Giang Le Hoang Truong

3D MODELING ASSETS AND PROPS WITH MAYA

General 3D Modeling Pipeline

Bachelor's thesis

Information Technology

Bachelor of Engineering

2022

South-Eastern Finland
University of Applied Sciences

South-Eastern Finland University of Applied Sciences

Author (authors)	Degree title	Time
Giang Le Hoang Truong	Bachelor of Engineering	May 2022
Thesis title: 3D Modeling Assets and Props with Maya		59 pages
Commissioned by: Henri Brandt of Dark Amber Soft work		
Supervisor: Reijo Vuohelainen		

Abstract

The objective of this thesis was to research and implement a modern, general 3D modeling pipeline from start to finish. This 3D modeling pipeline is covered by 3 phases: modeling, texturing, and final rendering.

Modeling covers the basic pre-production steps, what is and how to implement blocking, followed by subdivision model techniques. Conclude with model grouping, UV unwrapping, packing, and model exporting. Modeling will be done inside Autodesk Maya 2022.

Texturing focuses on how physics-based rendering works. How texture maps (height map, alpha map, grunge map, and other 3D-assist maps) help with the new 3D texturing process. This phase is done with the aid of Adobe Substance Painter.

Final rendering explores the industry-trending Unreal Engine. How to import a complete model into a scene, set up the light, and necessary conditions. Then in conclusion, several final render shots that could be used for portfolio or first impression/presentation are shown. Epic Unreal Engine 4, Maya Arnold and Substance Painter Iray were utilized in the final rendering.

The handgun model was successfully recreated from reference into a 3D digital process. The robot police model has proper PBR texture and surface development. Autodesk Maya as a tool has proven itself as a competent and powerful 3D application for modeling, rendering, animation, and more. Adobe Substance Painter is also capable and quick to set up and process a 3D model with its procedural effects and PBR-ready materials. Epic game Unreal Engine 4 is a powerful digital engine to house and host projects of any kind, from digital render showcase reels to high-quality product entertainment.

Keywords
3D modeling, Texturing, Physics-based rendering, 3D Normal map baking.

CONTENTS

1 INTRODUCTION

According to a record blog of Y.Boi (2021), modelling started with the rationalization of a 2D mathematical polygon. Then from 2D, mathematic matrixes were invented and were one of many key foundations in modern-day 3D application. Mathematics matrix helps computers calculate how light should bounce off surfaces. In the 1950s, computers incorporated matrix and polygonal mathematics. However, they only prioritized a scientific audience and were not commercially available. In the 1960s, the first Computer-Aided Design system or CAD was developed to create and speed up the visualization. Though target customers for CAD were only construction and visualization. With the invention of CAD and its visualization feature, 3D modelling was being expanded and grew more complex. Figure 1 is an example of 3D modelling application. 3D printing applications can have their templates created by the 3D modelling process.

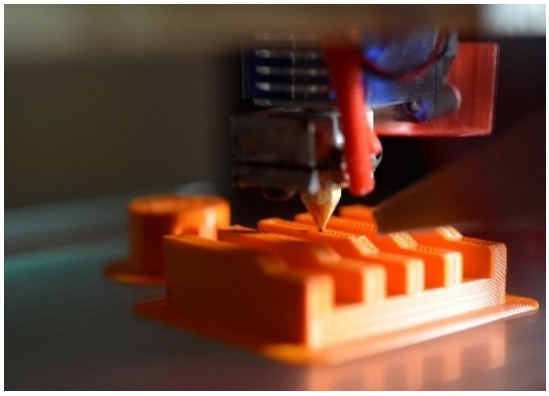

Figure 1: 3D printing from 3D digital illustration

At University of Utah, two students Gouraud and Phong discovered shading techniques, which improved computer calculation speed by simplifying the original algorithm. Phong material in some 3D was named after the student at Utah for his contribution. The Utah teapot was a digital object used by Martin Newell to test his graphical research. The research gave insights into surface reflection and object shadow development. The Utah teapot, in Figure 2, also got immortalized today by many shading developers as the first item to have proper shading.

Figure 2: Utah Teapot, the first shaded 3D object

At its core, 3D modelling is a process of turning fictional or real-life objects into digital rendition versions of said objects. Then digital objects can be used in various projects, such as mass visualization of buildings and cities, commercials and promotions video, video games and motion pictures, and many more. 3D modelling also creates templates of objects for technical visualization and explanation, 3D printing, sculpting, and fashion designing, as seen in Figure 2.

The primary aim of this thesis is to explore and go through the entire process of modelling a real-life object. Then give that digital object a texture or a coat of paint. Finally, render and take photoshoots of the digital textured object using a digitally created environment. The highlight of this thesis is the exploration and explanation of how the modern-day 3D processes called “Physical-base rendering” (abbreviated as PBR) and “Normal map baking” (also known as 3D baking) work. Another sub-process and theory about "Subdivision" that comes along with normal map baking will also be explored.

This thesis report is divided into three chapters. Introduction, theory, and implementation. Each chapter contains several subchapters, which dive into specific aspects. They are:

- The introduction chapter includes the thesis introduction, as well as history and today's application of 3D modelling.
- The theory chapter focuses on what physics-based rendering is, the basic elements of a digital model. Explanation about normals and texture maps, 3D planes, basic rendering, and basic camera settings in rendering. This chapter also dives into the fundamental understanding of a general 3D pipeline and introduces tools that are being utilized for this thesis.
- The Implementation chapter is going through some images taken during modelling a digital object and explains what is happening.

However, only selected steps are present due to the massive amount of work involved to make the handgun model.

2 THEORY

In this chapter, the basics of modelling, different 3D techniques, and the software will be explored.

2.1 Physics-based rendering and polygon theory

2.1.1 Physics based rendering theory

In his research, McDermott,W (2018) and a report done by Gamers Nexus YouTube channel (2015) with Crytek developers, stated that physics-based rendering (PBR) is a shading and rendering system, where real-world equations regarding how light works are used to calculate and simulate how different lighting conditions have effects on different materials in a digital environment. Before PBR was recognized, a 3D object behaved differently from the texturing process to real-time rendering inside a render engine. Not only does more work have to be done to make digital objects more believable, but the margin of errors when working cross-application is very small. There were workarounds to this margin, yet they only piled up to the enormous amount of effort put into texturing. PBR helps simplified and streamline the texture workflows. Since texturing application usually contains a light testing environment, they can behave differently compared to a final render engine environment.

Both light environments utilize the same physics equation to calculate how materials should react, this allows a more consistent result. With PBR, two types of workflows can be applied to all materials. A conductive or metal and a non-conductive or non-metal workflow. With each workflow, there are various options and approaches to keep in mind when developing the appropriate materials. For an instant, a metallic material will reflect light depending on its roughness. Whereas a non-metallic material will generally reflect less light than metal surfaces and that material will feature an albedo map.

2.1.2 What are polygons? Differences between a low polygon mesh and a high polygon mesh

According to Wikipedia (2022a), lines or edges connect to form a closed circuit of lines, or in other words, a plane or a polygon in 3 dimension. Edges are made up of 2 endpoints in the 3D space. Polygons are one of the foundation elements of a 3D model. A series of polygons is called polygonal mesh. Polygons are usually denoted by how many sides they have. An n-gon is a polygon with n sides. For example, a triangle polygon, which has the fewest possible sides polygon, is called a tri-gon.

Depending on applications and projects, various n-gons are more favourable than others. In animation or organic modelling, the entire visible mesh must consist of only quadrilaterals (or quads), very few tris, and n-gons if none are used. Since quads deformed much better when animated and will not leave mesh surface artifacts like pinching. Hard surface modelling and technical model can accept tris because their final product will not have deformation. However, n-gons are rarely used, only in very specific cases. Some examples of 2D and 3D polygons and primitives are shown in Figure 3.

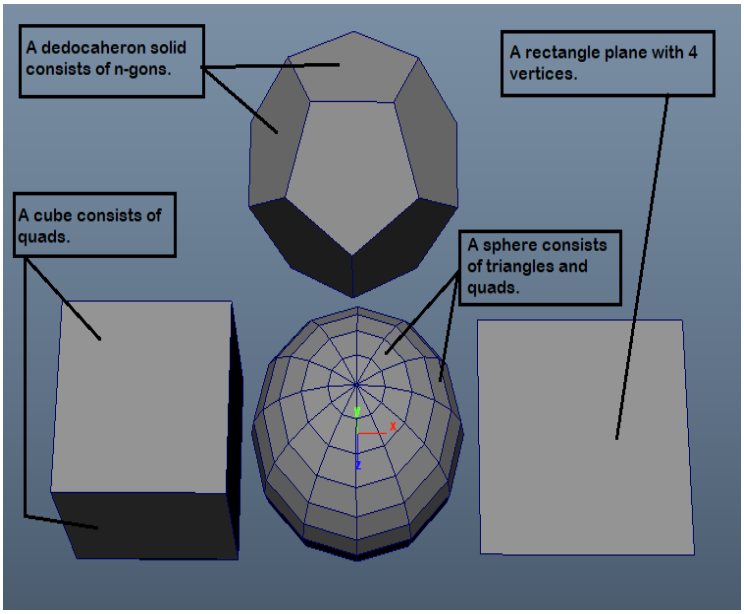

Figure 3: Polygon and 3D primitives

According to a published article by Wikipedia contributors (2022b), a low polygon mesh is a mesh that contains the fewest amount of polygons possible to form an object's silhouette from a viewing distance. Having a small number

of polygons helps manage the object's shape and optimize performance and memory when the mesh is being rendered in real-time.
The game industry and technical fields that need to render models in real-time will utilize low poly meshes to represent objects.

A high polygon mesh is a mesh that has additional polygons to increase its detail fidelity. Additional polygons are added in by various methods like subdivision and adding holding edge loops. High polygon meshes require intense graphic processing and memory allocation. Digital products like animated movies use high poly meshes. Contrary to video game models, motion picture models are pre-rendered. Both examples of quality polygon can be observed Figure 4.

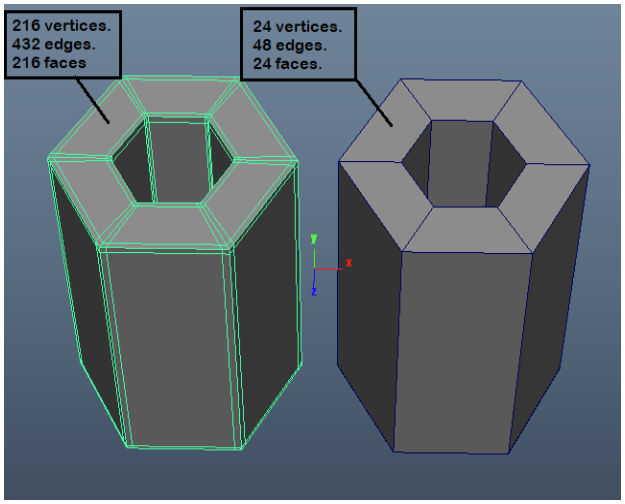

Figure 4: High poly object and low poly object

2.1.3 What are normals in 3D modelling

In a research published by C.Lemos(2020), a normal is a vector that is perpendicular to a plane, surface, or polygon, pointing away from that surface. The line does not need to be in the centre mass of a plane or surface, but it needs to be perpendicular. Normals are used to tell a render engine where the surfaces containing those normals are facing. Since normals are vectors, surfaces can only point in one direction. The render engine will take the normals information and calculate how light reacts and bounce off that surface.

Figure 5 illustrates how average normal calculation is done. By default, light rays will bounce off perpendicular to a surface, similar to real physics. However, if there is a gap between normals, or if two normals from the same vertex have an angle value above 0, then the light will bounce off from either direction of the normals. Thus a viewer is observing 2 different surfaces angled to and connected by a common border edge. In most 3D modeling applications, there are options to average out this gap between the surface to create an illusion of a smooth transition from one surface to the other. This process works by gradually aligning a normal to the closest polygon to it. Since light rays also follow the normals transition, essentially there will be no border edge between two angled after smoothing.

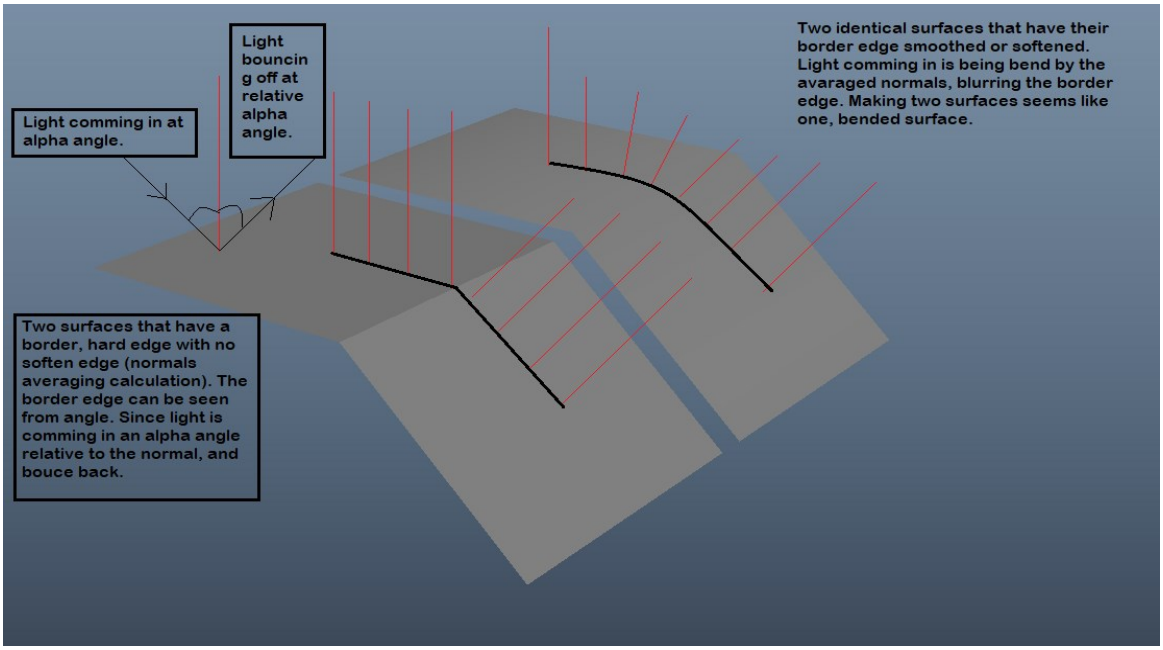

Figure 5: Normals in 3D modelling

2.1.4 Texture maps

According to a publication of Wikipedia contributors (2022c) Texture maps are a collection of 2D images that represent texture for a 3D model. A set of texture maps are created by taking a model and undergoing intense computing and calculation. The result is a set of detailed maps like heightmap, bump maps, normal maps, albedo maps, and many more.., that make up the desired characteristic of a target model. A 3D object when undergoing UV unwrapping will yield 2D UV shells, which can be turned into a 2D texture map. The modern process of computing and calculating texture maps is called "baking". The ideal of baking is illustrated in Figure 6.

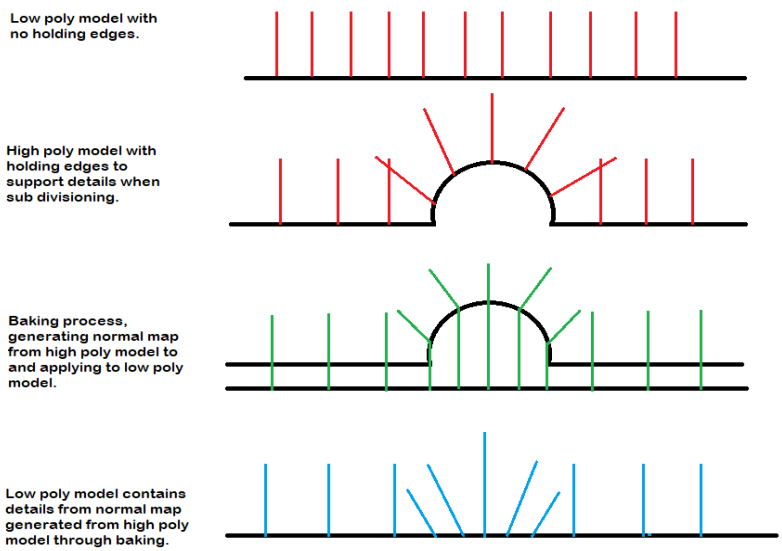

Figure 6: Baking process

Some of the texture maps are:

Bump maps help simulate surface roughness and wrinkle without additional work on the 3D model's surface.

A height map helps define elevation information on a surface. Usually visualized by a greyscale, or rarely, a colour ramp. Heightmap also helps in a quick terrain generator process.

Ambient occlusion map is a type of lighting map explaining how certain parts of an object react to an ambient light source, hence the name "ambient occlusion" in 3D graphic applications. Generally, the farther away those parts are from the light source, their ambient maps will be darker. An ambient occlusion map helps give depth perception to a scenery.

Normal maps are one of the defining features of modern high-quality modelling. A normal map uses computing techniques to simulate lighting, denting, and other details by modifying and manipulating normals. Then the generated normal map can be applied to a low polygon model. Giving it the illusion of a high fidelity surface, with a poly count of a low poly model, saving memory space for other applications. A normal map needs a high poly model to be correctly generated, which is created by adding holding edges and subdividing a separate instance of the final production low poly model.

2.1.5 Subdivision theory

Subdivision is a process of adding additional polygon to a mesh to make it appear smoother while maintaining the original shape. Subdivision generally takes place after a low polygon mesh has been created. Either the original low polygon mesh or a copy of it can undergo subdividing. In a 3D application such as Maya, when the task of subdividing a mesh, will fill in the space between edges with additional edge loops. This, in turn, add more detail and polygon to the mesh. The subdivision value can be adjusted as a multiplier. The default value of subdivision is 2, and increasing the value means having more polygons, this also increases the load of the graphic processing units and the final file size greatly. Figure 7 showcases the effect of subdivisioning on both high and low poly model.

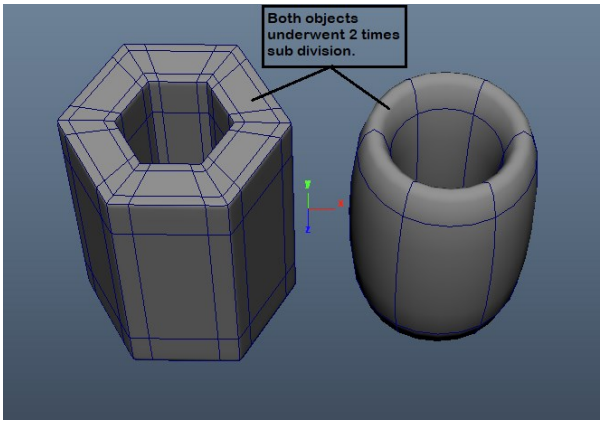

Figure 7: Subdivision between high and low poly

2.1.6 Basic 3D axes and planes

There are 6 orthographic views angles, they are Front, right, top, back, left, and bottom. Front in default world space is viewing from the positive Z-axis to the XY plane. Right is viewing from the positive X-axis to the YZ plane. The top is viewing from the Y-axis to the XZ plane. And back, left and the bottom views from a negative axis of the front, right, and top perspectival.

2.1.7 Rendering basic

In research of Wikipedia contributor (2022d), rendering is taking a 3D scene and converting it into 2D images with the help of a computer's CPU or GPU. Rendering can be either photorealistic or non-photorealistic stylized. There are

two types of rendering. Real-time rendering, which pulls objects and their textures from memory, put them together and into a scene at a very fast pace. The other type is non-real-time rendering, which subjects models to intense rendering time. This is done typically with a large amount of GPU power, spanning over a long time. The former is best suited for video games, where objects need to constantly appear and disappear in a scene. Objects in real-time rendering need to exist within a poly count budget to save render time and memory. While the latter is almost exclusively used for animation and video demo reels. Objects only render once at the beginning and the entire scene is recorded for future use. Therefore, a non-real-time rendering scene does not require a polygon budget.

2.1.8 Camera basic settings in rendering

Camera settings in renderers mimic their real-life counterparts. There are three key camera elements. Focal length, a field of view or angle of view, and aperture size. Figure 8 illustrates the relationship between focal length, field of view and magnification.

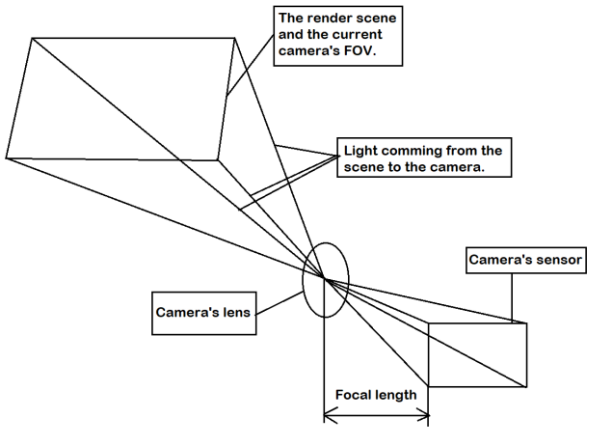

Figure 8: Camera setting visualized

Focal length is the distance between a camera lens and its sensor. It is measured in mm or inch. Focal length dictates when the lens is focused at infinity. Focal length correlates with both magnification and field of view. The greater the focal length is the more magnification and the less field of view.

Field of view (FOV) is the measure of how much of a scene a camera sensor can capture. FOV is calculated in degree. To increase FOV, both magnification and focal length have to decrease in their value.

Aperture is the measure of how much light is reaching the camera's sensor. With aperture setting, images will have depth to them. 3D render to create the depth of field effect by adjusting the value of aperture in the camera setting.

2.2 Pipeline/Workflow Overview

Below are the basic steps of most pipelines. Advancing through the next step does not mean that the previous step is over. Instead, the previous step could be revisited for troubleshooting and adjustments. Figure 9 is an example of a rendered model with texture.

Figure 9: A 2D render

2.2.1 Gathering 2D references

Gathering references is the process of building a collection or library of mostly images and information regarding the upcoming objects that are going to be modelled. This could be done by taking pictures of actual objects and taking notes about their appearances and characteristic. If the objects in question do not exist, the 3D artists are going to discuss and consult with illustrator members, who are handling conceptualizing, technical explaining, and reference building. This initial reference collection is key to a proper final model. The more references and information presented, the easier it is to block and model an object.

2.2.2 Setting up references and project folder

Images from the initial reference library can now be used inside a 3D application for blocking. Most modern-day 3D applications support importing references directly into the scene. Images that are used to set up object references are usually taken or drawn from orthographic view angles. Image planes are one of many tools used to contain object references. These planes have control over images selectable or non-selectable, grouping images and showing or hiding images.

3D applications also come with specific project creation tools to streamline and make initiating a project much easier and faster. These tools hold folder templates to sort all relative files and assets (models, images, movies, soundtracks) into different folders for management, tracking, and documenting.

2.2.3 Blocking

Blocking is a colloquial term and is used to refer to the process of combining and manipulating Primitives and basic components, such as curves, to form the most basic shape of a digital object. This shape can be used to discuss further art direction and aesthetics. Once satisfied, the shape can be expanded and developed further. In Figure 10, an object slowly reassemble it reference, with very little detail.

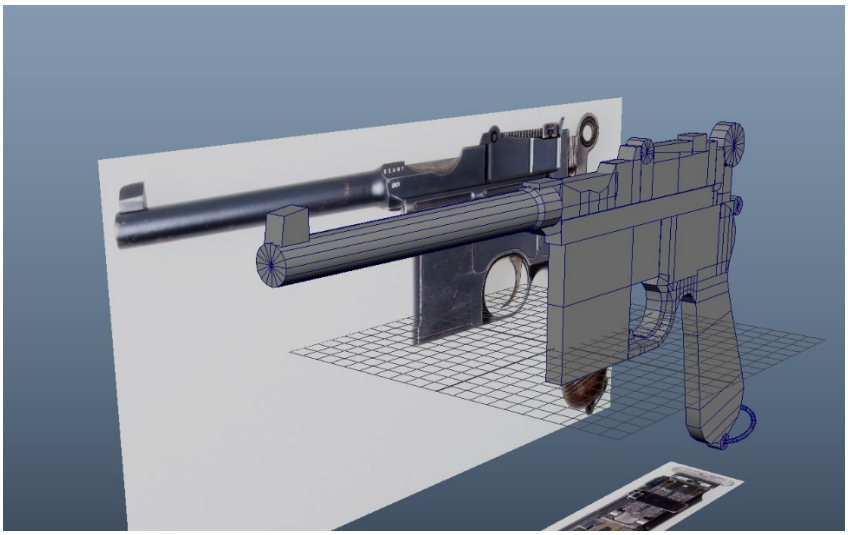

Figure 10: Blocking in an object

Blocking is done with both orthographic view reference images and non-orthographic view images. Orthographic view images essentially outline the silhouette of an object, making it easier for blocking. However, in an information-deprived project, where very few or only non-orthographic view references are present, the basic shape of an object can be deducted, broken down, and blocked properly.

2.2.4 Adding holding edges, de-resolution

Adding holding edges is a process that turns a low polygon model into a high polygon model. Holding edges or reinforced edges are edges or edge loops that help the subdivision process. Without holding edges, a low polygon mesh will collapse and deform into itself. Holding edges will prevent collapsing because the gaps between regular and holding edges are smaller. Objects will sufficient holding edges is a high resolution or high poly count objects. These high poly count objects can then be used to bake their surface detail into their low poly count counterpart. Figure 11 displays the differences between a high and low poly model.

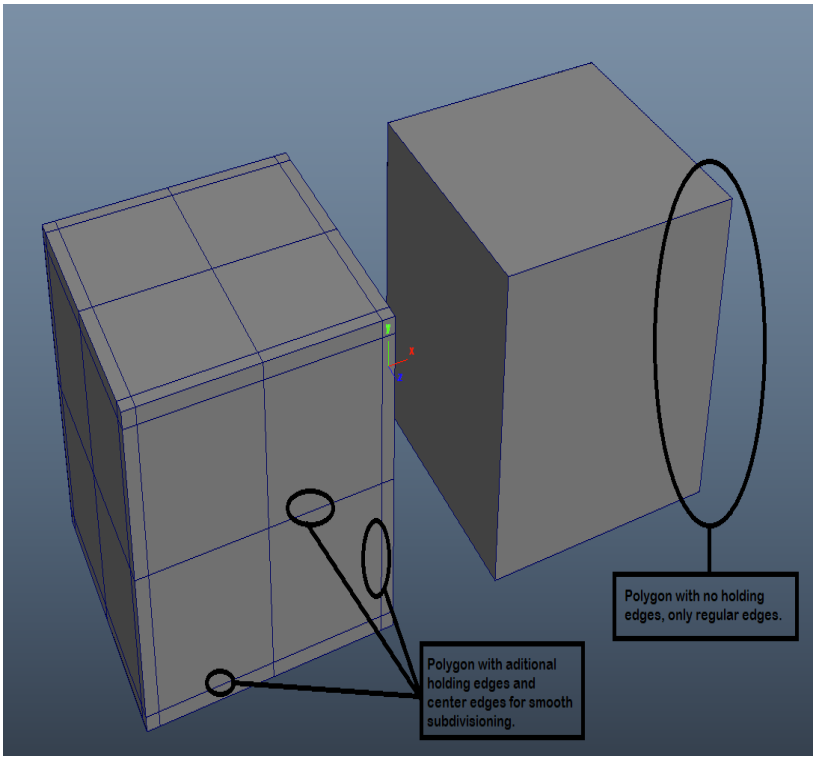

Figure 11: Holding edges and effects when subdivision

Figure 12 further explores how both models react under a 2x subdivision level. The cube on the right has collapsed and turn into a globe. Because there were no holding edges to reinforce its shape.

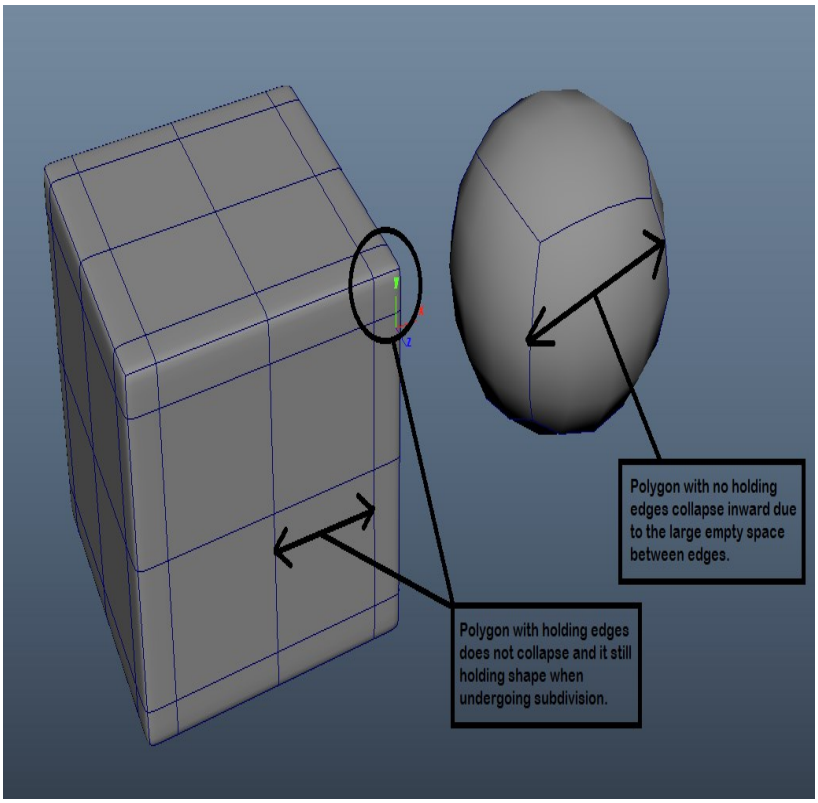

Figure 12: With and with no holding edges under subdivisioning

De-resolution is the opposite process. If objects are 3D scanned or 3D sculpted, their poly count is massively higher than those that were hand-modelled. Low poly count models of those objects are generated through remesh process, automatic retopologize model, or hand retopologize. Hand-modelled objects will usually be kept in three different versions. The original low poly as the backup. A copy of the original with reinforced edges up resolution and baking texture. And a copy with no holding edges for final iteration inside a game engine.

2.2.5 Naming, suffixes, grouping, mirroring and final review

Naming is giving proper names to all meshes within an object or a scene. This is pre-documented and discussed before modelling begins, to make sure that names are proper and consistent throughout the entire pipeline. There are multiple ways to name meshes and models, the most used methods are camel case and underscore. Suffix that helps with identification when baking

is also added when naming. It distinguishes the high poly model from the lower poly one.

Grouping is combining related meshes to sort parts into their proper family and clean up the mesh hierarchy. Grouping is also crucial for initial positioning and posing without animation. Since grouping reset the manipulation handle back to the world centre (x0, y0, and z0), a group of meshes can be positioned in a proper way for posing or later animation. Multi-layers groups are also a possibility.

Mirroring is the process of taking half of a model, duplicating, and shifting that half to the other side. Then both half will be welded together in their middle border edges. This technique is mostly relevant to symmetrical models. Modelling half an object will save time and there will be fewer errors to troubleshoot.

2.2.6 UV unwrapping and tiling

UV shells are in essence, the outer layer texture of an object. To unwrap UV is to cut these 3D shells and lay them out on a 2D plane, which is called UV. U is the horizontal axis while V is the vertical axis. UV unwrapping is done on the low poly count model. Tiling or laying UV shells is a legacy method of properly laying out certain shells for hand painting. Tiling can be done automatically in the present day. However, certain types of texture, painting methods, and requirements still demand proper hand tiling. There is no definitive method for all UV shells since all objects are distinguished from each other. There are only rules for certain shapes or objects. UV shells after tiling should be optimized and also maximized in the space within a tile, which is the plane that houses UV shells. As examined in Figure 13.

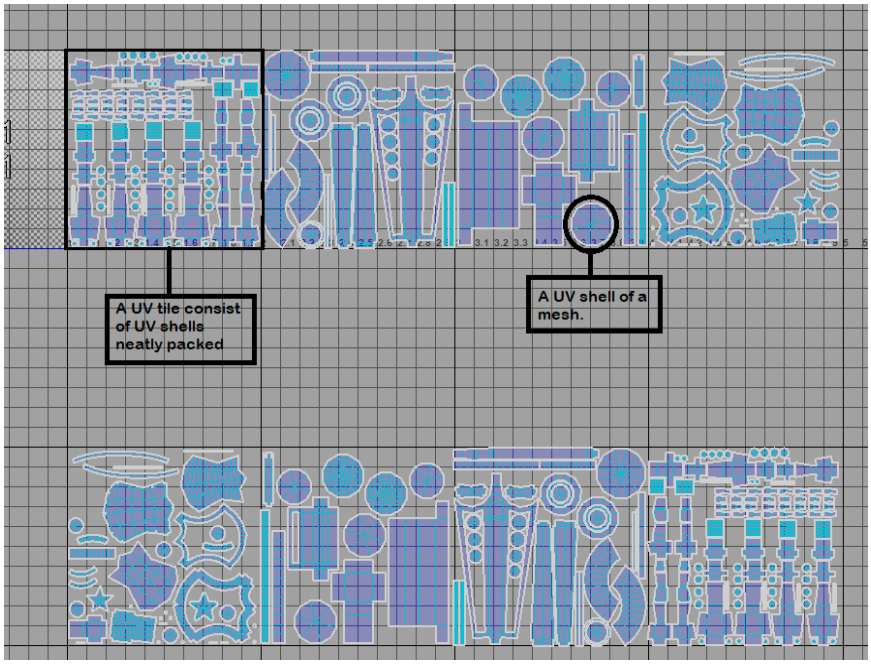

Figure 13: UV shells and tiles

2.2.7 Prepare model for texturing

In preparation for texturing, there will be two meshes. One is a low poly count and optimized mesh that is used for texturing, rendering, and final application. The other one is a high poly count mesh that contains holding edges, with no artifacts such as pinching, or stretching when undergoing subdivisions. Each model has been properly named, with all parts categorized into groups. All UV shells are unwrapped and tiled. Each model should also have its history memory cleaned, all translations are zeroed and its translation values froze, which is returning all values to "0". The high poly model will be processed and permanently lock in the subdivision, while its counterpart has all of its edges smoothed or softened.

With the high poly model, extra steps can be done to further add control when texturing. Faces and vertices can be painted with different colours. Since these colours' ID information is exported along with the model, texture applications can recognize this information, and put them into what is called "texture sets". A texture set function is similar to a mesh group. It contains all faces or vertices that have the same colour. Different effects, materials, and other textures can be applied to the texture set. Therefore save time when mass painting serval parts. Then each model will be exported out as either an OBJ file or an FBX file to import back into a texture application later.

2.2.8 Texturing and troubleshooting errors

Texturing is giving an object colour information and details information in a 3D environment. Texturing at its basic is drawing and giving information to a blank image or multiple images, which have been layered on top of a model's UV tiles. Legacy 2D texture method was a more direct approach, with having to paint and draw detail on 2D blank images. The 3D texturing application still retails this ability in addition, with new features such as separating meshes by the colour ID or name to quickly paint all. Painting directly on an object and developing its surface in real-time.

Texturing is also a continuously going back and forth process between 3D modelling applications and texturing applications to troubleshoot errors. Errors can be improper pack UV shells, UV shells misalignment, pinching and stretching on a high poly model, normal map artifacts, texturing bleeding into neighbour meshes, and more. After texturing is completed, all texture files can be exported out as image files (.PNG, .JPG and other image formats). These image files represent information correlated to texture channels. Such as a base colour file is for a model's base colour channel.

2.2.9 Environment setup, import, combine with texture and render

Before importing both the low poly model and its texture file, the rendering environment needs to be set up. This involves the placement of light sources, additional visual effects if related, and cameras. Light sources can be global illumination, direct and indirect sources, and other types. Light sources also contain adjustable parameters for fine-tuning. Visual effects can be brought into a render scene from other software or can be done locally with a 3D engine effect library. Camera placement and camera parameters are also vital for the final render. Since photo shots are taken from them. Modern-day render or engine allows for various options for camera positions, camera properties, which resolution the scene should be, and more.

After setting up light and cameras, models and their texture files can now be imported into the rendered scene. Most renderers and game engines contain tools and templates for storing and grouping texture files with their proper models. Objects, accompanied by texture files and effects make up a render

scene. Within a renderer, texture file nodes (normal map, base colour map and others) need to be connected with the model to form a complete textured object. The corresponding UV shells need to match the original position during texturing. Scene setup is also a back and forth process between the renderer and the texture application. With physics-based rendering, most colour and material information can seamlessly transit between texture and render.

2.3 Applications for Modelling, Texturing, and Rendering

2.3.1 3D computer graphic application

3D computer graphic applications or modelling applications contain tools to create and manipulate 3D shapes. 3D modelling applications can also create and handle different types of 3D shaders and materials, which support certain project types like technical modelling, film making and digital entertainment. There are many 3D computer graphics applications like Autodesk 3ds, Autodesk Maya, Blender, Mudbox, Modo, and many more. Some applications are paid-licensed while others are free. The final version of the primary model in this thesis will be made with Autodesk Maya, version 2022.

2.3.2 Texture application

Texture applications are applications that house painting, material, and effect tools. Legacy texture applications such as Adobe Photoshop used the 2D images or UV files unwrapped from a 3D model and paint directly onto them. Modern 3D texturing applications allow for direct painting and adding materials to a 3D model. Modern texture applications can also handle the creation of new materials, paintbrushes, alpha effects, and other texture tools for utilization in different projects. The chosen texture application for this thesis is Adobe Substance Painter, version 7.4.1. Alternative texture applications are photoshop, Marmoset tool bag, Graphite and Mari.

2.3.3 Rendering process, render application and game engine

Renderers are software that takes a 3D mesh file and its texture files, then combines them into a proper object. Then this object is placed accordingly with other objects within the current environment, which is called a scene.

With multiple lighting sources and settings, the entire scene can now be exposed to various cameras that also have been placed around the scene. Different angles of a scene can then be rendered out using either a computer CPU or a GPU. Game engines contain their renderer and also accept visual effects, soundtracks and music, and more into a scene. Dedicated render application are Marmoset toolbar, Keyshot, Redshift from Maxon, V-ray and Maya's Arnold. For this thesis, three renderers have been chosen. Iray, a native renderer inside Substance Painter. Arnold, Maya's proprietary 3D renderer and Unreal Engine 3D standard environment and camera works. These renderers will be compared with each other for setup difficulty, render time, and final fidelity.

3 IMPLEMENTATION

In this chapter, image references will illustrate the steps and how they are done. For the whole blocking phase, the target object will be the C96 handgun and its references. The robot police model will continue from naming, grouping, UV unwrapping, texturing, and final rendering. Since the robot model was prepared ahead of time, there are no records of the robot being blocking from its reference.

3.1 Modelling Phase

3.3.1 3D computer graphic application

The object that will be modeled is a Mauser model C96 handgun. There are three orthographic view reference images and additional angled view references. Going through the images, the model is dominantly blocky. Most parts can be done with basic primitives like cubes and cylinders. However, there are large pieces that can be broken down into small, simpler pieces. They can be modeled separately for easier edge flow management. And they can be reconnected together later to reform the original large pieces. Figure 14 analyzes the left side of the handgun.

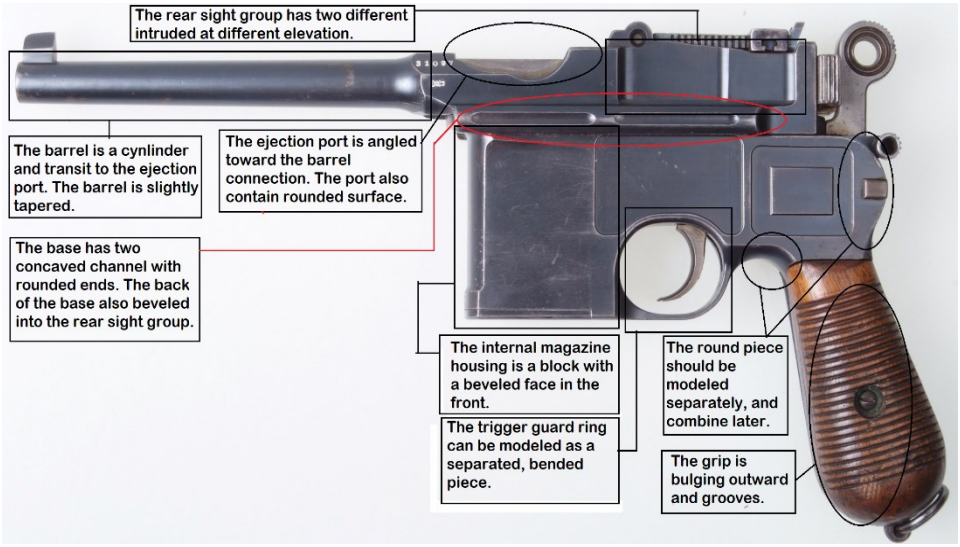

Figure 14: References left side breakdown and analyzed

While Figure 15 goes into the detail of the right side. However, both side share many similarities with only minor differences.

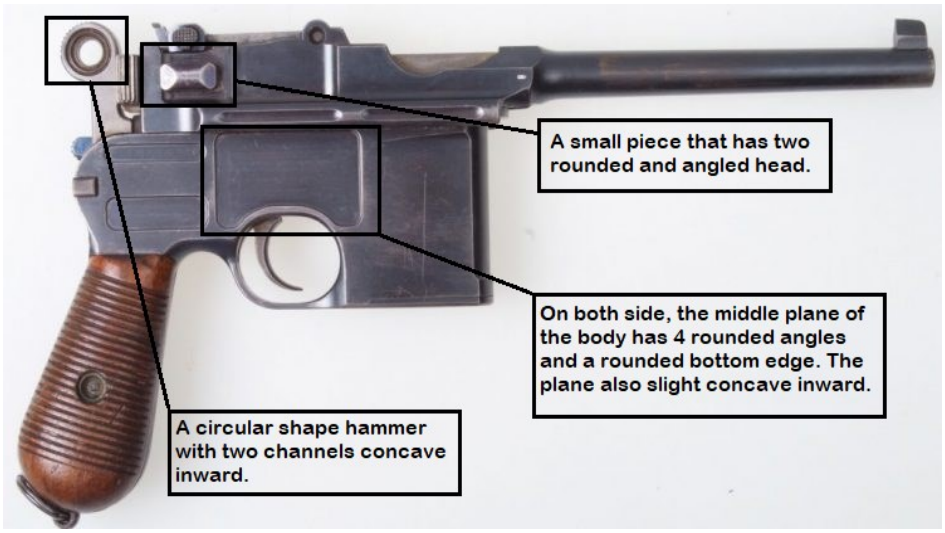

Figure 15: Reference right side breakdown and analyzed

3.3.2 Start a project and reference setup

In Maya, by default, the location of the root folder, which holds all other project-related subfolders, files, and assets, needs to be initiated. The root folder can be designated by the **Set Project**, the option can be found in **File** sub-menu. Figure 16 shows the location of both options.

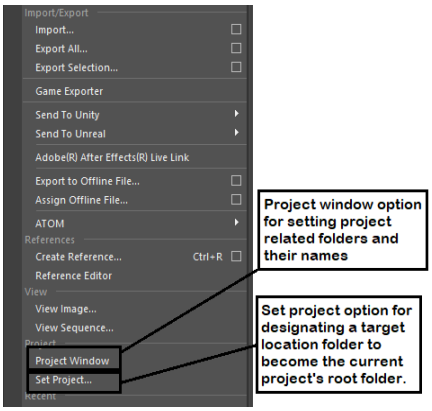

Figure 16 Set project

Both left and right-side references can be brought into the scene for initial blocking. A camera angle can be selected first, then an **Image Plane** can be created to contain reference images. The camera angle can be found in the general hot menu. The image plane option can be found in the current view port's **View** menu. In Figure 17, the location of the left orthographic view can be seen. Figure 18 illustrates where to create image planes.

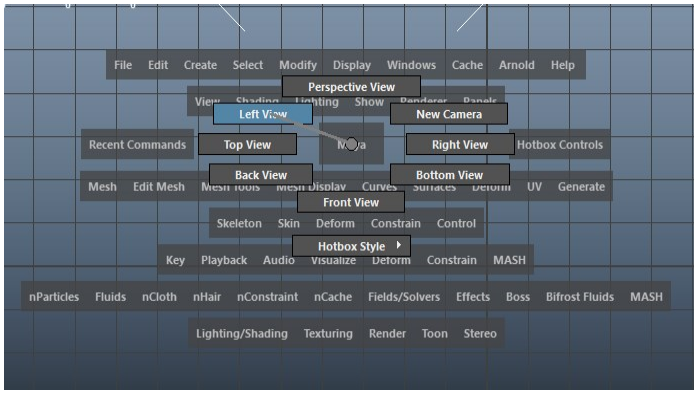

Figure 17: Orthographic view in the hot menu

The image plane creation tool can be found in the view menu on the left of the viewport. Then image references can be imported view the import tool of the newly created plane.

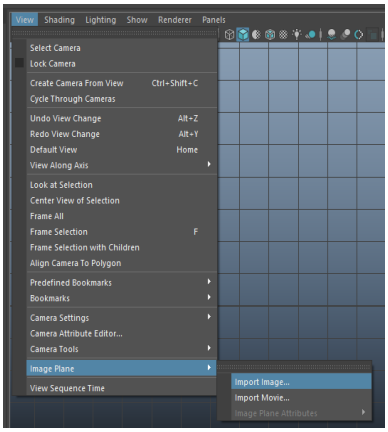

Figure 18 : Image plane creation in the View menu of the viewport

With both image planes now occupied with references, they need to face the correct, positive Z angle. This means that both reference gun barrels should be facing the Z direction. Image planes are treated as a regular object within a scene. All manipulating tools are going to affect them. Figure 19 shows a cube with it manipulator handle, and two references are pointed at the Z direction.

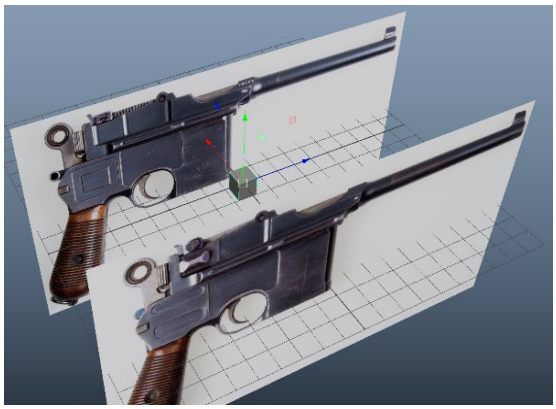

Figure 19: Image planes alignment to the Z direction

Both image planes can be grouped into a layer for more control during modeling. Managing layers can be created with selected objects in the bottom right of the viewport. Layers will allow for objects to be non-selectable. So the image planes will not interfere when modeling. Figure 20 shows the location for creating a managing layer.

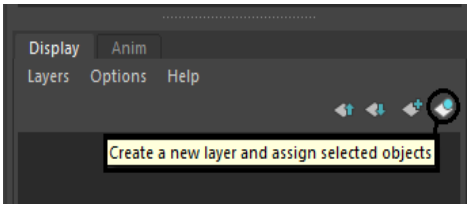

Figure 20: Managing layer creation option

In Figure 21, the current status of said managing layer can be toggled. A null status will make a layer react normals. While a reference mode makes it un-selectable.

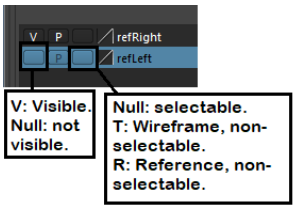

Figure 21: Managing layers status toggle

The size of references should match the original object's size. Or the overall length and height of references should follow the object's actual length and height. However, in an information-deprived situation, a sample human model can be imported into the scene, and objects that require correct technical data can be referenced from this model. The human model used for this project is approximately a 180cm tall, human male model. Figure 22 shows that imported images may be smaller than a regular human model.

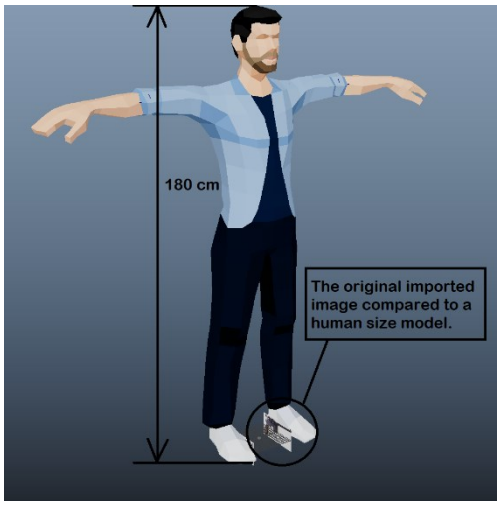

Figure 22: Human model overall size comparison

The reference is properly scaled to mimic the its real-life size in Figure 23. The handgun does exist in real life. Matching the grip of a human model is one of many ways to gauge an object's size.

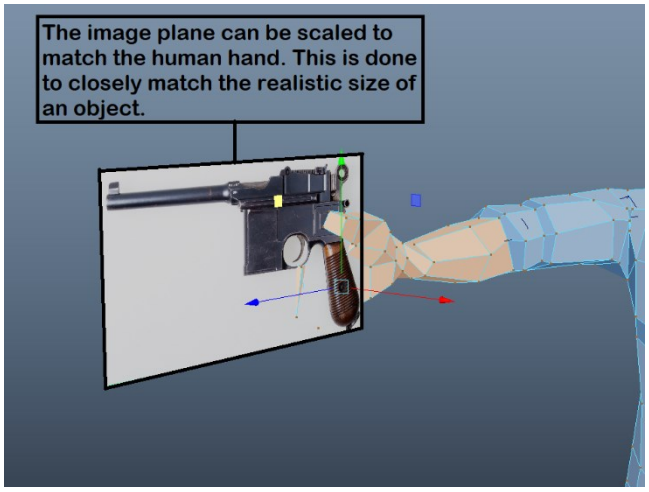

Figure 23: Human model hand size comparison

3.3.3 Blocking from references

When blocking an object from reference, it is crucial not to put too much effort into details and edge flow yet. Since blocking is to recreate the silhouette of an object, hence the term “block”, means that the model in this stage only consists of blocky primitives like cubes and cylinders. Figure 24 shows different blocked parts and how they were created.

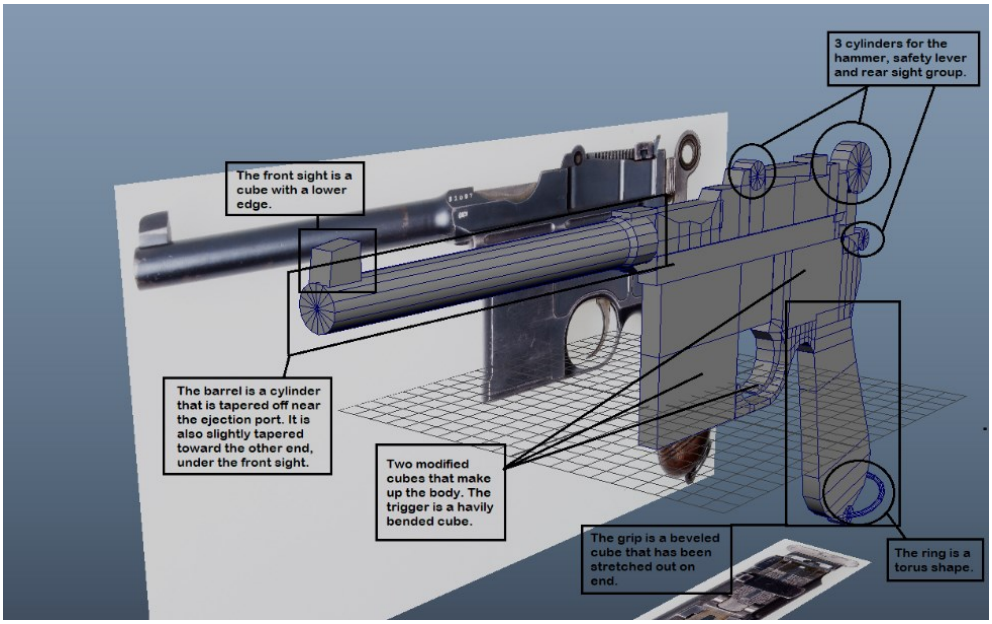

Figure 24: Blocking in an object

In Figure 25, the block-in cross compared to the left side reference. The standard material has it transparency toned down for easier comparison.

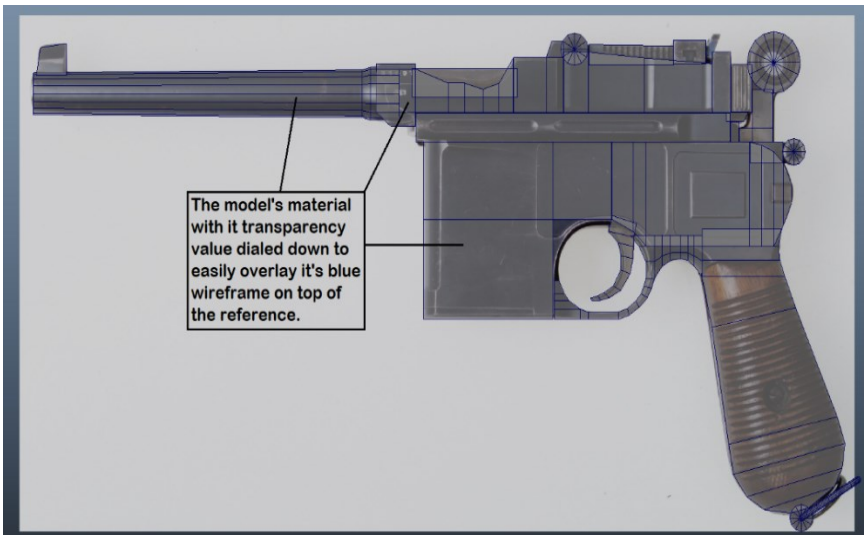

Figure 25: Blocking in cross reference

3.3.4 Adding details and adjustment to base model

Common 3D modeling practices for a symmetrical object to be only modeled on one side. Then the finished side can be duplicated and reversed to the other side. Border edges and vertices between the two mesh can be welded to create a proper, final model. The ring-shaped hammer can be offset and extruded inward to create the hole. The two channels around the hole can also be created similarly. The process of adding details to the handgun's hammer is explained in Figure 26.

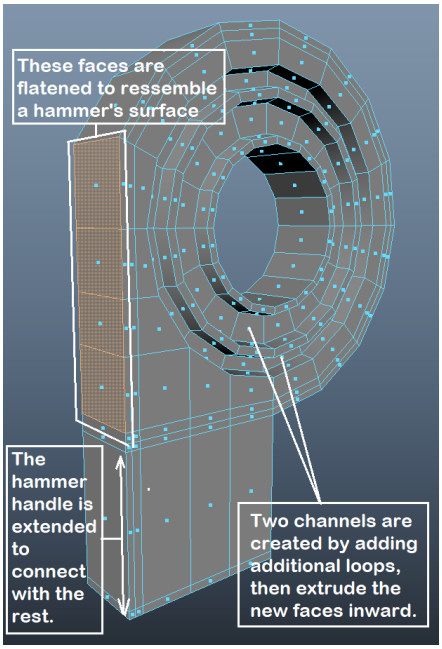

Figure 26: Gun hammer modeling

The body of the gun is divided into 3 different pieces. A front, mid and back piece. Then, a proportion of the surface can be extruded inward to create the concave. To properly bevel the midpiece to match the trigger and concave the face at the same time, sufficient loops can be added.

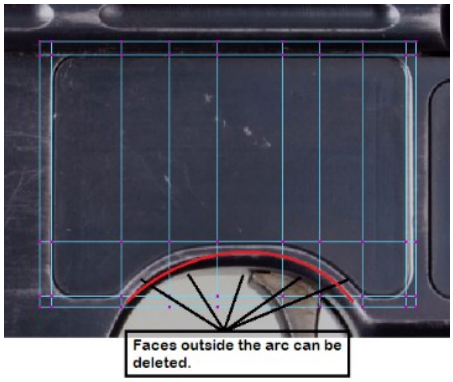

Figure 27: More loops to define the arc

Then all the faces outside the trigger arc can be deleted to match the shape. Figure 28 shows the surfaces that were reverse extruded.

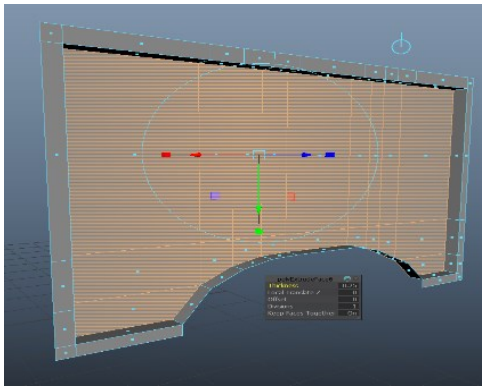

Figure 28: Side plate concave with reverse extrusion

After extruding the surface, the mid-piece can now be combined back with the front and back pieces to form the necessary large piece. As can be seen in Figure 29

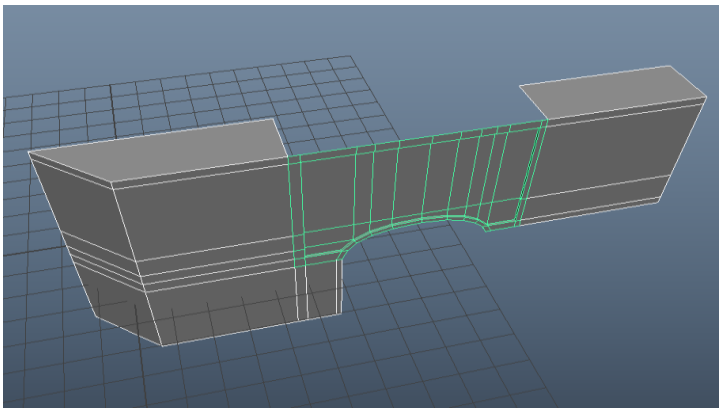

Figure 29: Side plate combined with front and back piece

The hand rest piece that connects the trigger guard with the body can be hand beveled instead of using the bevel tool. The piece itself is a combination of an arc running from the grip to the guard. Then the surface of that arc gradually extrudes downward to create the thickness of the piece. The arc can be created by beveling two-quarter pieces of a cube. Then loops can be added and the vertices by the new loops and the bevel from before can be moved to recreate the piece from reference. Figure 30 shows that hand modeling can be viable in some instances, for some difficult pieces.

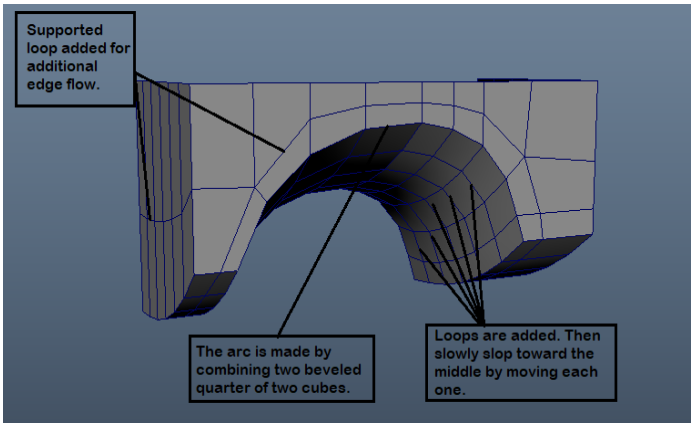

Figure 30: Hand bevel

The slop piece from the ejection port can quickly be drawn as a 2D surface first using the **quad draw** tool, Figure 31 shows the location of the quad draw tool in the modeling toolkit.

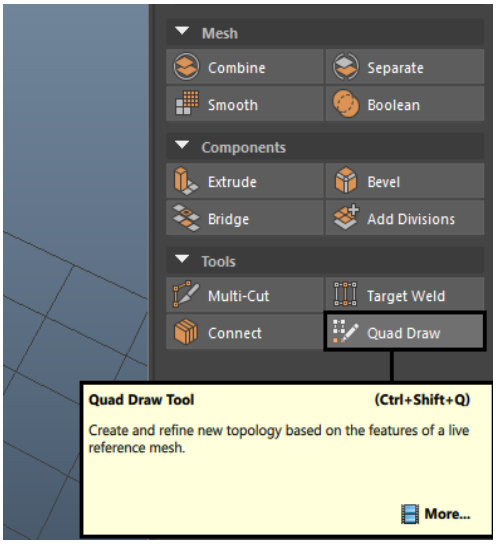

Figure 31: Quad draw tool in the modeling toolkit

In Figure 32, a surface is placed on the top reference, then the quad draw can be used to trace the shape of the slop face. Then the shape of the rounded face is traced by the quad draw tool.

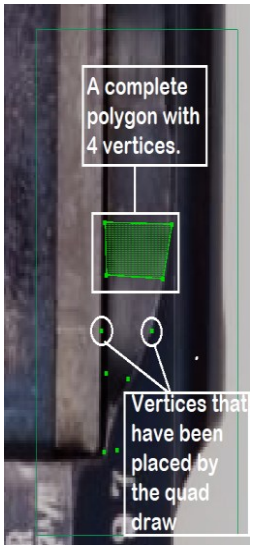

Figure 32: Recreate the shape using the quad draw

Then the surface can be bent by using the **bend deformer**. The bend deformer can be found in the general hot menu, in the **create** sub-menu, and the deformer with enough loops will bend a surface from flat to arc. Figure 33 shows the surface that has a bend deformer applied to it.

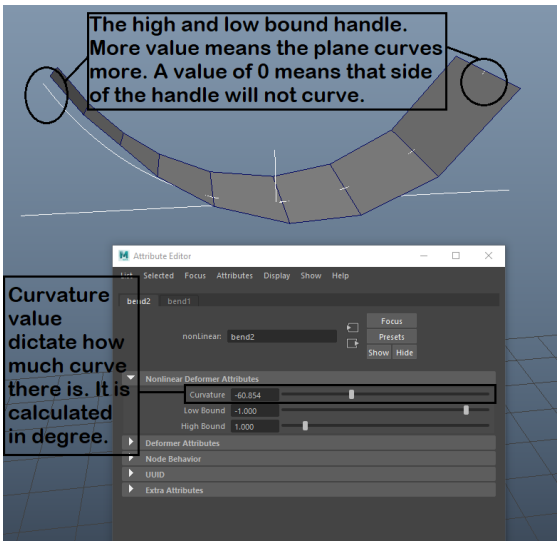

Figure 33: The bend deformer

In Figure 34, after bending the surface, the side edges of the surface can be extruded out and angled outward to create the slop face near the ejection port.

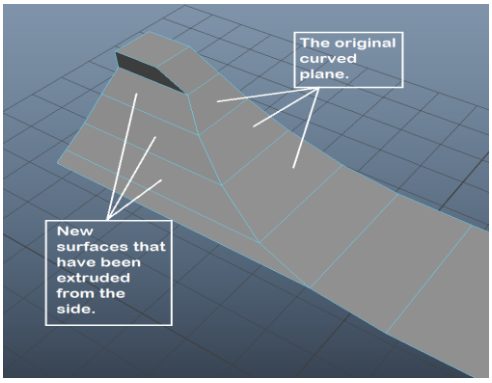

Figure 34: Slop piece

The base above the main body is composed of a rounded surface at the back, and two channels with rounded ends. The two channels have an elongated stadium shape. This can be obtained by splitting a circle plane in half and moving both hemispheres in opposite direction. Figure 35 illustrate two half of a circle plane.

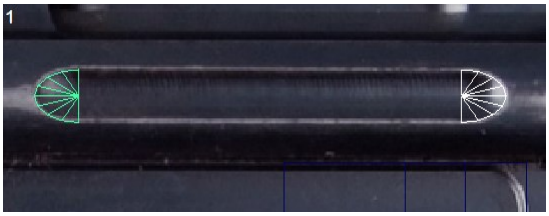

Figure 35: Two half circle line up with the reference

The gap in between them can be bridged back to create a stadium shape. As shown in Figure 36.

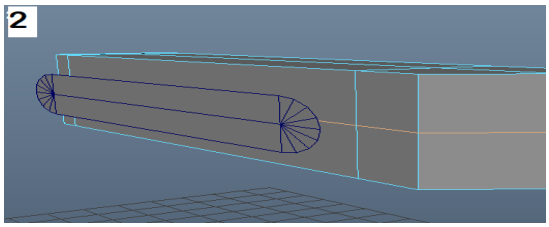

Figure 36: Stadium shape combined with the base

The stadium shape can now combine with a cube to form the base in Figure 37. The number of edges around the stadium shape should match the neighbor edges on the base in order to bridge properly.

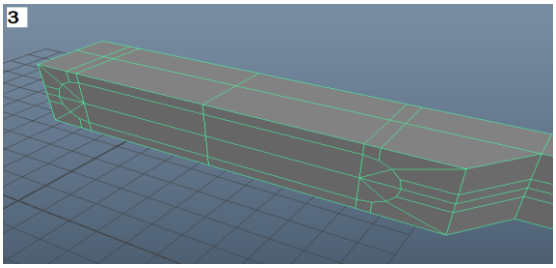

Figure 37: Stadium shape bridged with the surrounding faces

The channel can be concaved by offsetting and extruding the stadium shape inward. The stadium shape needs to be smaller than its original size to produce a channel similar to the references. Figure 38 exhibit the complete stadium shape channel.

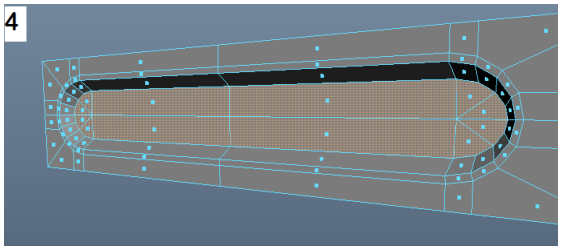

Figure 38: Concaved channel

The grip is half a rectangle on top and half an oval on the bottom, with a slight bulge outward. A smaller face can be pulled away from the grip to create a bulge similar to the reference. Figure 39 shows the shape of the bulging grip.

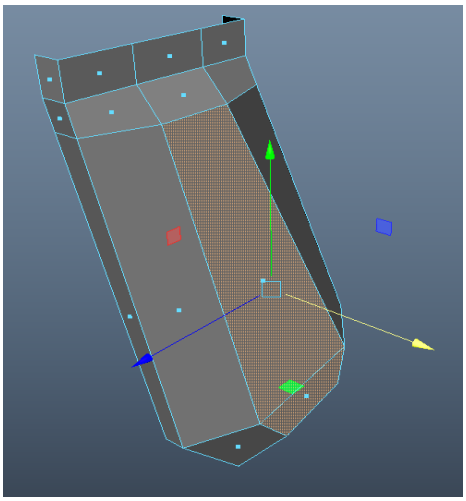

Figure 39: Grip bulge

Loops then can be added in either bevel a single loop or by the multi-cut tool, and each subsequence loop is scaled smaller than the one before. In Figure 40, the bevel tool was use since it is faster to access from the hot menu.

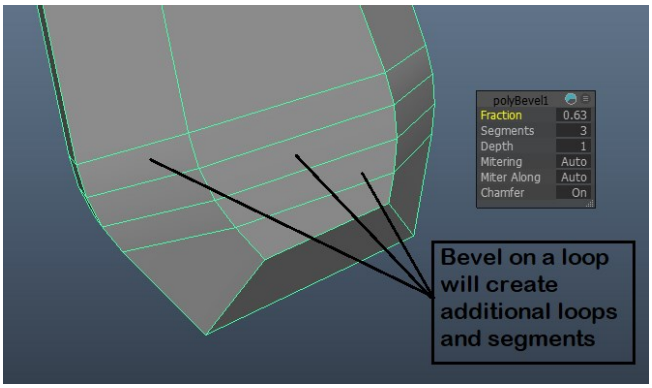

Figure 40: Bevel tool can create additional loops

Loops can then be added to the grip, using the insert edge loop tool for equal distance loops. Illustrated in Figure 41

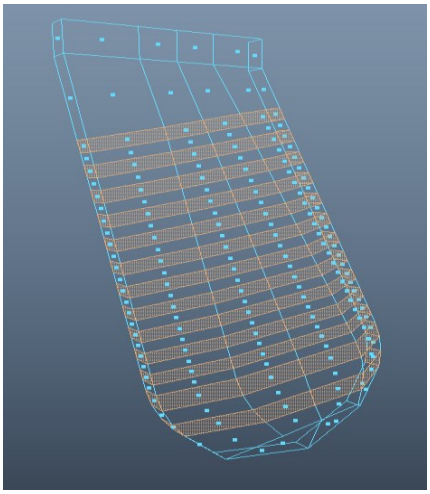

Figure 41: Addition vertical loops to recreate the groove's surface

This will create several smaller faces on the grip, which can be extruded to make the grooves that resemble grooves in the reference. In Figure 42, several faces are extruded at once using only a single extrude command. This is done to keep the groove's thickness consistent with each other.

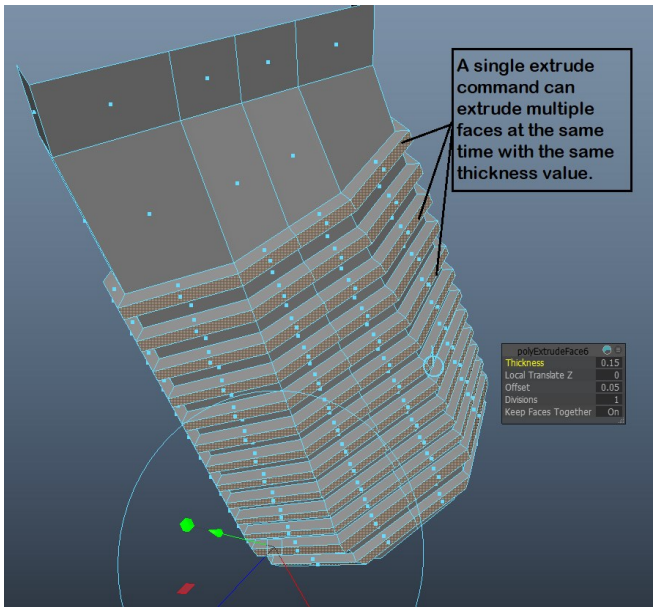

Figure 42: Grip groove extruded

3.3.5 Adding resolution to create high poly model

Holding edges can now be added to hold up a certain part of the mesh. Edges that can take advantage of the collapse and bevel effect when sub-dividing should not have holding edges around them. This step can be done in a separate copy of the original model. Reserve it as the original backup model to fall back on if issues arise. And by up resolution on a different copy of the

model, another instance of the original model can be treated as the low poly count that will be imported into the final project. Eliminate the need for de-resolution. Figure 43 and 44 showcases the model in it high poly, subdivided form. Its edges are crisp and contain zero artifact.

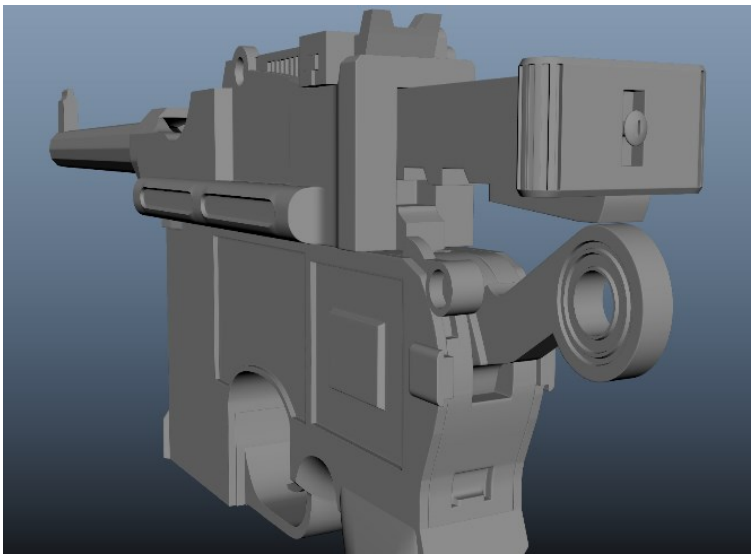

Figure 43: Model finalized and reviewed under 2x subdivision level

Figure 44 showcases the ejection port and the rear sight group under sub division.

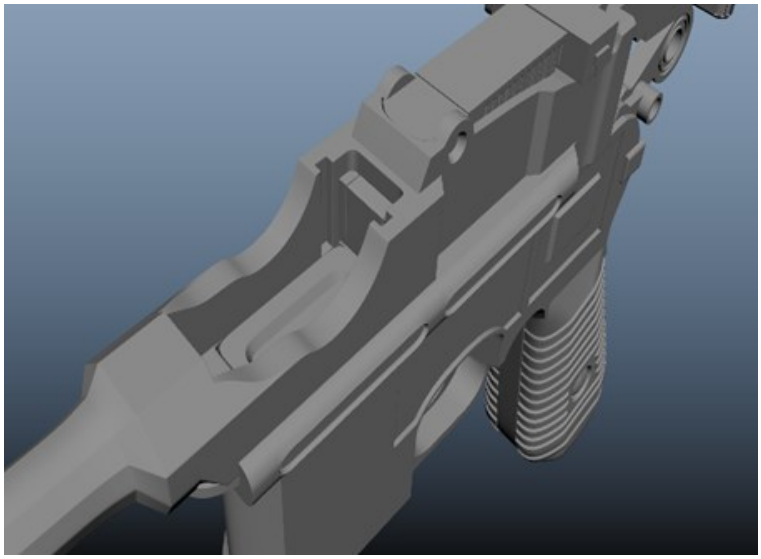

Figure 44: Model top view with 2x subdivision level

Figure 45 going into the ideal of how to take advantage of the subdivision and create a beveled surface. The orange selected edge does not have two holding edge beside it.

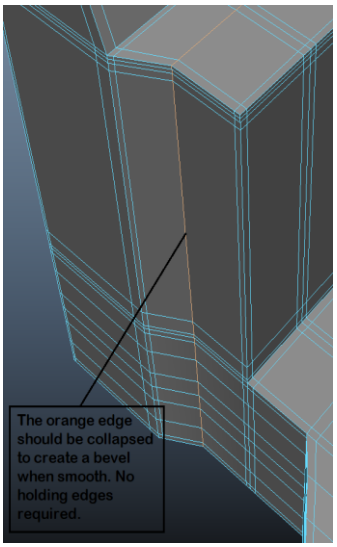

Figure 45: No holding edges can create bevel when subdivision

Both high and low poly model side by side for comparison in Figure 46. The high poly model is populated by more edges, most of them are holding edges.

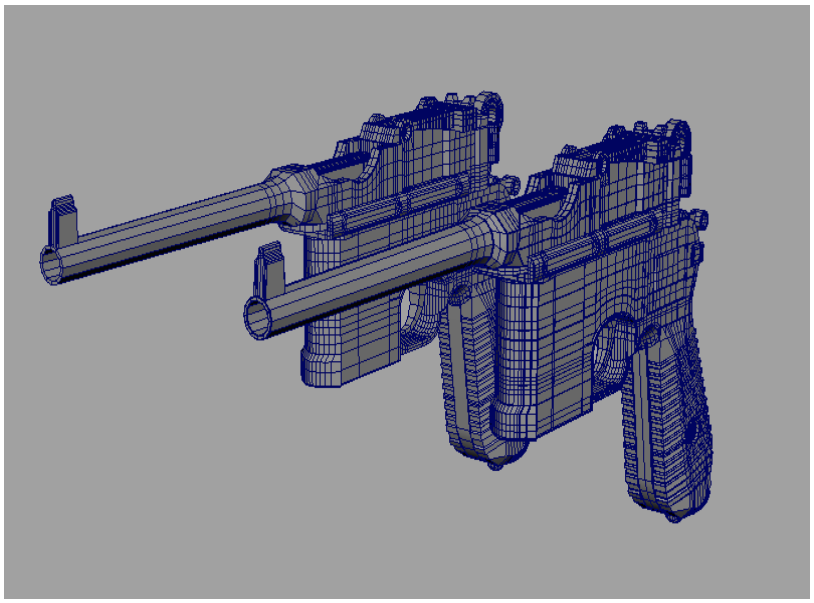

Figure 46: High poly model (right) Low poly model (left) with wireframe

3.3.6 Grouping and naming and suffix assigning

At this point, there will be two model within the scene. A high poly model and a low poly model. Since both models are copies of the original blocked model, they should share similar mesh names. The mesh's names should be identical except for the suffixes. Since baking will locate the correct name and apply the detail of one high poly count mesh to the low poly count mesh. Figure 47 gives a first look into both robot models.

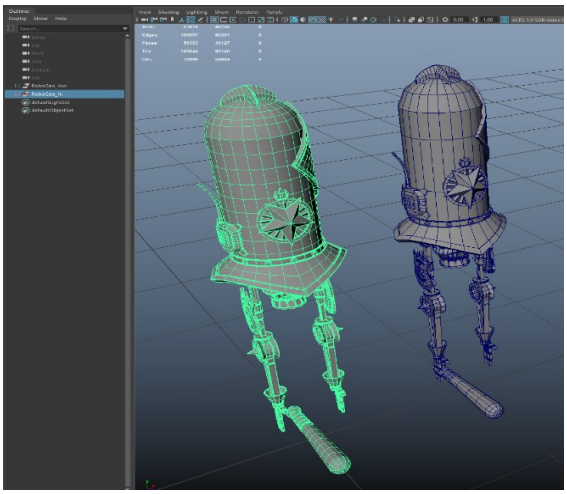

Figure 47: Finished high and low poly models

Figure 48 display the naming and grouping of parts of the high poly model and their appropriate part groups.

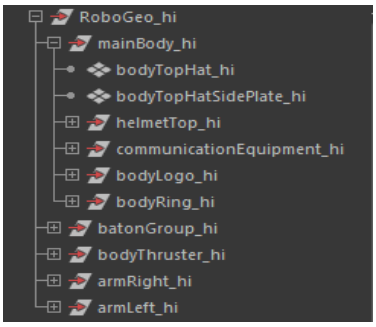

Figure 48: Naming and grouping with suffixes

Figure 49 shows the naming of the low poly model. The structure is identical to that of the high poly one. With the exception of a suffixes.

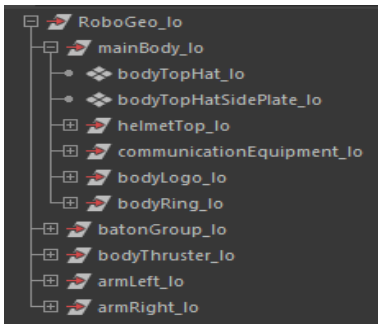

Figure 49: Naming and grouping with suffixes

3.3.7 Unwrap and pack UVs

Unwrapping UV for a low poly model has 5 key steps:

- Create a new, temporary UV to override the exist UV.
- Select where the cut lines, or the UV seams should take place. The seams are what transform a 3D object into one or more 2D UV shells.
- Cut the seams with the application's UV cut tool.
- Unfold the UV shells from 3D back to 2D.
- Layout or packed UV shells within UV tiles properly.

Maya support many methods to rebuild UV. The simplest way is to use **camera-based** projection. The method send out rays from the current view perspective, then cast the shadow of the object onto a plane. The shadow of that object is the new UV shell. Figure 50 exhibit the methods of projecting new UV onto an object.

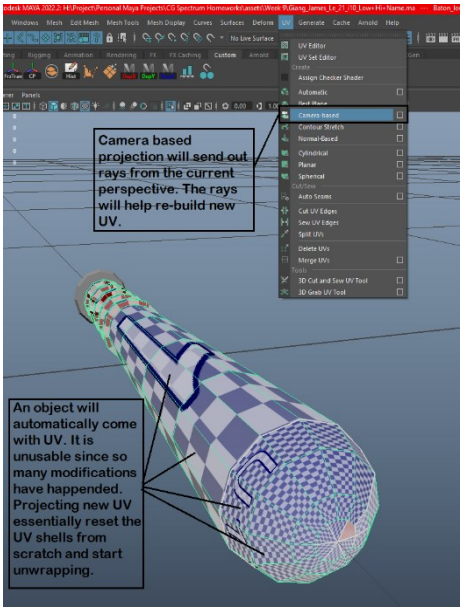

Figure 50: New project UV

While figure 51 display how the newly projected UV appear in the UV editor.

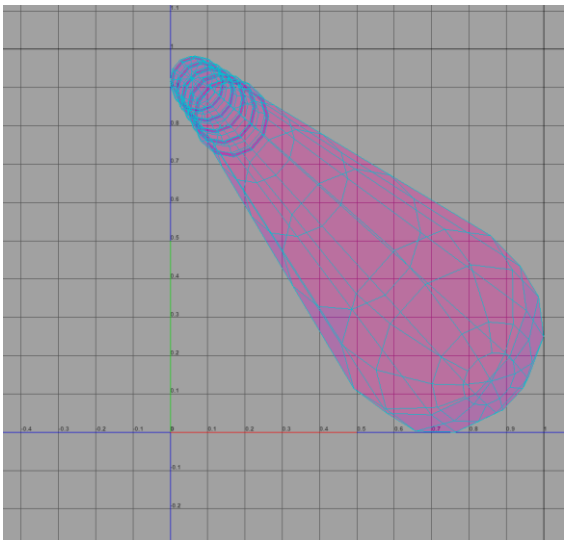

Figure 51: New project UV in the UV edditor

This object consists of a cylindrical body, a rounded head, and an end cap. Therefore, it can be split into three UV shells. The UV seam can be placed where the cylinder starts to transit into half of a hemisphere.

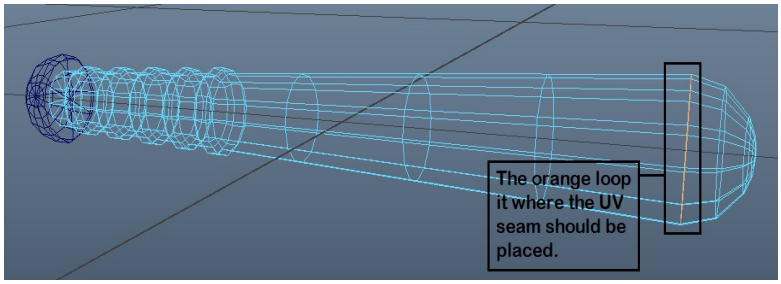

Figure 52: UV seam placement

Then another cut can be placed along the body of the cylinder to open it up. An end cap is a circle plane, so the seam is the border edge between the cap and the body.

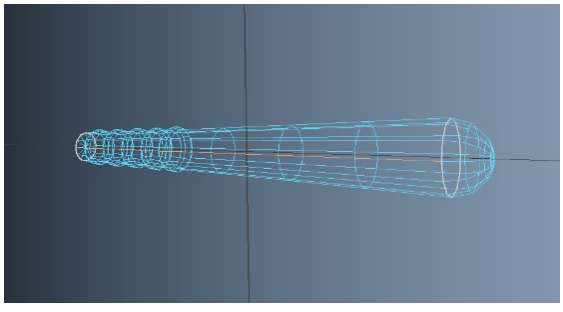

Figure 53: UV seam horizontal placement

Figure 54 shows how to cut a line and turn it into a UV seam. The edge can now be turned into a UV seam by the **cut** tool, which is located in the hot menu when hovering in the UV windows. The tool will cut every edge that are being selected. Then all three shells can now be unfolded by using the **unfold** tool located above the cut tool. The result is two circular planes and a rectangle shape plane.

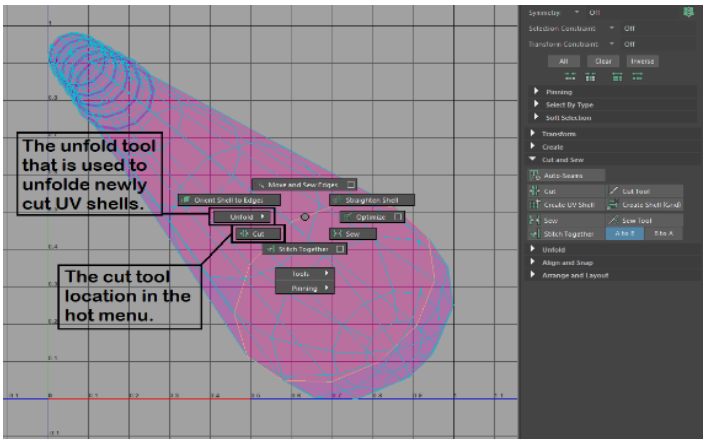

Figure 54: Seam cut tool

Figure 55 shows the unfolded shells of the baton after seams are placed. When unfolding, the resolution of the shells will automatically match the current tile. This can be later hand packed to save space for other shells.

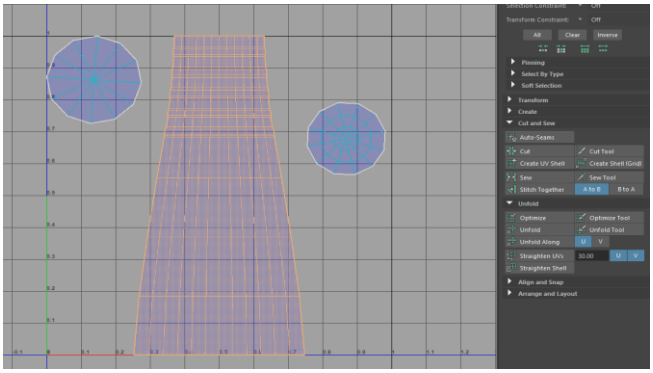

Figure 55: UV shells unfolded

With UV shells properly cut and unfolded, they can now be packed in a UV tile. This can be done in two ways, hand packing individual shells for more precision and management, but take significantly longer for a large object. Or using the automatic pack UV function of the 3D application. It will save time and the shells mostly spread out. However, the function will not prioritize

which shells need more space for more resolution. Therefore, packing UV can be done by both hand and automatic. Figure 56 shows the shells of the robot's hand, finger and the baton and how they are hand tiling as well as resolution prioritized.

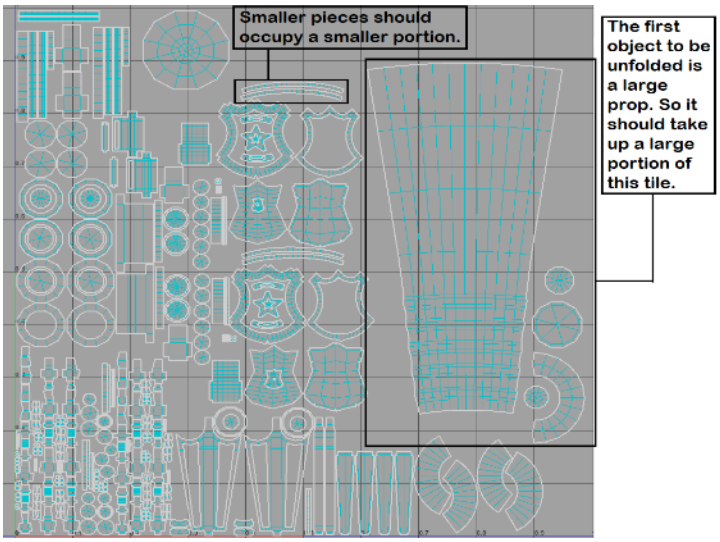

Figure 56: UV shells priorty

According to a published article from UDIM Workflow (2016), recent UV packing techniques and applications have been implementing a UDIM workflow. It will allow for multiple tiles of UV to exist, instead of all shells needing to lie on top of a single or two tiles to save memory. UDIM allows for renderers and game engines to detect and correctly assign UV shells from different tiles to their original meshes. The robot police model with UDIM enable has four UV tiles.

3.3.8 Exporting model for texturing

With UV shells properly packed in the low poly model. A final **clean-up** tool can be run on the low poly model to detect and fix potential issues before export. The tool has several options and can be set to either detect or detect and apply fixes.

Then each part of the low poly model can be assigned with identification materials. These materials will carry into the texturing application. Allowing multiple parts with the same materials to be painted, apply effects and masks together. Figure 57 shows the cleanup tool and its options.

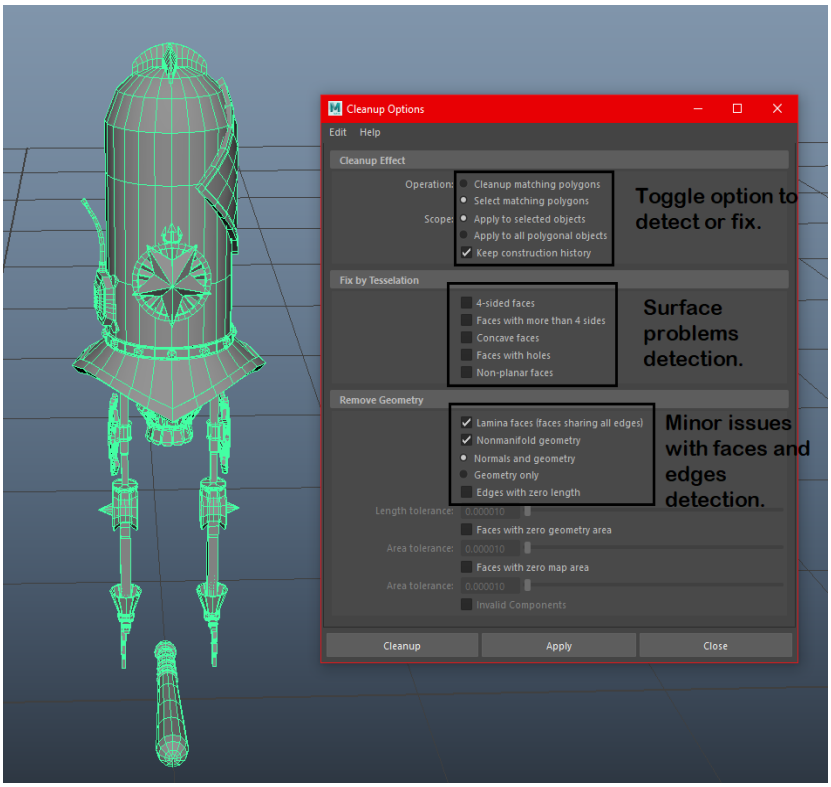

Figure 57: Cleanup tool

After running the cleanup tool and assigning identification materials on the low poly model, both models should be their files. Because Substance Painter will accept separated files better. The low poly model now receives a softening edge treatment. As mentioned above, softened edges will average the normals out, and support the later baking process. Finally, each model file can be exported out as a .FBX file for both texturing and final rendering.

3.4 Texturing Phase

3.4.1 Importing model and bake texture maps

When bringing a model into Substance Painter, the application will ask for a resolution. For this model, a 2K resolution was chosen. However, resolution can be up scale during the texturing process. The template for texturing was the PBR Metallics Roughness standard with UDIM enabled. Figure 58 goes into the baking option in Substance Painter.

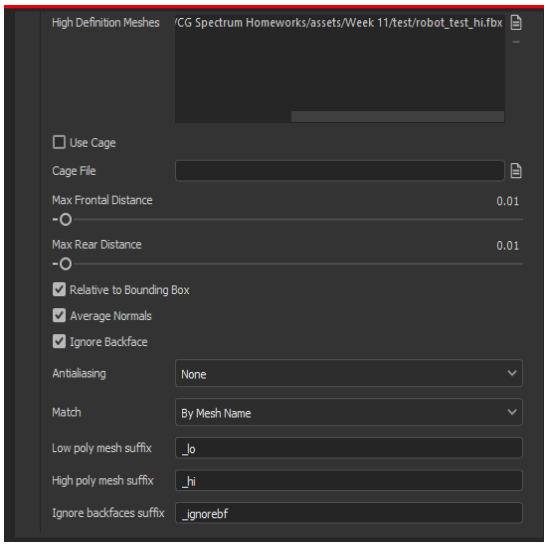

Figure 58: Baking option

The model can bake with details generated from the high poly model. Within the baking windows, the first important setting is to define the location of the high poly model. Then there are the suffix, which was set during naming and grouping meshes together. Incorrect suffix identification will fail the baking process. Substance Painter will also ask which identification baker parameter to use for baking. Since the low poly model was assigned with different materials, material colors is used. After baking is completed, the low poly model will now carry the details and resolution of the high poly model.

3.4.2 Adding ID layers and folder groups

Substance Painter support folders to house and sort a different type of materials and effects. Folders can have their names changed to represent parts that they are affecting, and can be nested into each other. Figure 59 display the value sliders to develop a surface with metalness. As well as showcasing the folder that contain the material.

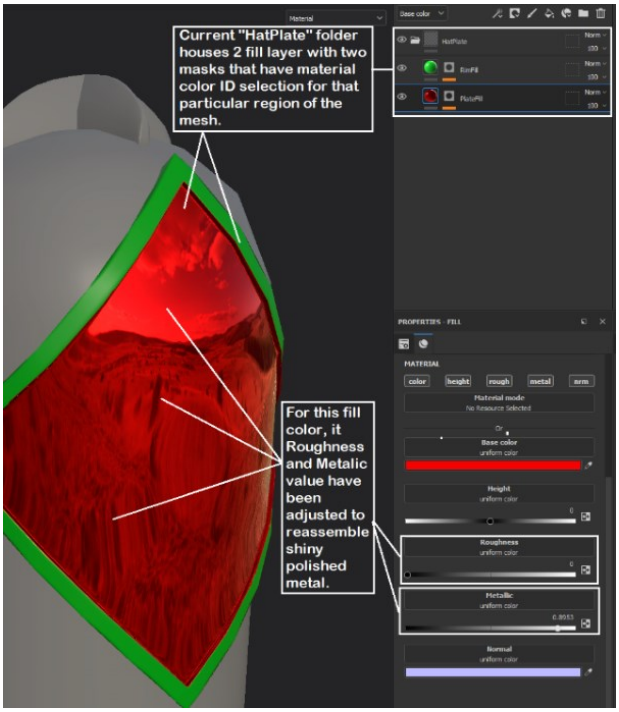

Figure 59: PBR value in Substance Painter

The low poly model can now have its parts and meshes identified by using fill layers and black masks with color selection. Using this method, parts that share the same material can be painted quickly and still share consistency. A fill layer is a layer that can fill an entire mesh at the same time. Fill layers also come with PBR channels like normal, height, and metallic, which can be toggled on or of and have their parameter adjusted. A mask is a general method to blend different materials. There is a suite of different blending effects that a mask can have. One of them is the ability to select and remember which materials have been chosen to apply paint.

3.4.3 Adding additional procedural effects and smart material

After identifying all parts of the robot, more materials, masks, effects, and smart materials can be applied. The goal is to break up the blandness and flatness of the surfaces. First, a base fill layer of metallic paint is applied. Then another silver metallic paint is applied on top of the base layer, and it is accompanied by a black mask with an edge wear effect. This effect automatically detects edges that are from 60 degrees and beyond, and applies the silver paint. Creating an edge-damaged effect with the base layer. Then, additional fill layers with masks that carry dust effect and random scratch generator are applied. Dust and scratch effects are procedurally

generated. Then finally, some decals are added onto the surface for more aesthetic appeal. However, the decals need to be underneath the dust. Since Substance Painter will read materials from the bottom up, the decal needs to also be dusty and scratched. The surface will now develop more captivatively and less flat than before. Figure 60 shows how multiple material, masks and effects can stack up and create a more developed, interesting surface for the robot's hat, which is the largest piece of the model.

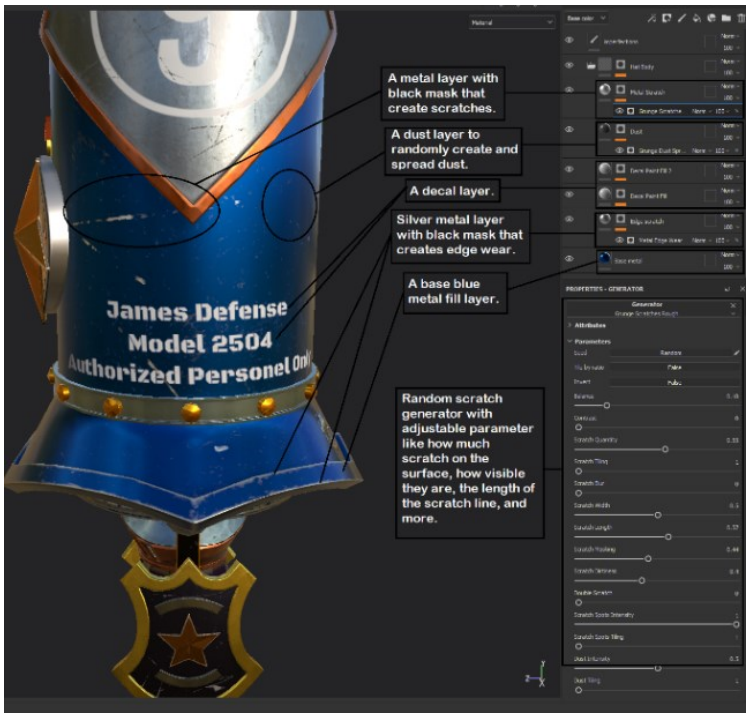

Figure 60: Procedural effects and random generators

Smart materials are set of materials, layers, masks and random generators that have been altered and combined together to form a unique set of advanced materials. Smart materials have the advantage of quick to apply. Figure 61 is showing how parts will develop when applying a smart material.

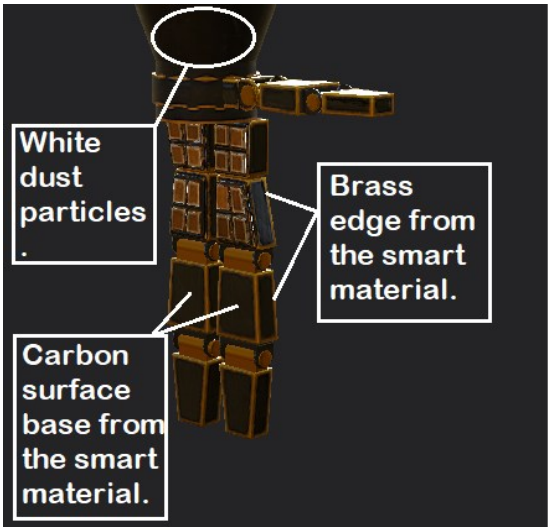

Figure 61: Mesh with smart material applied

Figure 62 goes into detail what content is in the smart material folder. Each individual element within a set can be adjusted if necessary.

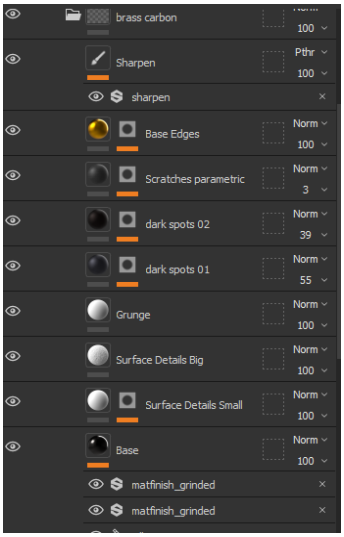

Figure 62: A smart material folder

3.4.4 Exporting texture files for render

After texturing, the texture of the robot can now be exported as images. Substance Painter support wide export templates for different renderers and software. For standard regular export, the robot texture can use the PBR Metallic Roughness and be exported as a 2k .PNG images file. The template also comes with UDIM enabled. Therefore all images will be UDIM compatible. Figure 63 display the export template for the robot's texture files.

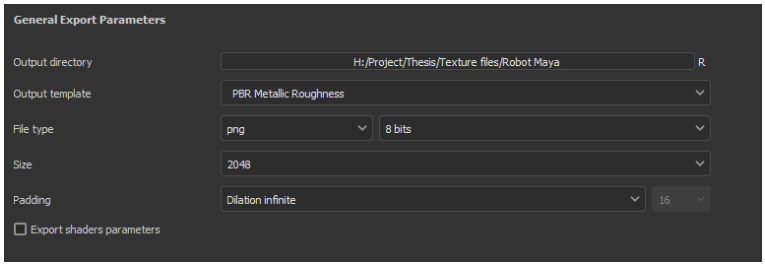

Figure 63: Exporting windows

There will be four sets of textures for the robot model after exporting. They are:

- The base color texture with no light information.
- Metalness map, which shows the metal region as white and the non-metal region as black.
- Normal map, which was baked from the high poly model.
- Roughness map, which displays region with glossiness value.

Each texture map above represents the PBR value of the first of two UV tiles of the robot model. Since UDIM was enabled, each texture set has 2 maps. After exporting, Figure 64 shows the base color of the robot model with no light information.

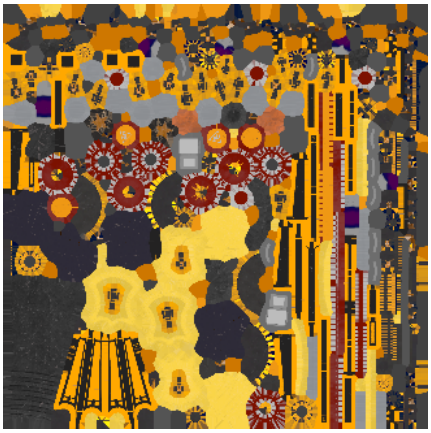

Figure 64: Base color texture map

Figure 65 is the metalness map of the robot model. UV shells are white are identified as metal surfaces. While black is a non-metal surfaces.

Figure 65: Metalness texture map

In figure 66, the roughness, or glossiness map explains how a surface would react with light and reflection in grey scale. Regions that are closer to a white color are less reflective material (such as rubber surface). On the other hand, regions that have a darker color are more reflective (such as a shiny metal surface)

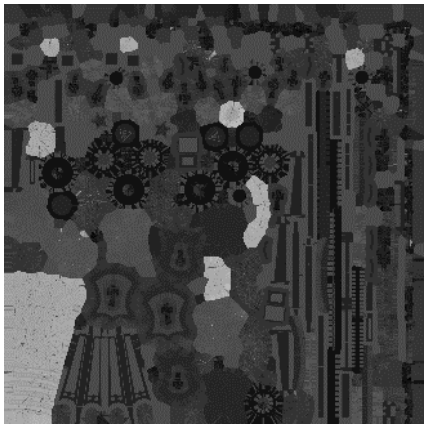

Figure 66: Roughness texture map

Figure 67 displays a normal map of a model. Normals are compiled from three channel color (red, blue and green) to create the illusion of details such as crack and holes, without having to model those details. Normals are baked

from high resolution model to it low resolution counterpart. Combining a high level of details and a light, memory efficient model.

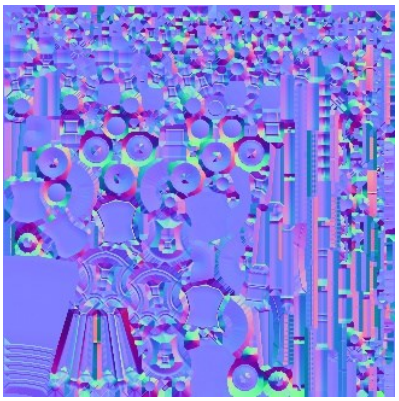

Figure 67: Normal texture map

3.5 Rendering Phase

3.5.1 Common practices for PBR texture files

In Unreal Engine 4 (UE4) and Maya, the texture file can be imported and applied back to a model using a node-based system. The node-based system gives more flexibility when applying a certain texture style. In UE4, an option called virtual texture need to be enabled to use the UDIM format. An empty material can be used to bring in the texture file. Open the material editor will show a node graph that allows for multiple channel creation and plug-in. Both the base colour and the normal map will plug into the sample material through the RGB channel. This means that both of those texture maps only output one type of channel. However, the occlusion, roughness, and metallic information were stored as one set of texture files instead of three. Therefore the node that houses this texture set will output three separate channels. Red for occlusion, green for roughness, and blue for metallic. After all, nodes have been configured and plugged in properly, the sample material can be dragged to the imported robot model mesh. Figure 68 displays the material editor windows of UE4. This is the place to reimport texture files and have their information channels sorted into a sample material. This sample material then can be re-applied back onto a model.

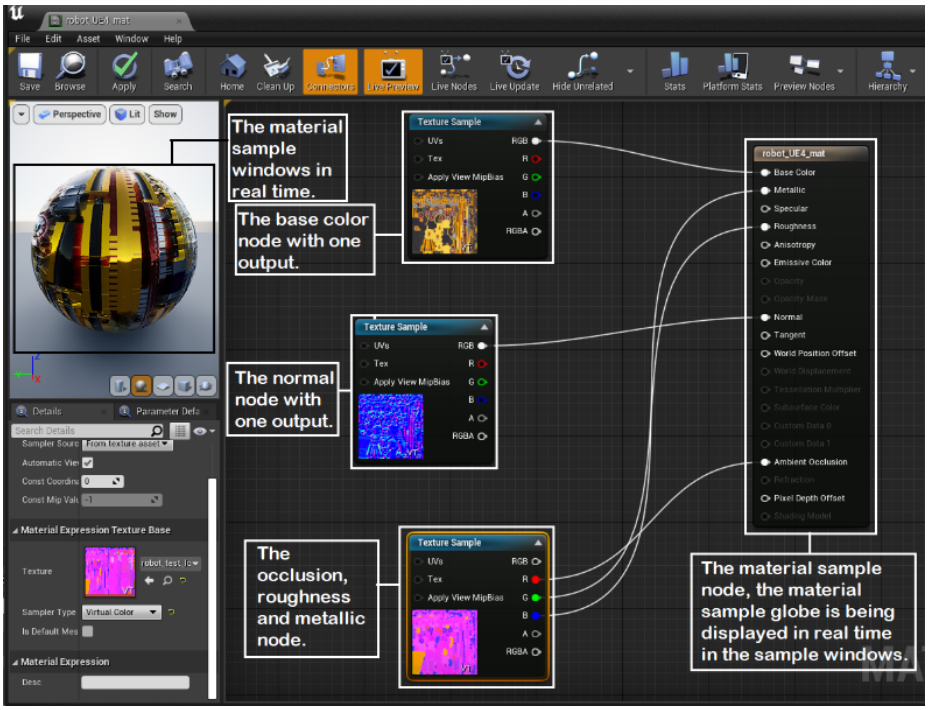

Figure 68: Texture node editor and model with texture

Figure 69 exhibits the re-textured robot within UE4's environment. The sample material works with the UDIM system, or virtual texture in UE4 to identify and apply texture to their correct parts.

Figure 69: Sample material can now be dragged and applied to the model

Substance Painter's Iray utilizes HDR images to set up the rendering environment. HDR or High Dynamic Range images are very high-resolution images that were taken from a continuous shot from a 360-degree rotate camera on one spot. These images will be wrapped around the inside of a sphere that has its normals flipped inward. Then the renderer will use the color

information on those images to cast a ray into the scene and create a surrounding environment lighting. This is quick to set up and the environment for testing and rendering can be changed by swapping out HDR images. Maya's Arnold and UE4 can also be set up to use HDR images, Figure 70 show an example of an HDR image.

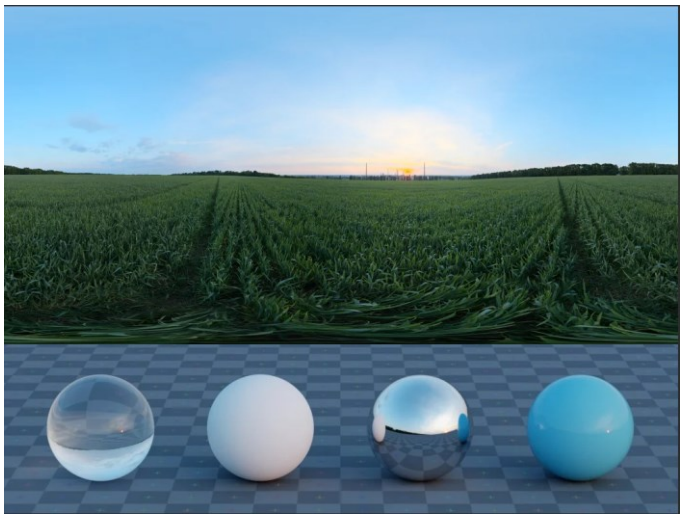

Figure 70: An example of a 4K HDR image

3.5.2 Substance Painter's Iray renderer

The final render for the robot model was taken at a camera setting of a field of view of 49 degrees, a focal length of 17 mm, a focus distance of 2.38, and an aperture value of 0. The render image resolution was 2560x1440. The environment chosen was a studio-style HDR white background. The final render for the robot model was taken at a camera setting of a field of view of 49 degrees, a focal length of 17 mm, a focus distance of 2.38, and an aperture value of 0. The render image resolution was 2560x1440. The environment chosen was a studio-style HDR black with white light background. Figure 71, 72, 73 and 74 showcase the completed renders of the robot using Iray. Render time within Iray depends on how many samples and the target resolution the renderers are working. Both renders have the option to limit how many samples and how much time are allowed to render. Figure 71, 72, 73 and 74 was rendered for 1 minutes 32 seconds with 1000 iterations limit.

Figure 71: Iray render, overall view

Figure 72: Side view

Figure 73: Back view

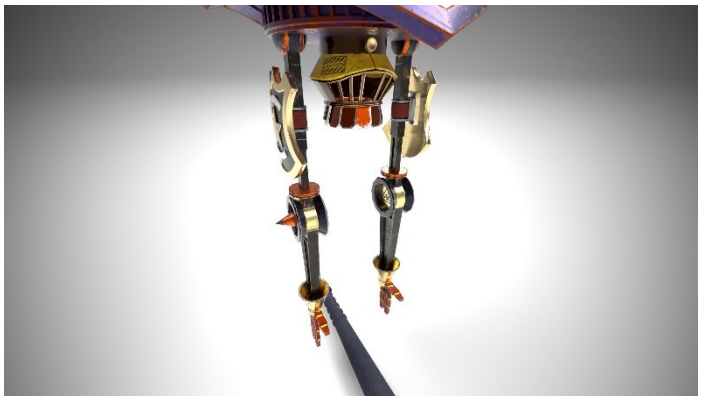

Figure 74: Front view close up

Iray uses HDR images as its primary environment setup. It also uses the current perspective camera as it renders the camera. The camera setup includes a field of view slider, a focal length slider, a focus distance slider, and an aperture slider. Additional post processing effects are available in Iray. A vignette parameter allows for the camera to focus on an oval shape around the centre of the scene. And a glare parameter, which reflects light from glossy surfaces.

3.5.3 Maya Arnold renderer

Maya needs to setup its render scene for Arnold to function correctly. Maya also imports and applies it texture similar to UE4, through a node-base system. Arnold uses which ever camera is being selected. So Maya can setup multiple camera angle within a scene, then cycle through them if necessary for good angles. Camera settings are also similar to those of Iray. Like texture, the render environment need to be setup. The setup are rather simple, a large curve surface with two large wall on two sides. The scene can be lit up using the Arnold's AI area light, it functionality is similar to that of a real panel light. The light has two crucial settings. Intensity and exposure, the former tell how bright the light source is and the latter is an exponential multiplier of intensity. Figure 75 show how a basic render scene, with walls and light, is setup.

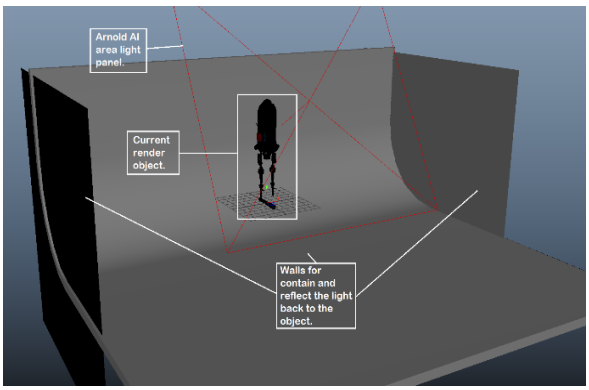

Figure 75: Maya render scene setup

The final Arnold renders were taken in a simple walled environment, with a camera setup that has a field of view of 54 degrees. A focal length of 35 and an aperture of 36mm. Figure 76, 77, 78 and 79 showcase the renders from Maya Arnold, with basic environment and light setup, with modified camera setting. All four render images were render at a 1920x1080 resolution with a

render time of 2 minutes 50 seconds with a sample limit of 1000 iterations. The render shots were compiled on an Intel core i5 9500F, a model that does not have a small number of cores and threads.

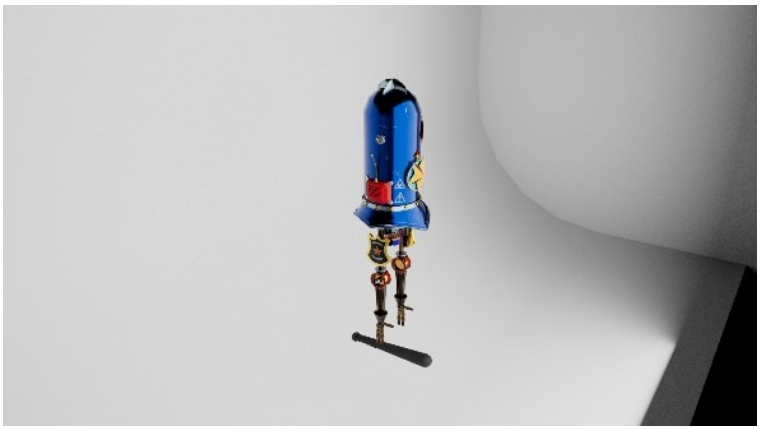

Figure 76: Maya Arnold renders, overview

Figure 77: Side overview left side

Figure 78: Under view left side

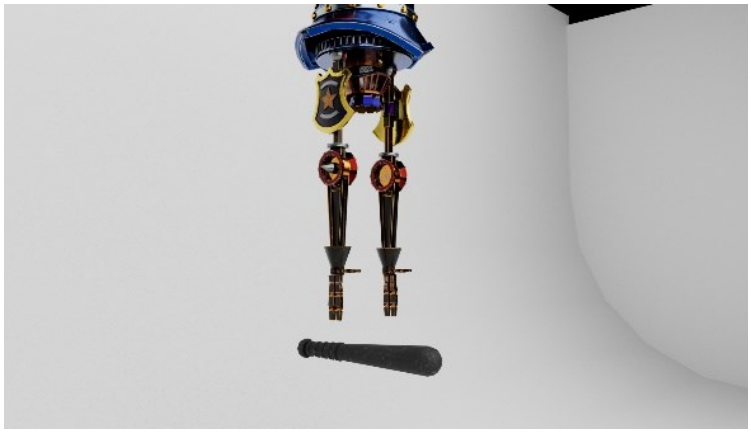

Figure 79: Under view right side

3.5.4 Unreal Engine scene setup and screenshots

In UE4, after setting up a project, there are two ways to bring an object from a 3D application like Maya into UE4. The first is regular export as a .FBX file then import it into UE4. Or use the direct export to engine option from the 3D application. UE4 support multiple types of lighting options for a scene. A default directional light that acts to light the Sun in a world. There are cones and point lights for more traditional scene building. There is support for HDR image lighting, also known as HDRi backdrop. The HDRi backdrop was chosen for consistency and a simple setup. In terms of camera options, there are two important values. Focal length and aperture value. UE4 does support the camera post-processing effect more in-depth than Iray. And camera sets ranging from filming settings to video game settings. UE4 only supports taking a screenshot with the current camera. However, the screenshots are taking in real-time render.

The screenshots were taken from a standard camera, with a focal length of 30mm, a field of view of 60 degrees, an aperture of 25mm and no post processing. Figure 81, 82, 83 and 84 showcase the screenshots inside UE4, with simple outside HDR image environment, no post processing and modified camera settings. All four render images took around 37 seconds to rebuild

light and complete a scene. The render, or rebuild time in UE4 are various from lighting condition to how complex a scene is.

Figure 80: UE4 screenshots, left side view

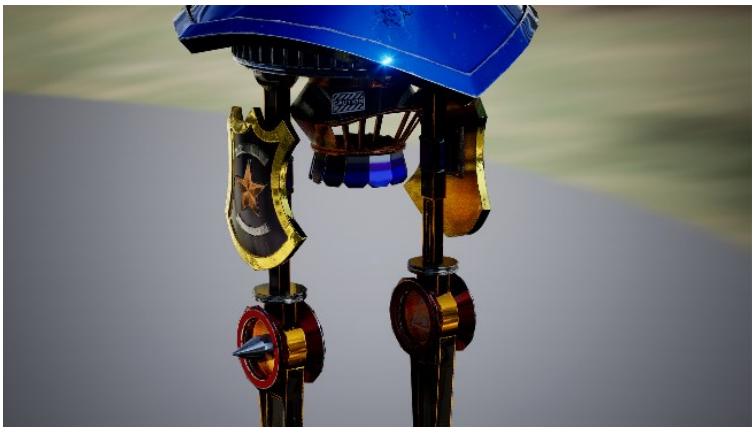

Figure 81: Close up view

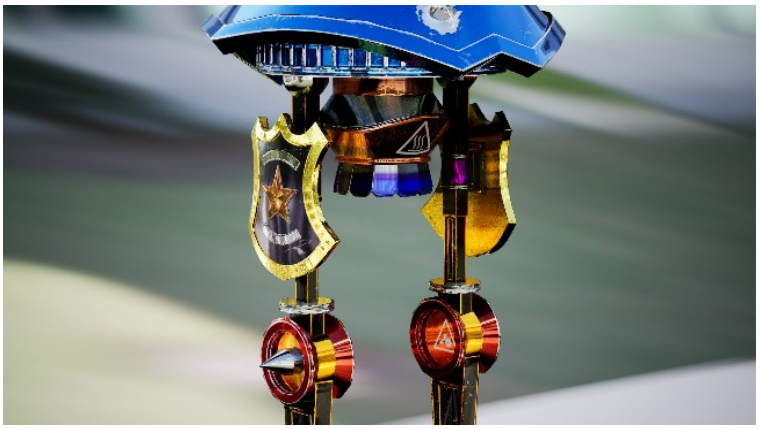

Figure 82: Close up back view

Figure 83: Overview

3.5.5 Renderer conclusion

All three renderers have shown their capability in terms of rendering the robot model. Three renderers were using CPU as the primary processor, which has the benefit of increasing color accuracy and surface development. With the disadvantage of slow render time in larger and more complex scenes depending on the CPU generation.

Iray is a native, lightweight renderer inside Substance Painter that allows models to quickly test their surface development and how the materials would react to a different set of environments.

Maya's Arnold utilizes a more streamlined node-based system to re-integrate texture back into a model. Arnold also provides an accurate result thanks to its sampling adjustment, complex scene, and lighting setup.

Although UE4 does not have a dedicated renderer, it is a production environment. Therefore what inside a scene after rebuild is going to be like that in the release version of a UE4 project. UE4 allows for a higher complexity of scene building by incorporating elements such as a VFX file, cinematic camera setup and record, and a physic base backbone engine. UE4 also uses a node-based for texture re-importing.

4 CONCLUSION

The handgun model was recreated using references and successfully up scaled to have better quality for later baking and texturing. This completes the exploration of modeling an object, from referencing to high poly upscale. And with the robot model has proper texture and procedural effects as well is beautifully rendered with a different application. The texturing and rendering phases are completed. Both models can now be further passed down into other departments and eventually, present in the final version of a digital project. Finished models can be modified to reuse and save time. Overall, the

power and capability of all three software are on display. Maya has industry-standard methods and can model nearly every object. Maya can also performs mass instancing to build cities and massive cinematic scene. Substance Painter and the suite of materials, masks, generators, and smart material, brings color and texture to models. And Unreal Engine 4 is a powerful hosting engine for any type of project, from video games to cinematic expression and entertainment.

REFERENCES

Boi, Y. (2021, December 24). *HISTORY OF 3D MODELING: FROM EUCLID TO 3D PRINTING*. Ufo 3d. Web article.
Available at: https://ufo3d.com/history-of-3d-modeling/
[Accessed March 21, 2022]

Gamers Nexus. (2015, March 13). *What is PBR? Physically-Based Rendering Explained* [Video]. YouTube.
Available at: https://www.youtube.com/watch?v=7NjGETJMZvY
[Accessed 20 March 2022].

Lemos, C. (2020, March 24). *Tutorial: How Normal Maps Work & Baking Process*. 80.Lv. Web article.
Available at: https://80.lv/articles/tutorial-how-normal-maps-work-baking-process
[Accessed March 19, 2022]

McDermott, W. (2018, February). *The PBR Guide - Part 1 on Substance 3D Tutorials*. Substance3D. Web article.
Available at:
https://substance3d.adobe.com/tutorials/courses/the-pbr-guide-part-1
[Accessed March 20, 2022]

UDIM Workflow. (2016). Learn.Foudry.Com. Web article.
Available at:
https://learn.foundry.com/modo/901/content/help/pages/uving/udim_workflow.html
[Accessed April 19, 2022]

Wikipedia contributors. (2022a, March 17). *Polygon*. Wikipedia. WWW document.
Available at: https://en.wikipedia.org/wiki/Polygon
[Accessed March 19, 2022]

Wikipedia contributors. (2022b, April 1). *Texture mapping*. Wikipedia. WWW document.
Available at: https://en.wikipedia.org/wiki/Texture_mapping
[Accessed March 19, 2022]

Wikipedia contributors. (2022c, April 16). *Low poly*. Wikipedia. WWW document.
Available at: https://en.wikipedia.org/wiki/Low_poly
[Accessed March 19, 2022]

Wikipedia contributors. (2022d, April 18). *Rendering (computer graphics)*. Wikipedia. WWW document.

Available at: https://en.wikipedia.org/wiki/Rendering_(computer_graphics)

[Accessed March 24, 2022]

LIST OF FIGURES