the tools I need to texture my 3D assets and also this software can help me to bake a normal map and metallic map.

3.1.4 Unreal Engine

The Unreal Engine is a game engine developed by Epic Games, first showcased in the 1998 first-person shooter game Unreal. Although initially developed for first-person shooters, it has been successfully used in a variety of other genres, including platformers, fighting games, MMORPGs, and other RPGs. With its code written in C++, the Unreal Engine features a high degree of portability and is a tool used by many game developers today, with it being source-available. The engine can be used to create three-dimensional, two-dimensional, virtual reality, and augmented reality games, as well as simulations and other experiences.

This is the main program that I use to create a virtual world.

3.2 Materials

2D images with different views, panoramic pictures and videos of USTH are the most important materials on this project. Beside that A good computer which reaches the minimum requirement to run Unreal Engine and also can run Linux/MacOS/Window to export application.

3.3 Work flow

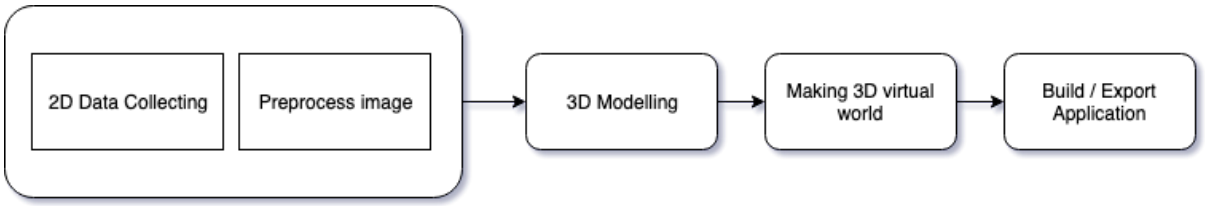

Figure 2: Work Flow Overall Diagram

Having three months internship at USTH ICT lab, my work will be divided into four main steps as the figure above to successfully build and create both USTH 3D models and virtual world. After organized some meetings with my supervisor about the project aims, we will start with taking pictures and capturing videos of USTH and do some preprocess data. 3D modelling will be the next step of this project and finally build the 3D virtual world - the final aim of this project. This is a visualization and a brief summary of the work, We shall be discussing these terms in the work breakdown figure below and in depth in the methodology section.

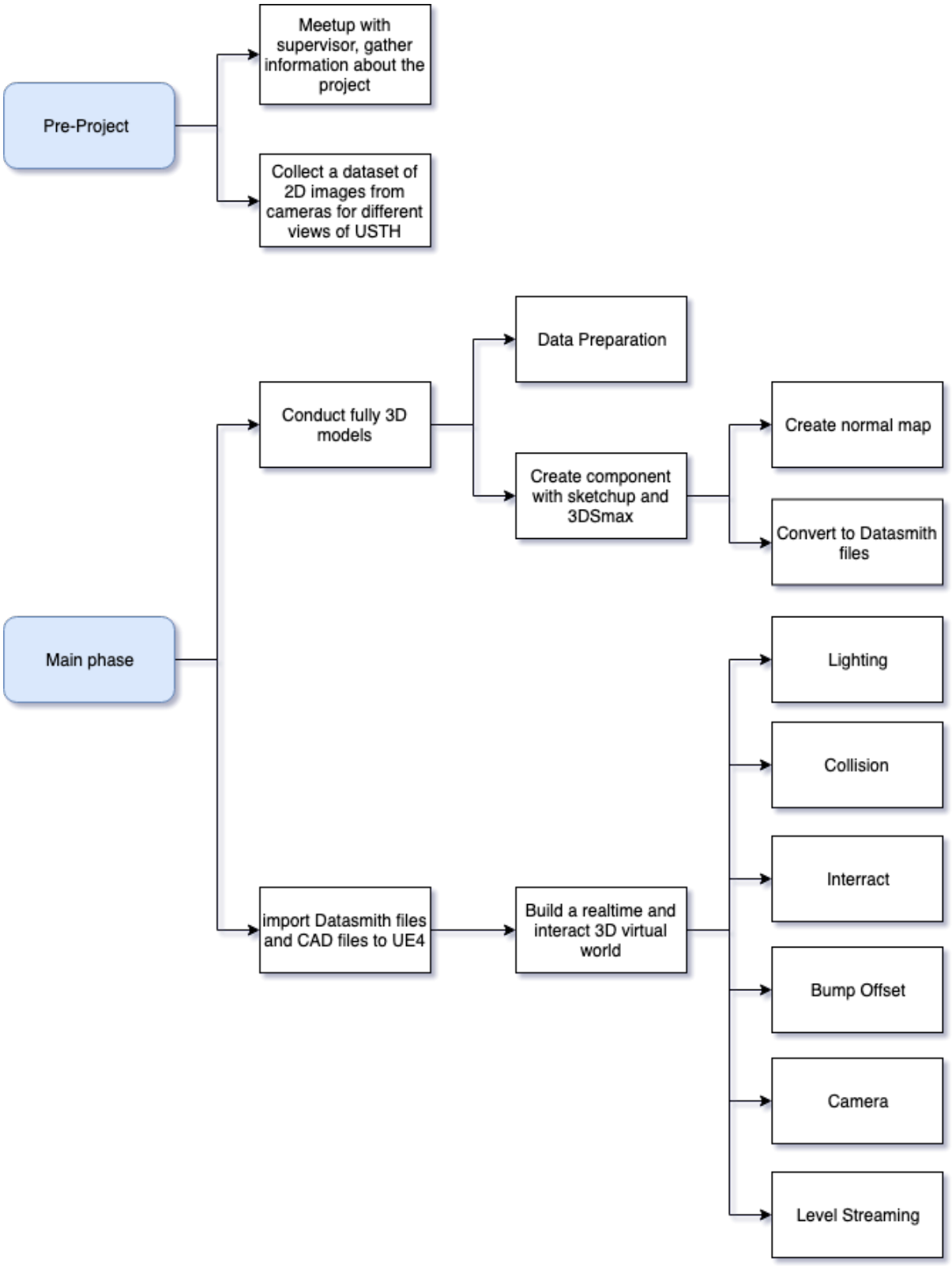

Figure 3: Work Breakdown Diagram

3.3 Methods

This section will explain all the methods of the work.

3.3.1 Pre-project

- **Conduct a meeting with the participant** of the Planning Phase, which involves setting expectations, articulating likely risk, etc.
- **Develop Work Breakdown Structure:** break the project into smaller, measurable of work packages with a work breakdown structure.

3.3.2 Input Data

- **Collect a dataset** of *2D images* USTH:
  - This work is the first step and is one of the important task.
  - Data will be 2D images, videos.
  - We will use a simple camera from a smartphone to perform this task. After taking pictures with different view and also furnitures of USTH, these photos will be classified to specific folder (e.g: Room 5/Lab ICT). This approach has two drawbacks howerver, it is time-consuming and easy to make mistakes.

3.3.3 3D Modelling

Conduct fully 3D models from collected 2D images which will be a 3D model foundations for other reaseacher or developer to create others world of USTH. To create a fully 3D models from scratch (2D images, videos, floor plan), this task will be breaked down into four steps as the figures below.

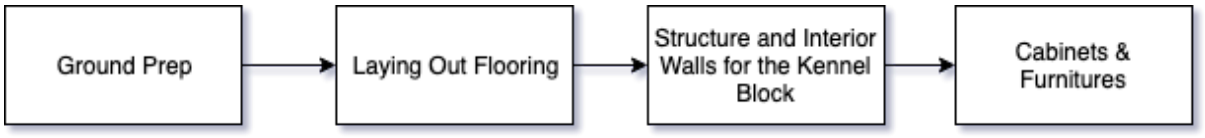

Figure 4: 3D modelling steps

a. Ground Prep / Floor plan: First of all we need to create a Floor Plans. When you install SketchUp, it also comes with Layout application. These application is designed to work together. After you insert a SketchUp models into a LayOut document, you can design the document to highlight your 3D model's best features. If you ever need to change or update your original SketchUp model, you can update the model in LayOut so that all the details are synched automatically. Ground Prep is also known as Schematic Design in Architecture, it is the first phase of design. The architect establishes the size, location, and relationships between all the spaces. Thus USTH provided me a 2D legacy floor plan images, but it is captured from a cemera intead of a digital floorplan, so i will need to transfer it into a digital version as the picture below.

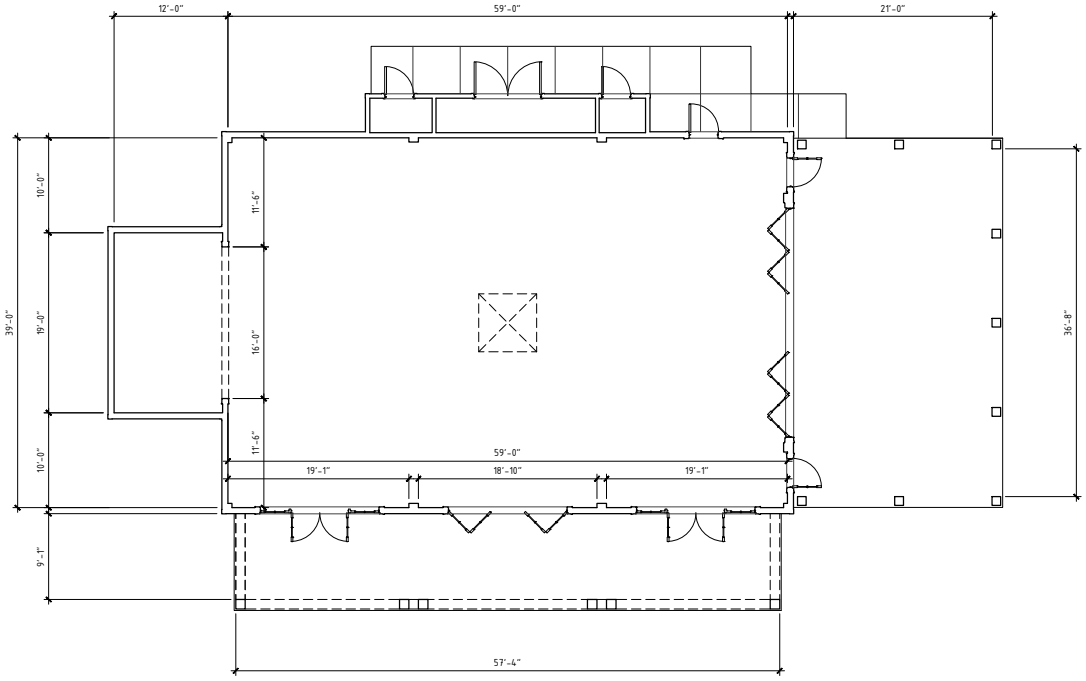

Figure 5: Floor Plan

b. Structure and Interior Walls: To structure and interior walls is quite simple thus we already have the digital floor plans version, Now by using *offset* tool, it will allow you to draw a offset of a perimeter, so in this case i can dictate how thick our walls is. And as the default parameter in Architecture field, i will set it equal to six inches.

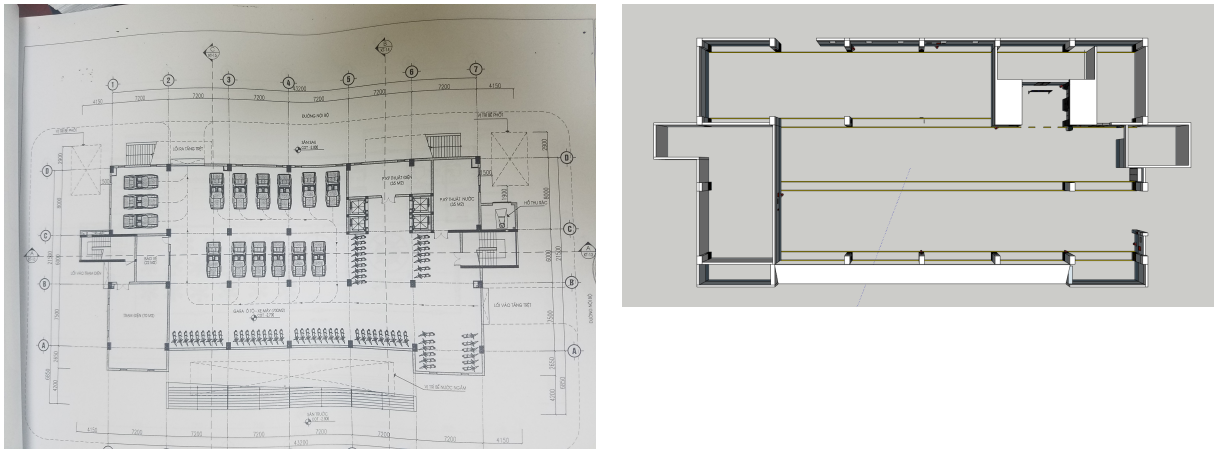

Figure 6: Stucture Walls

c. Furnitures: Finally in order to make the 3D model complete, we import sketchup textures, create custom materials and also add furniture objects. These furnitures can be found in the 3D warehouse store which comes with SketchUp when installed. Sometime I also need to create some special models in USTH thus it cannot be found in the store.

Figure 7: Adding Furnitures, Textures

Remind that in this internship, the final aim of this project is not only to create a virtual world but also to make USTH 3D model for other developer to use on their own project. So in this task, I will split into two smaller tasks, one is to create a fully 3D model, the other task is belonged to my work with Unreal Engine. After reading some papers, documents and doing some research over the internet, to improve the performance of Unreal Engine, Instead of importing a fully 3D models to Unreal Engine which may be slowed down or overflow memory, the 3D models files might be split into smaller part. For example, as I mentioned earlier, we will try to reduce the object as much as possible to increase performance, so i make as many as repeating elements as possible

and convert these model to *Filmbox* file type to import to Unreal Engine. We still keep the floor plan, walls and the basic hallway then import to UE4 but all the furnitures will be split into individual files so that we can only need to import these files once. These pictures below is an example that i create a door and student table using 3DSmax and then import back to UE4. This technique not only help me to improve the application performace using *Level Streaming method* but also by making individual object with 3DSMax, i can control the pivot of these object which cannot be done in Sketchup.

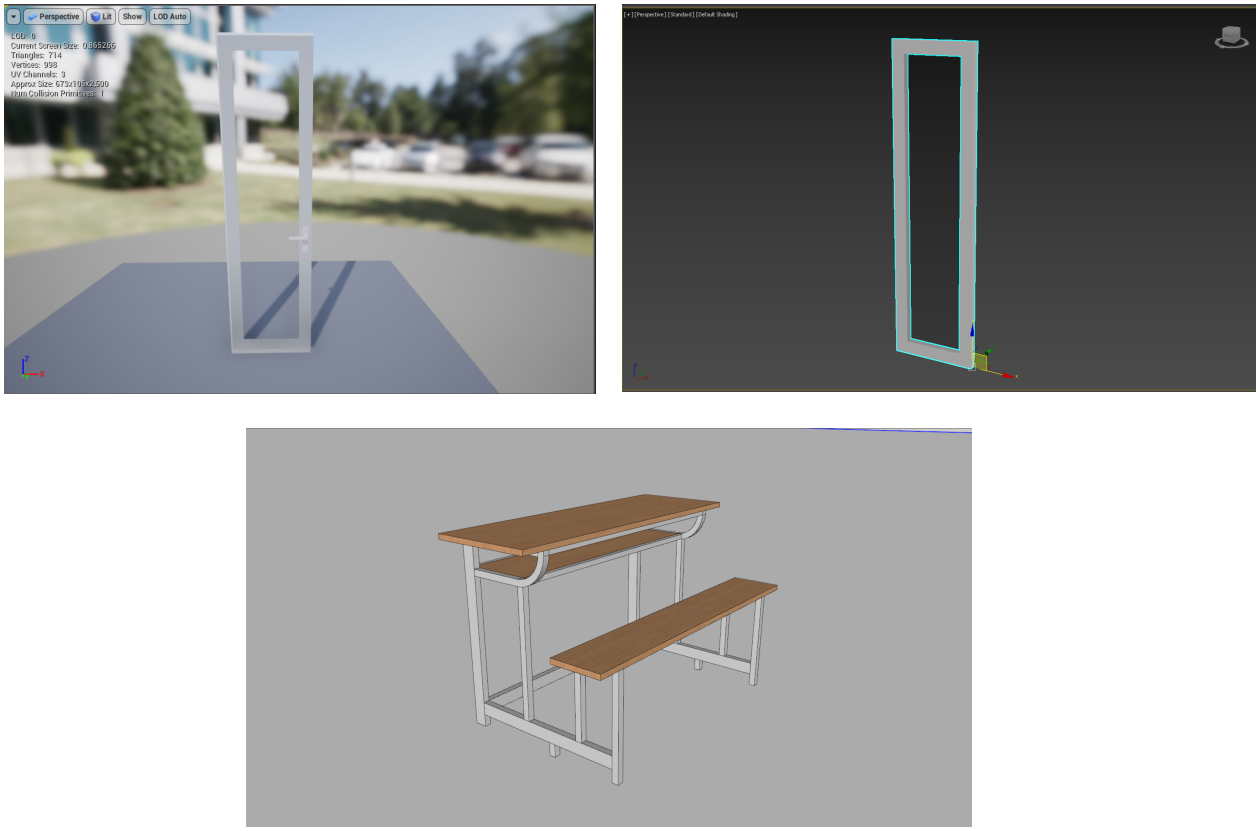

Figure 8: 3D object in UE4 and 3DSmax

d. Normal Map: To finish the 3D models and prepare models for Unreal Engine, the final step is to turn a model into a normal map for exporting to be used in 3rd party renderers. We use **Substance Painter** application and also a plugin called **CLF Normal**

Map Maker from Sketchup. It will paint the model as a normal map, export an image, and then revert all materials back to what they were.

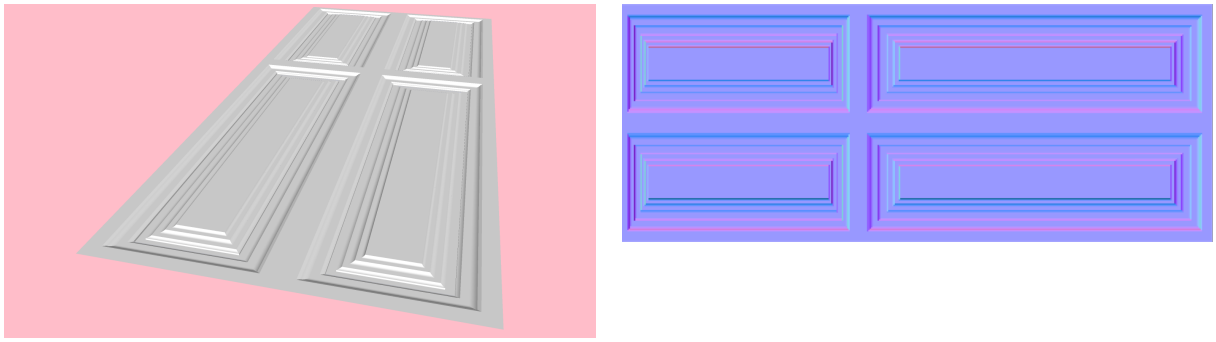

Figure 9: Normal map example

e. Preprocess data: Before we can get started working with SketchUp content in the Unreal Engine, we need to install the Unreal Datasmith Exporter plugin for SketchUp and able to export a scence from SketchUp as a *.udatasmith* file and also *.fbx* files, thus the previous application creates models in a .SKP file.

3.3.4 Unreal Engine and Creating USTH 3D virtual world

Build an interact 3D virtual world: We start making our 3D virtual world by adding player, light, interact, and some technique like *level streaming*…

Preparation: Unreal Engine can be dowloaded from the official website - it is free for everyone. As I mentioned before, to be able to run Unity, and create a new project, we'll need at least direct x11 for simulation of lighting, shadows on the scene.

Start with importing our 3D models to UE4. Datasmith brings structure data from an combination of source applications into Unreal Engine, frequently to fabricate steady representations and experiences around that data. In any case, routinely, whereas

you are working on building those visuallizations and experiences in UE, our scene needs to be changed in order to meet new requirements or incorporate feedback from stakeholders.

In our 3D models, i tried to make as many as repeating elements as possible such as windows, doors, chairs, thus when Datasmith detects multiple copies of the same component in your SketchUp scene, it only creates one set of Static Mesh assets for that component, and places multiple instances of those Static Meshes into the scene. So that it is typically better for the runtime memory requirements and performance, as well as making it easier to manage the number of Static Mesh assets.

Units and Scale: In the Unreal Engine, all separations are constantly estimated in centimeters. However, other 3D design applications typically offer a choice of units of measurement. But Datasmith naturally deals with modifying the size of our scene so our geometry shows up at the very same genuine size in the Unreal Engine, and at the correct areas in 3D space. So we do not need to change anything from the previous work with Sketchup application.

Materials: The Datasmith import process creates a new Material Asset in the Unreal Engine Project to represent each different set of geometry surface properties it recognizes in the scene it imports. But we also need to tweak these Materials after import thus most of the materials assets are *Material instances*, it is referred to as the Parent Material, which can be used as a base to make a wide varitety of different looking Materials.

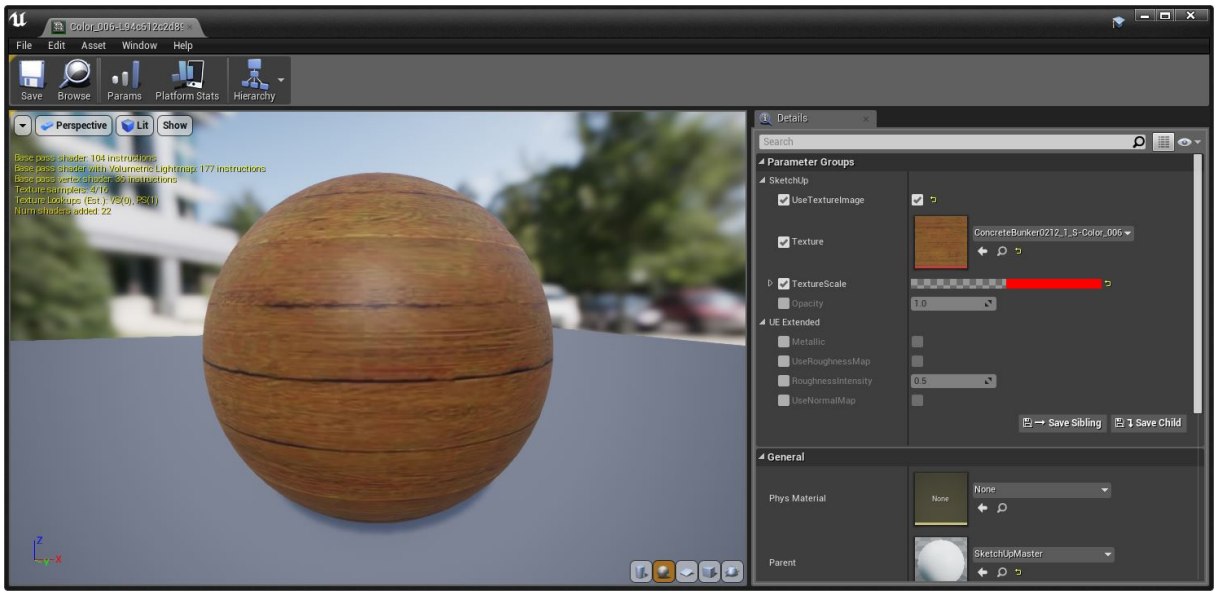

Figure 10: Unreal Engine Materials

The Datasmith Color Material: Sketchup use simple surface color to shade geometry, so we can easily use Datasmith exporter to brings these surfaces into Unreal as Instances of the *Datasmith-color Material*. The color of this Material is typically pre-set to exactly match a color from our Sketchup models, but as i mentioned before, we sometimes need to change/tweak some values. For example the brightness of the color maybe too bright, so we need to reduce in order for our lighting to look realistic.

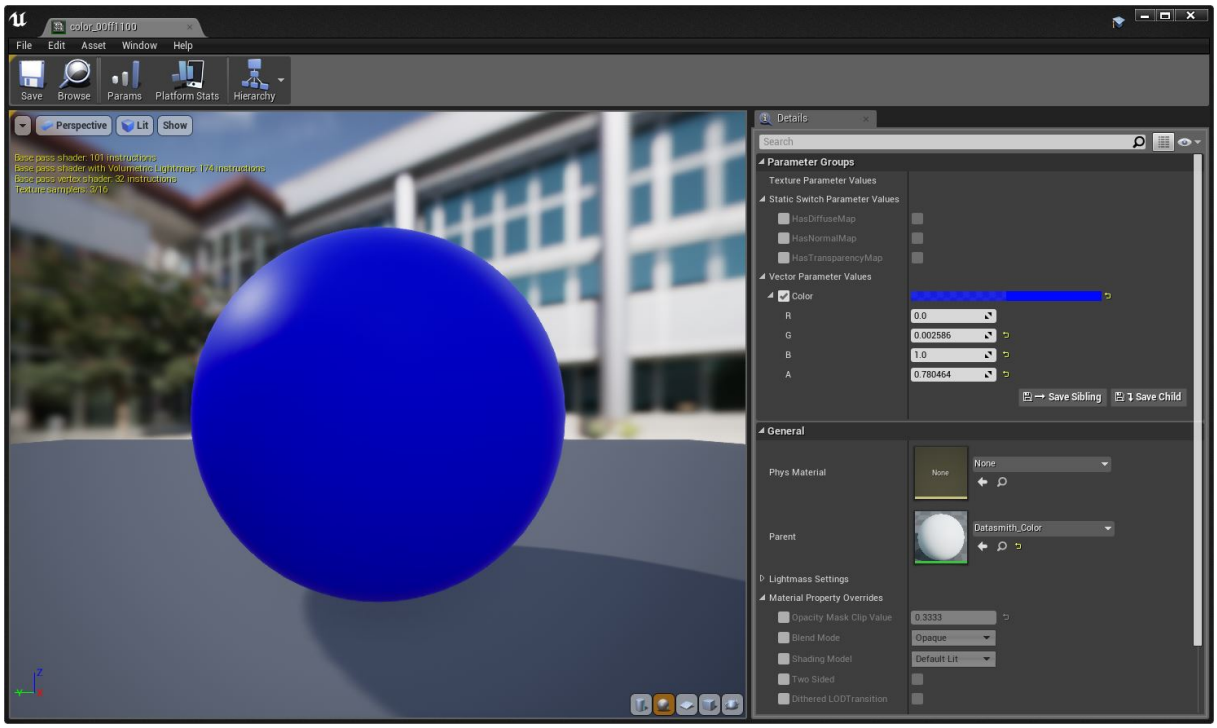

Figure 11: Unreal Engine Object Color

Layer: While creating 3D models with SketchUp, I use *Layer* tool a alot to organize the content, eg: a roof-floor will be a layer. Datasmith preserves that organization in the Unreal Editor so that we can easily show or hide layer to quickly find and select all objects in a layer.

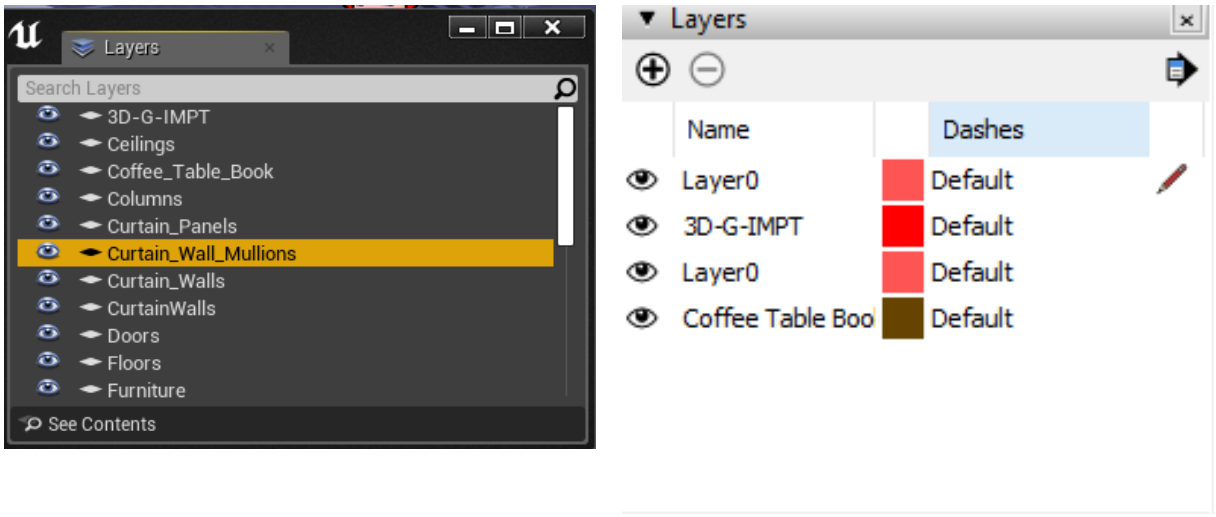

Figure 12: Left: Layer in UE4. Right: Layer in SketchUp

Player/Cameras: To be able to interact with the objects, a player should be added into the scene. Initially the player will not move without any script attached to it, and the most basic way to do this is through Camera Actors. The Camera Actor can be found in the Modes panel under the All Classes category. To place one into the world, drag it from the Modes panel into the game world. Using a Camera Actor is as easy as selecting it from the Modes panel and then dragging it into the world.

Thus my main purpose of this application is for students can be taken on virtual school trip before they apply to USTH, so that I only need to make my application as a first person view game.In order to do that I just attached a Camera actor into the player, once we have the Camera Actor in the world, I can then use it in combination with the Blueprints or Matinee to use it as a view point for my level.

Because when I created 3D models with SketchUp, I also saved a lot of camera views with *Scene*. So that when developing for this application in UE4, these views is also included at the time we converted these models, and now i can let user change their starting point with some specific hotkeys.

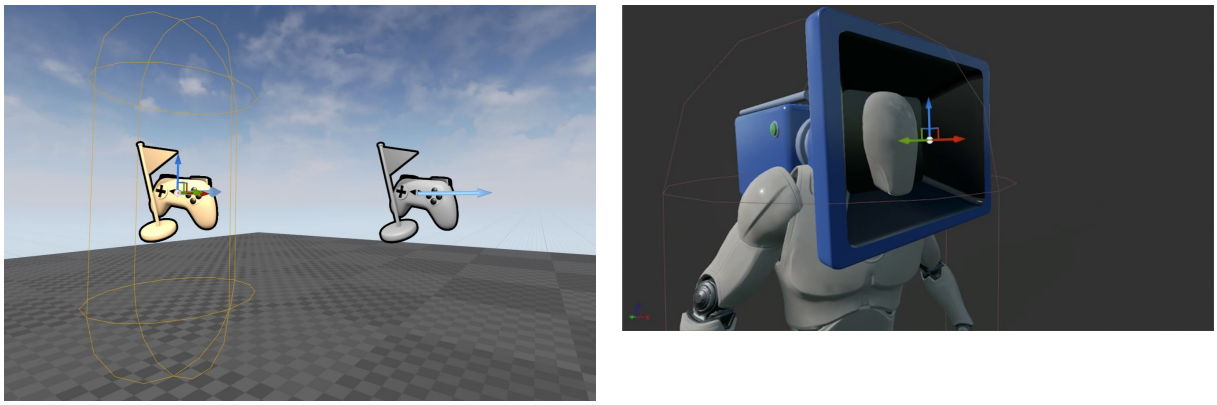

Figure 13: Player start and Cameras actors

Collision/RayCasting: Collision Response form the basis for how Unreal Engine 4 handles collision and ray casting during run time. Every object that can collide gets an Object Type and a series of responses that define how it interacts will all other object types. When a collision or overlap event occurs, both (or all) objects involved can be set to affect or be affected by blocking, overlapping, or ignoring each other. With the ability to get information from the object that the player affect, this is the main method for this project, as the character is able to interact with other objects in the virtual world.

Interact/Box trigger: Triggers are Actors that are used to cause an event to occur when they are interacted with by some other object in the level. We use this method to make some actions like door opening/closing or toggle on/off the light. There are 3 different type of triggers in Unreal Engine 4: Box Trigger, Capsule Trigger and Sphere Trigger. The method behinds these technique is quite the same, differing only in the shape of the area that player/object interact with as the name of each trigger types. When you are using trigger, you will notice that there are several different types of *events* that trigger can active. But the most common one is that using triggers with a collision in the previous section such as hitting, overlapping, etc…

In this project, i use *Box trigger* and *Overlap* most of the time thus it is the default and

common one. For an overlap to occur, both Actors need to enable Generate Overlap Events. For instance, to make the door open or close, I will create a collision component for the door then in the blueprint get a reference to the doors collision and connect it to a “set simulate physics” node with the activation box checked. I use the same method with toggle the light when player is entering the room.

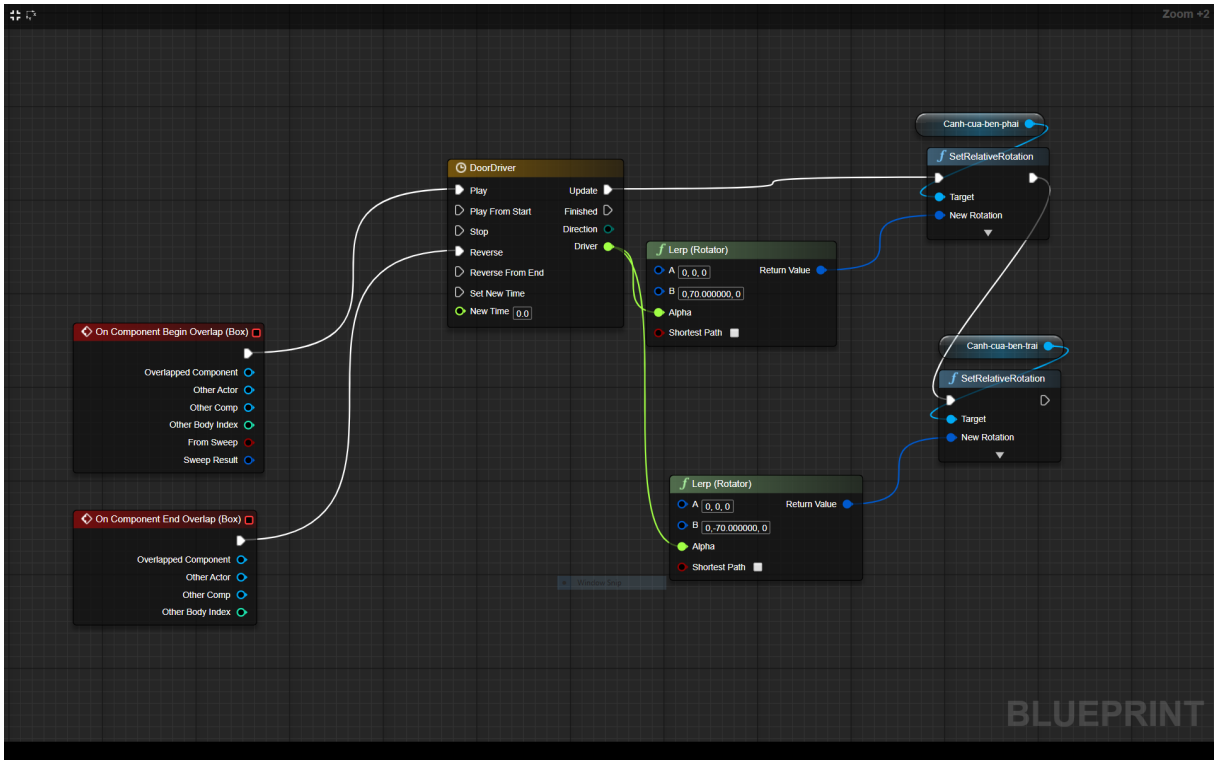

Figure 14: Door Open/Close

Figure 15: Box Trigger Example

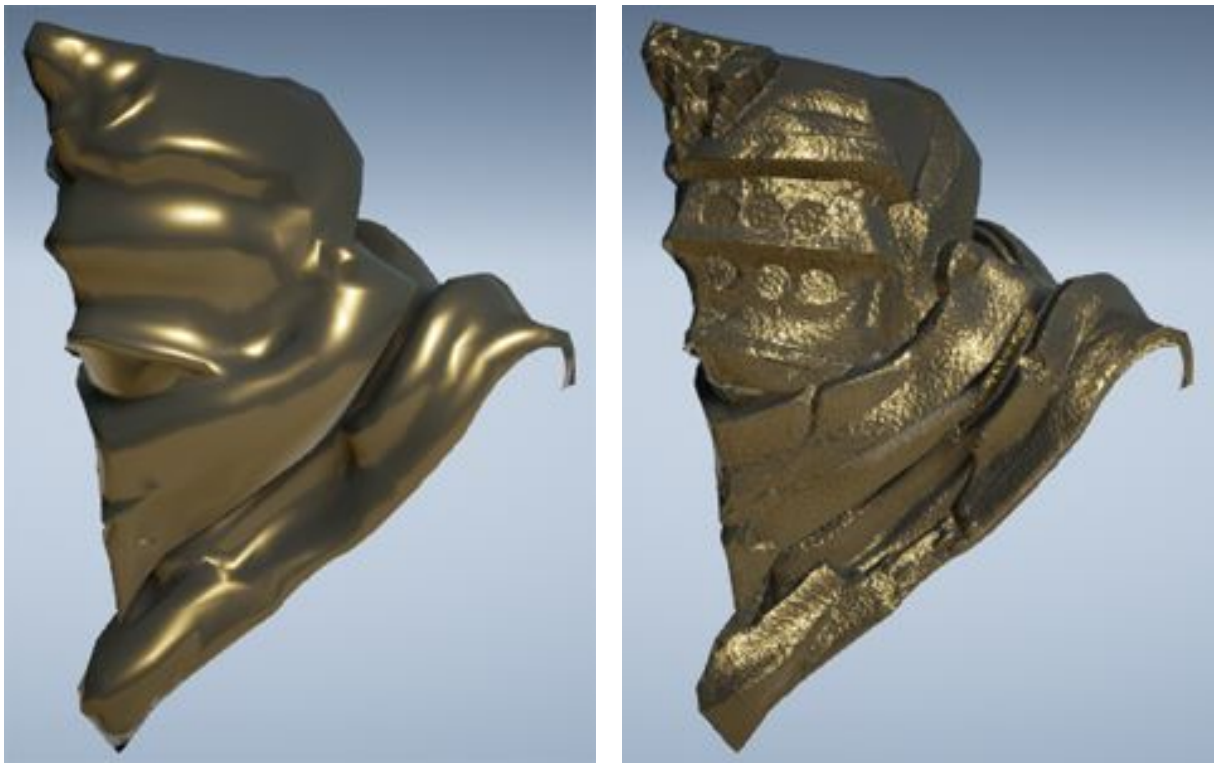

Figure 16: With/Without Normal map[2]

Bump Mapping/Bump offset: In order to make the object surface look more realistic, bump mapping/bump offset is one of technique in computer graphic to do that simulating small displacements of the surface, create the illusions of depth and texture of a 3D model using computer graphics. However, the number of polygons does not increase. The modified surface heavily relied on light reflection, define how the light should shine on the surface. Comparing to Displacement - one of another method for find geometric surface detail, it is much faster than displacement, no extra memory necessary so as i said before, the aim of this project is also focusing on performance but the quality is much lower. In this project, I only use *Normal map* as a type of *Bump map*. They are a special kind of texture stores a color in each pixel of the texture that allow you to add surface detail such as bumps, grooves, and scratches to a model which catch the light as if they are represented by real geometry.

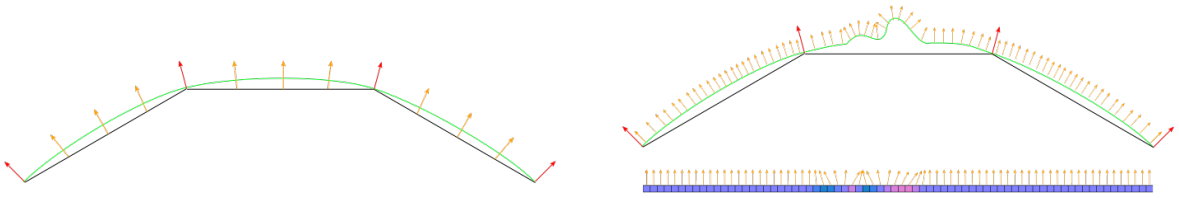

Figure 17: Left: Smooth shading on three polygons. Right:Normal mapping across three polygons / viewed as a 2D diagram[3]

The picture above describes how normal map affect the surface, the surface normals are represented by the orange arrows - used to calculate how light reflects off the surface. It means that these light will respond the same across the length of each polygon. So what happended if we add normal map, it will modify the surface with the texture that store the information. Thus, the normal map is an image texture, does not contain any actual light or dark shading which is mapped to the surface of the model, so it will represent a deviation in surface normal direction away from the true surface normal of the flat polygon.

With the normal map, the surface will look like in the picture below, as how the light shines on the surface.

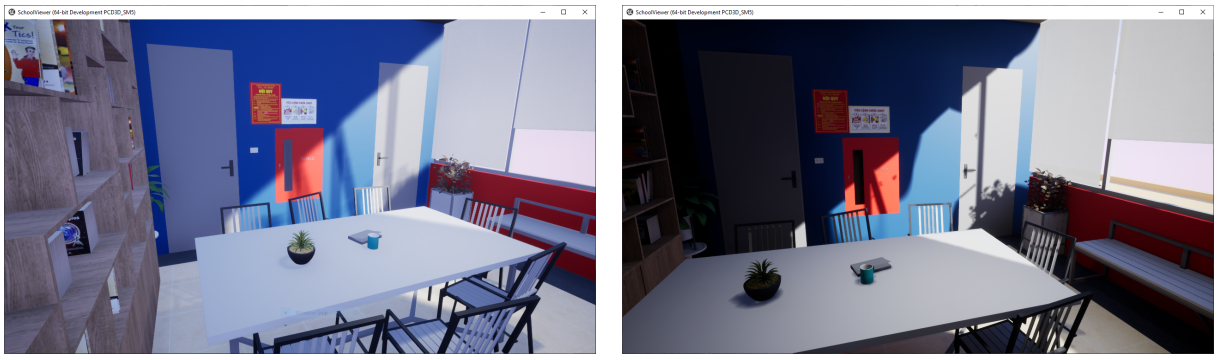

Figure 18: With/Without Normal map

Level Streaming: The Level Streaming feature makes it possible to load and unload map files into memory as well as toggle their visibility during play, making it easy to have large levels broken into pieces. This makes it possible to have worlds broken up into smaller chunks so that only the relevant parts of the world are taking up resources and being rendered at any point. If done properly, this allows for the creation of very large, seamless games that can make the player feel as if they are playing within a world that dwarfs them in size. You have just a small, whats referred to as a persistent level, to start and load other levels into and out of your persistent level as needed. So you set up a trigger so when the player walks through another part of the level is loaded. For example, I use this method to hidden/visible furnitures. Is it a process that can be seen so you must take care to make sure the player can't see that level until its completely loaded.

Lighting: Lighting is an essential part of every scene. In Unreal Engine, while the mesh, textures define the shape, the look of an object, the lights will define the color, the mood of a 3D environment. Lights can be added into the scene just by going to **Modes menu** -> **Light menu**. Different types of Light Components will allow you to add a light source as a sub-object to an Actor, depending upon the effect you aim to achieve. Once the light has been added into the scene, we can adjust the properties of the light to the way we want in the inspector, there are many different options within the Light Component. For example, we would like to adjust the position and rotation of the light using the position and rotation widgets like any other object.

There are some attributes that i will use as described in the table below.

Table 1: Lighting Attribute

Species	Attribute	Description
Intensity	Brightness	Intensity determines how much energy the light outputs into the scene. eg: 1700 lumens = 100W lightbulb.

Species	Attribute	Description
Light Color	R/G/B	Light Color will adjust the color of the light and the sprite that represents the light in the editor will change its color to match.
Attenuatic Radius	Source radius Source lenght	Define the size of specular highlights on surfaces.

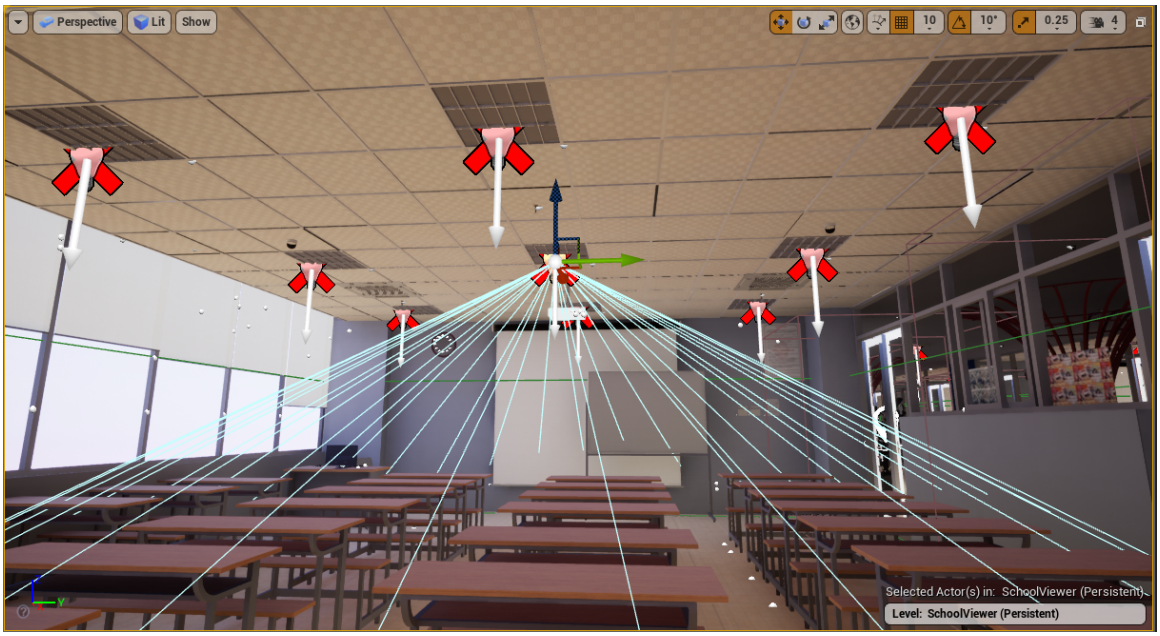

Figure 19: Lighting Example

IV Result and Discussion

Dataset of 2D images:

- **726 images**, **18 videos** with different views of USTH which is:
  - High resolution
  - Easily captured.
  - Cheap.
  - Popular.

Raw 3D models of USTH

- **6 3D models** - equivalent to 6 floors in USTH which is: 1st floor, 2nd floor, 4th floor, 5th floor, 6th floor and 7th floor.
- More than 300 different objects and textures.
- Foundations for other reaseacher or developer to create others virtual world of USTH.
- Can be exported to many datatype to use in any tools.
- Free to download all of my models in here.

Fully interactive virtual world of USTH

Succesfully implement 2 floors to create a virtual world with Unreal Engine.

- Lighting
- Shading
- 30 blueprints.

V. Conclusion and Future work

In this project, we managed to build a fully 3D CAD models for USTH build and a 3D virtual world using these CAD files. The simulation can be presented on USTH 10 years

celebration day. Yet, there is still a lot of room for improvement for this project. Here are some of the things that we want to do in order to make the simulation more realistic:

- **Beter Material**: to improve quality of the virtual world.
- **Increase Frames – per – second**: in this project, we implemented *level streaming* to Asynchronously loading and unloading levels during play to decrease memory usage and create seamless worlds. But, to conduct a full USTH building, we also need to find some others techniques instead of *level streaming* thus when i tried to import more models and increase texures quality, the computer started crashing due to memory overflow.
- Add more interactions.
- Ray Tracing

Appendix

Some demo pictures with different view in USTH 7th floor.

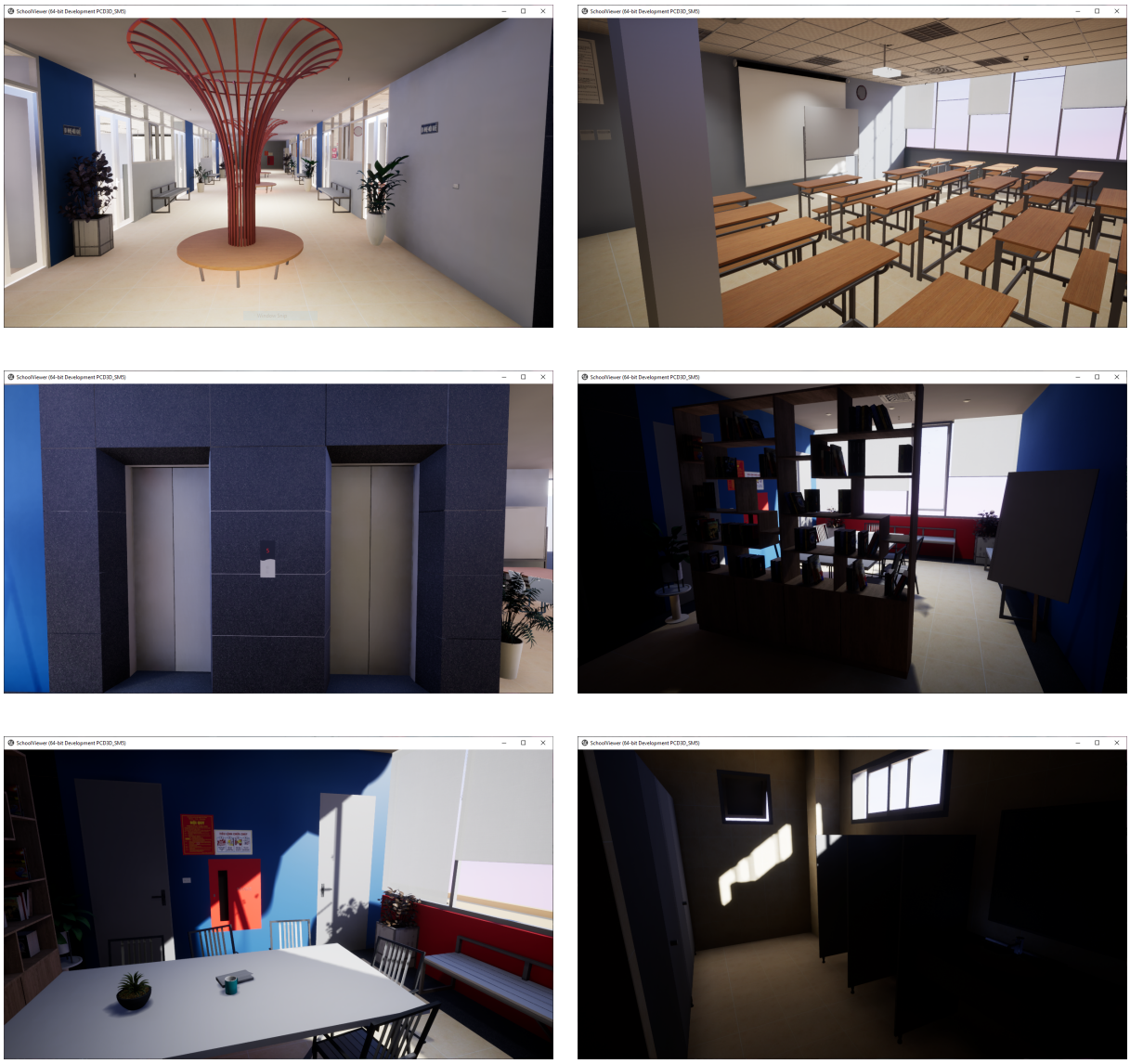

Figure 20: USTH 7th floor virtual world

Pictures about 3D models USTH laboratory in SketchUp:

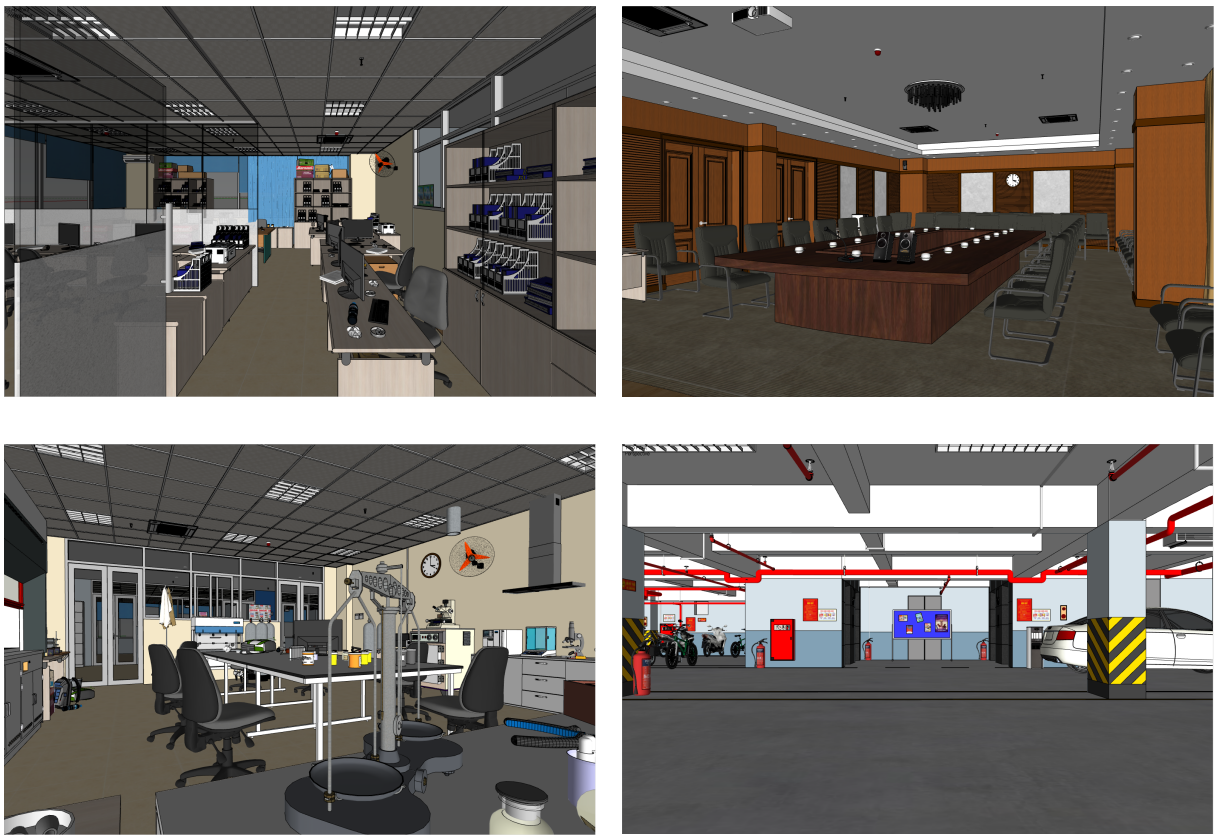

Figure 21: SketchUp models

Comparasion:

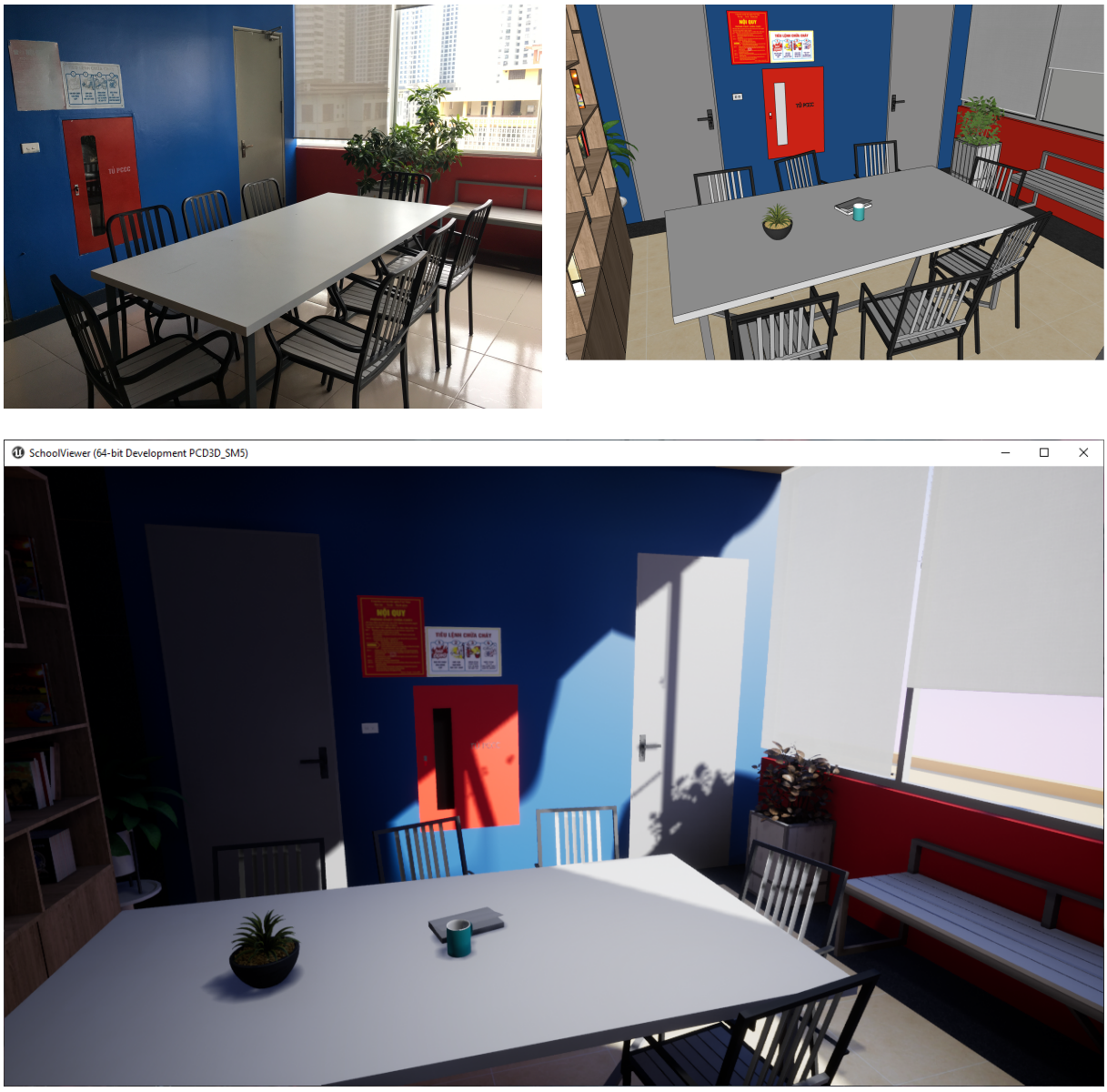

Figure 22: From 2D Image to Virtual World

References

[1] Unreal Engine Manual https://docs.unrealengine.com

[2] SketchUp Manual https://help.sketchup.com

// more references 6v5t9c