Bachelor's thesis

Game Technology

2018

Antti Lindsten

EXPLORING MODERN METHODS OF 3D ASSET CREATION

TURKU AMK

TURKU UNIVERSITY OF APPLIED SCIENCES

BACHELOR’S THESIS | ABSTRACT

TURKU UNIVERSITY OF APPLIED SCIENCES

Information and Communications Technology

2018 | Pages: 38

Supervisor: Principal Lecturer Mika Luimula, Adj. Prof

Antti Lindsten

EXPLORING MODERN METHODS OF 3D ASSET CREATION

The aim of the thesis is to research the modern methods of 3D modeling. Because the methods are evolving at a rapid rate, some of the methods are already outdated and replaced with more modern approaches in the industry. Finding relevant data can be difficult, and the aim of the thesis is to find the relevant information and create an intact paper that can be used for reference in the future. There are many stages of 3D asset creation, and all of them are researched in the thesis. The thesis begins with an explanation of the fundamentals of 3D modeling and later on moves to the practical part, in which the researched methods are used to create 3D models.

The majority of the information was found online, from various well-known websites and forums. Researching online seemed to be the best approach for this kind of research. Professional 3D artists commonly share their knowledge on forums, which was utilized for research purposes. The results of the research were applied to create 3D characters for the fire safety simulation of the Turku Game Lab. The models were created in Blender and textured in Substance Painter.

The objectives of the thesis were successful, as the new models looked more appealing with similar time investment when compared to the old models. By using the new methods, the character models were more realistic than before, and some laborious steps could be skipped entirely, saving time and resources.

As a conclusion, it is crucial for modelers to invest time to research modern methods of 3D modeling. This saves time in the long run and the models will be more appealing.

KEYWORDS:

3D Modeling, Production Pipeline, Real-time Rendering

ASIASANAT:

OPINNÄYTETYÖ (AMK | TIIVISTELMÄ

TURUN AMMATTIKORKEAKOULU

Tieto- ja viestintätekniikka

2018 | 38 sivua

Ohjaaja: yliopettaja Mika Luimula, dos.

Antti Lindsten

3D-MALLINTAMISEN MODERNIT TYÖMENETELMÄT

Opinnäytetyön tavoitteena on tutkia 3D-mallintamisen moderneja työmenetelmiä reaaliaikaista hahmontamista varten. Koska mallinnustekniikat kehittyvät nopeaa tahtia, monet aikaisemmin käytetyt menetelmät ovat jo vanhentuneet ja korvattu modernimmalla lähestymistavalla. Relevanttien lähteiden löytäminen voi osoittautua vaikeaksi, ja työn tavoitteena on löytää modernit työtavat vanhentuneiden tapojen joukosta ja muodostaa tiivis kokonaisuus, joka antaa kattavan yleiskuvan kokonaisprosessista. Mallinnukseen liittyvässä työnkulussa on lukuisia eri vaiheita. Opinnäytetyö alkaa teoriaosuudella jossa pohjustetaan mallinnukseen liittyviä perusasioita ja -käsitteitä. Tämän jälkeen siirrytään käytännön osuuteen jonka tavoitteena on luoda hahmomalleja tutkittujen menetelmien mukaisesti. Koska jokaiseen mallinnusvaiheeseen liittyy paljon yksityiskohtia ja monimutkaisuuksia, kaikki prosessit tulee tutkia ja käydä läpi tarkasti.

Pääosa tutkimuksesta tehtiin Internet-lähteiden pohjalta, käyttämällä lähteinä lukuisia artikkeleita ja keskustelupalstaviestejä. Artikkelit on valittu tunnetuilta sivustoilta, varmistaen tietojen luotettavuuden. Tämän lisäksi lähteenä on käytetty pelifirmojen julkaisuja joissa kerrotaan heidän lähestymistavoista. Keskustelupalstoilta kerätyt tiedot ovat peräisin ihmisiltä jotka työskentelevät alalla ja ovat tietoisia nykyisistä metodeista. Käytännön osuudessa käytettiin mallinnusohjelmana Blenderiä josta mallit siirrettiin Substance Painteriin viimeistelyä varten. Malleja käytettiin Turku Game Labin paloturvallisuus-simulaatiossa.

Tutkimuksen tulokset osoittautuvat lupaaviksi, ja näin ollen opinnäytetyön tavoitteissa onnistuttiin. Uudet hahmomallit näyttivät realistisemmilta kuin vanhoilla menetelmillä tehtynä. Lisäksi uudet menetelmät helpottivat mallintamista yksinkertaistamalla monia vaiheita mallinnusprosessissa.

Johtopäätöksenä voidaan todeta, että uusien menetelmien tutkimiseen kannattaa käyttää aikaa ja vaivaa. Käytetty aika maksaa itsensä takaisin realististen 3D-mallien muodossa ja leikkaamalla monia turhia työvaiheita.

ASIASANAT:

3D-mallinnus, Prosessivaiheet, Reaaliaikainen hahmontaminen

CONTENTS

PICTURES

LIST OF ABBREVIATIONS (OR) SYMBOLS

AAA	Classification for video games produced by a big publisher
BIM	Building information modeling
CGI	Computer-generated imagery
CAD	Computer aided design
FPS	First person shooter
NDA	Non-Disclosure Agreement
NPOT	Non-Power-Of-Two, in context of textures
PBR	Physically-Based Rendering
RGB	Color mode specified with red, green, and blue channels
VR	Virtual reality
UV	Coordinates used for Texturing
2D	Two-dimensional
3D	Three-dimensional

1 INTRODUCTION

The aim of this thesis is to first provide an introduction to 3D modeling assets for real-time rendering engines such as Unity 3D and Unreal Engine 4, and then to research and gather modern approaches and methods to one concise package that can be used as a reference point. An introduction to the fundamental elements of 3D modeling, such as terminology and different stages of creating models, will be explained in the second chapter. The fundamentals of the software that are used will be explained to give an understanding of the working environment. This knowledge is important in order to understand what is actually happening in the modeling software. Later on, the actual creation of the game-ready asset will begin and the process will be documented and explained from the concept stage to the finished product. The process of the character creation follows the industry-standard workflow as closely as possible, with some differences due to the non-industry standard software. The non-industry standard software is not a problem in this case, as most of the procedures can be carried out in other software as well, with some slight differences. The goal is to give the reader a good understanding of the general pipeline that is used among 3D modelers. Naturally there is some variance to the pipelines depending on the studio, but all things considered they can be simplified into a single pipeline. Even though the steps are used to create various character models in the end, the methods can be used to create other types of models, as well.

The thesis is carried out because the methods of 3D modeling are constantly changing and evolving, so finding relevant documentation can be challenging. The problem is not a lack of information, but rather the overabundance of it. If the modeler does not know precisely what to look for in the Internet, it can be challenging to find the latest information among the outdated articles. The methods used in the early 2000s are vastly different to those of today.

As the game industry is becoming more competitive and populated, the 3D modeling methods have to be streamlined in order to stay relevant and efficient in the field. Additionally, with the emergence of hyperrealistic models, customers are expecting to see models of similar quality, and models that used to be realistic 10 years ago might already look jarring.

The practical part of the thesis uses the information that was gathered from the research to create a character model for a VR project by Turku Game Lab. Two software were used for the practical work. Blender was used for modeling purposes, and Substance Painter for texturing the models. Other options were also explored during the project, but in the end these software proved to be most efficient for the project, as a majority of Turku Game Lab uses them as well. During the project, realistic character models were created in a short timeframe of 2 months, and the modern methods were utilized as best as possible. A comparison is then made between the new and the old models to see if the new approach if suitable for this kind of work. After the comparison, the results of the study are evaluated by inspecting the models visually and also by comparing the timeframes of these different approaches. A conclusion is then drawn if it is worthwhile to invest time in learning the new methods.

2 BASIC THEORY OF 3D MODELING

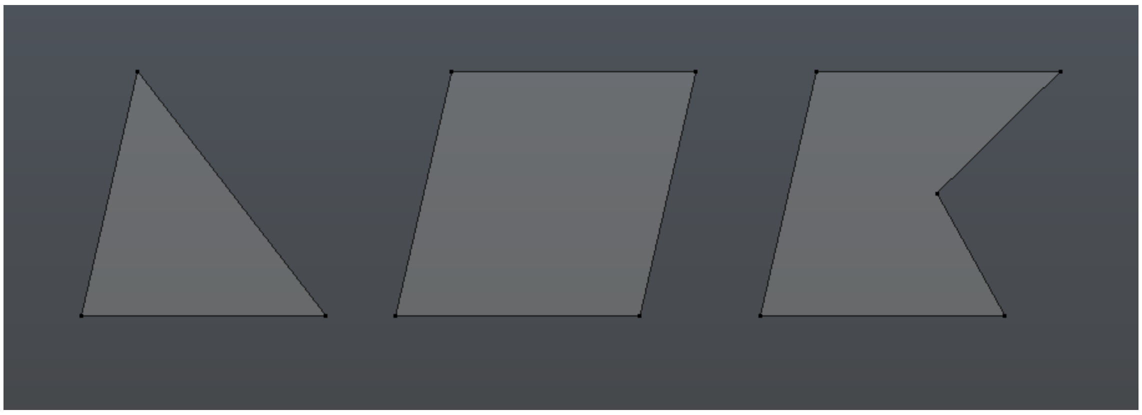
3D models are used to represent objects in virtual space. Essentially 3D models are mathematical representations of imaginary or non-imaginary objects that are three-dimensional. These days, 3D models are used practically everywhere, in architecture, films, games, simulations, and the list goes on. Most modern games, regardless of platform, rely on 3D models of characters and environments. 3D models are used as a final product in games, but they are also great for visualizing early concepts or selling an idea, as 3D gives more information about the idea than 2D illustrations. With the emergence of modern tools and methods, concepting in 3D is becoming more common and a vital skill to have. (Rouse 2018).

A 3D model is basically a collection of vertices that are connected to each other. Vertex is a data structure that represents the position of a point in 3D space. Using 3D software, the positions of these vertex points can be manipulated to create 3D models. Manipulating these vertices on virtual axes is the most common way of creating polygonal models. So, in its most basic form, 3D modeling means moving vertices around in 3D space and creating virtual representations of objects this way. A typical model is made up of a wireframe that is formed with vertices. (Slick 2018).

2.1 Modeling and terminology

In order to understand the chapters below, it is important to know the basic concepts of 3D modeling. Firstly, this chapter introduces the components that represent the surface of a 3D model. In polygonal modeling, the surface of a model is made from triangular polygons, quad polygons and/or N-Gon polygons; Triangular polygon(tri, triangle) is a three-sided polygon that has 3 vertices with 3 edges connected to them. Quad polygon(quad, quadrilateral) is a four-sided polygon that has 4 vertices with 4 edges connected to them. Quads are generally considered the most optimal type for modeling for their compatibility with engines and predictable behavior with modifiers. N-Gon polygon(N-Gon) is a polygon with more than four vertices. It is the most challenging of the three, as it is not supported by some software and can behave unexpectedly in game engines. The clear difference between these polygons can be seen in Picture 1.

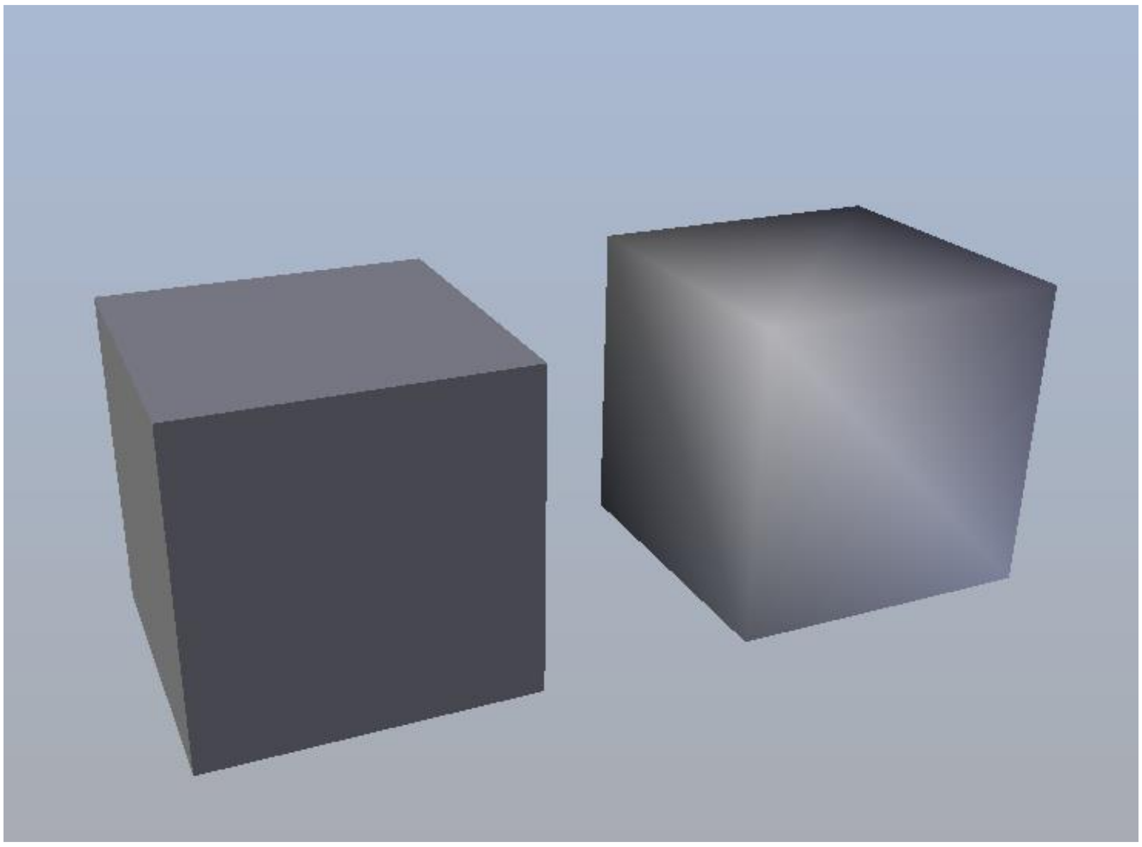
(Turbosquad 2017)



Picture 1. The differences between tris (left), quads (middle), and n-gons (right).

2.1.1 Shading

The shading of the object’s vertices can be set to hard and smooth. With hard shading, the transition between each polygon can be seen clearly as hard edges. Smooth shading averages out the transition between the polygons and by doing so, it makes the model look smoother. Generally, smooth shading is the desirable option for 3D models, but it can cause issues with hard edge transitions because the edge shading is smoothed out. This can clearly be seen by applying smooth shading to a cube, as shown in Picture 2. There are two options to fix the issue; using a normal map, a type of texture map that creates an illusion of surface detail on the model, or sharpening the shading of the edges manually, by adjusting vertex normals. This means that some edges are marked as sharp, while curved surfaces retain the smooth shading. (Polycount 2018).


Picture 2. Default flat and smooth shaded cubes.

2.1.2 Modeling methods

There are severaldifferent approaches to 3D modeling. Usually, modeling methods can be split into categories like polygonal modeling, curve modeling, and sculpting. Some methods are better geared towards different goals. For example, sculpting is traditionally used for organic modeling. Organic modeling is a term that is used for 3D models of characters, creatures, clothing and so on. Using the sculpting approach for organic assets is useful as it gives the object more of a natural look without the modeller having to worry about topology or edge control. Box modeling and poly modeling is suitable for mechanical modeling and for objects with hard edges and surfaces. Box modeling means that the model is first made from basic primitives, and then the desired shape is sculpted out. This can be achieved by adding more vertices to the object and then manipulating these to a desirable position. The vertices are usually added by inserting edge loops around the object. Polygon modeling is often coupled with modifier modeling, which means using the built-in modifiers in 3D modeling software. There is a great variety of modifiers, perhaps the most used modifier being the subdivision surface modifier. The subdivision modifier smooths out the mesh by subdividing the polygons, and typically one level of subdivision increases the vertex count by 4 times of the original count. (Polycount 2018; Taylor 2016).

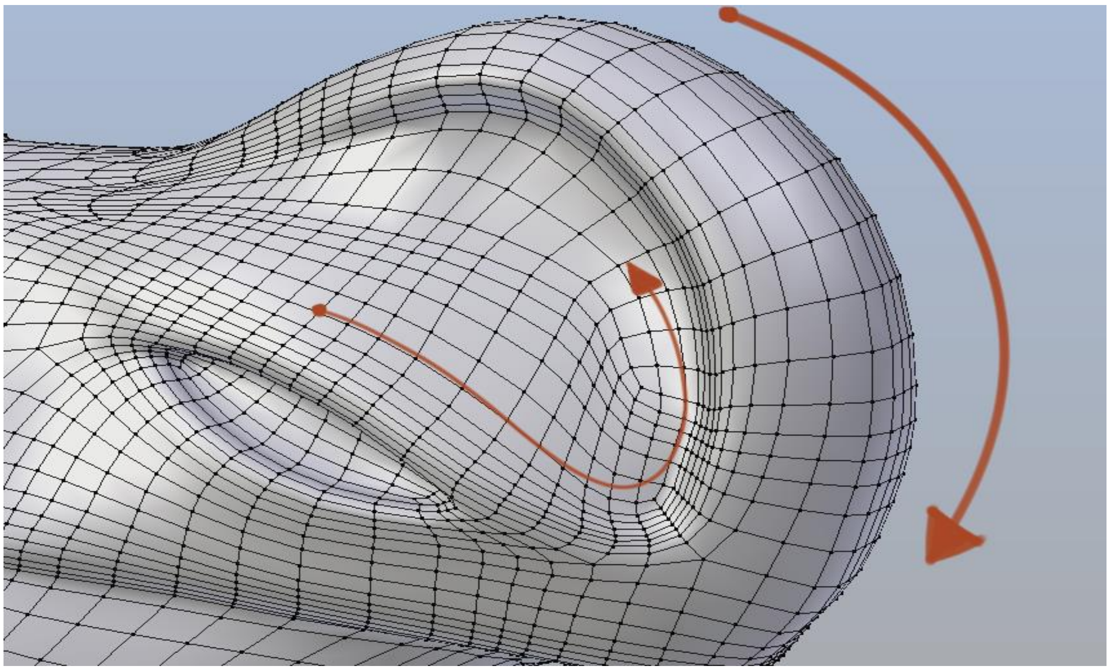
Sculpting has quickly become the main tool for many modelers in the industry. It resembles the workflow of a traditional sculptor, meaning that the modeler manipulates the mesh as they would with actual, physical clay. When a modeler applies a brush stroke on an object, the vertices that are influenced are moved in a direction that depends on the brush being used. By default, additive brushes like clay strip and clay add more volume to the model, while subtractive brushes like crease do the opposite. Regardless of their type, brushes can usually be reversed, so instead of adding volume to the object, it is subtracted and vice versa. Sculpting is best done with a digital tablet, as it gives great control over the stroke sensitivity. Tablets have pressure sensitivity in regards to brush size and stroke strength. The harder the pen is pressed on the tablet, the more it influences the 3D object. The pressure sensitivity can be further adjusted with options in digital sculpting software. For example, Blender has a curve option that can be used to control the strength falloff of the brush. (Sosak 2018).

A rather new invention in the realm of digital sculpting is adaptive sculpting. This feature is called dynamic tessellation in Zbrush 2018 and Mudbox, while in Blender it is called Dynamic Topology(Dyntopo). The main principle stays the same regardless of the software; instead of only/simply stretching and moving pre-existing vertices, dynamic tessellation adds more resolution(vertices) to the model with brush strokes. The amount of resolution that is going to be added can always be adjusted in the options. The reason why this feature is a game changer is that previously the whole model would have to be subdivided in order to add finer detail. This adds resolution everywhere, even in places where it is not needed, as a result making the performance of the model worse. This problem is fixed with adaptive sculpting as the placement of additional vertices can be controlled easily. (Adams 2018).

The main advantage of sculpting is its speed, which is precisely the reason it is used widely in the game and film industry. Sculpting is very quick to create concepts and see how they work in 3D instead of relying on 2D illustrations. Another advantage is that sculpting can be used at any stage of modeling. It can be used to create a base model from scratch or to add detail to an otherwise ready model. Sculpting has its own flaws as well; if the modeler is not careful, the vertex count can increase quickly, making the viewport of 3D modeling software laggy. Additionally, precise control is difficult to achieve by sculpting. So mechanical objects that require high precision are better modeled with traditional modeling techniques. The performance of high poly models depends greatly on the software. Zbrush is considered to be the best software when it comes to performance of high resolution models. Sculptris can also handle high amounts of polygons efficiently and is free, but lacks some features that are found in Zbrush. 3D coat is also a good option for sculpting, as it is more affordable than Zbrush and can also be used to create texture maps if needed. Polygonal sculpting software like Mudbox and Blender’s sculpting tools are somewhat limited when it comes to highly detailed models, as they can not handle heavy models as well as Zbrush, 3D Coat or Sculptris. (Taylor 2016).

2.2 Topology

Topology is basically the layout of a 3D model, meaning how the vertices and edges are placed to create the surface and wireframe of a model. A good topology means that the vertices are organized in a way that they are clean, as efficient as possible when it comes to performance and also retain the details of the surface. Most models have a clear flow and sense of direction in the wireframe, which is commonly known as edge flow. (Lifewire 2018) This edge flow can clearly be seen in the wireframe of the model in Picture 3.



Picture 3. The direction of the edge flow.

If a model lacks a good edge flow, it can look like it is made of play-dough and there is no sense of volume or detail. The advantages of knowing how to achieve a good topology are tremendous, and as such, it is a crucial skill to have as a 3D modeler. A good topology makes working with the model easier, minimizes strange artifacts and pinching, the modifiers such as subdivision work more smoothly, and above all, a good topology makes the model more optimized so it requires less performance and memory. Topology is especially important in real-time rendering, with models that have deforming animations, like character models, and with models that have highly reflective surfaces, for example, cars. (Thilakanathan studios 2016).

The topology of models that are created for games should be optimized to retain their silhouettes as well as possible. Surfaces with high curvature tend to require more polygons to support the structure. Edge loops that do not affect the silhouette are the first to be deleted if a model has to be optimized further. Of course, in some cases, this can not be done if the edge loops are there to support the deformation of the model. Additional edge loops are commonly used in places that have a high degree of deformation when animated. For example, an area that requires an excellent topology and a high number of edge loops is the face of a character model. If the topology is not done well, the facial animations look off-putting and unnatural. (Thilakanathan studios 2016).

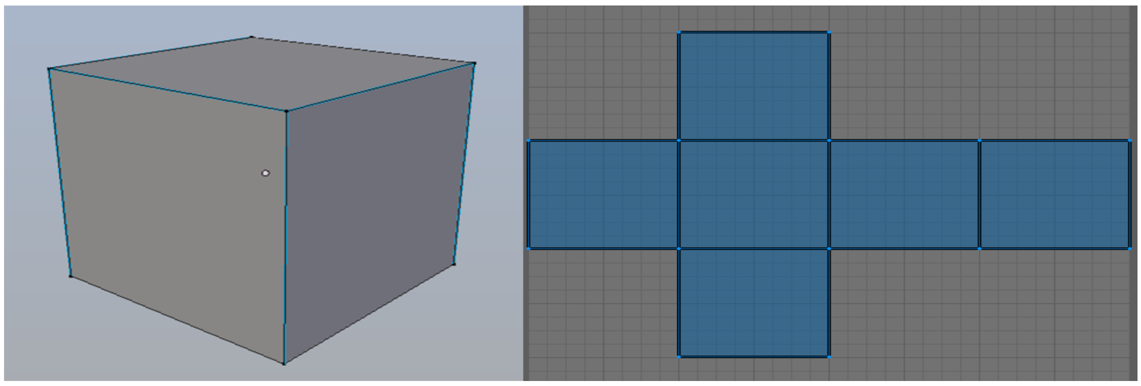
A model with the most optimal polygonal structure for games is made entirely from quads. Quads are the most reliable polygons, as they work well with modifiers, are easily editable and smoothing is predictable. However, sometimes the amount of effort it takes to model everything out of quads can be too much, and the problem could be solved simply by using triangles in some places. In fact, triangles are the easiest solution to a topological problem, and if they are placed in areas that do not deform or cause issues in shading, it is completely acceptable to use them. (Taylor 2015).

The usage of triangles and n-gons result in vertex structures called poles. They are a crucial part of modeling, and the term pole refers to a single vertex that has 5 or more edges converging to it. Poles cause pinching in their proximity and for this reason should be placed in places where the topology can accommodate them without any pinching. If a pole is changing the curvature of an object, it should be removed or moved to another place on the surface. Poles are less visible on flat areas and on areas with only slight curvatures and should be placed in places like those. Poles with more than 6 edges should be avoided and their usage is generally considered to be bad practice. (Topology Guides 2018).

A crucial step of 3D modeling related to topology is retopologizing, also known as retopo. Retopology is done to change the topology of the 3D model, with the goal of reducing the total amount of polygons. The targeted final polygon count has a huge variance. For example, a console game, Infamous Second Son, had 120 000 polygons per main character, while another console game called Killzone Shadow Fall had 40 000 polygons per character model. Naturally, models in mobile games can not afford to have as many polygons as console and PC games. (Polycount 2014).

2.3 UV Mapping

UV Mapping is the 3D modeling process of projecting a 3D model onto a 2D plane. Visualizing a 3D object that is flattened to a 2D representation can be difficult, and to grasp the concept clearly it is easiest to unwrap a simple cube as shown in Picture 4.


Picture 4. Unwrapped cube.

This unwrapping makes it possible to use image textures. When the model is projected onto a two-dimensional plane, each polygon, edge and vertex are transferred on the UV map. A texture is then placed on the UV map, based on the unwrapped polygons. Game engines use two texture coordinates for mapping the width and height of a texture; U is used for width, V is used for height. U and V are used to avoid confusion because x and y are already used to represent the 3D space. Texture coordinates are based on a grid in a scale of 0.0 to 1.0, 0.5 being the middle point. When a model has a UV scale distance that is greater than 1, the texture is then tiled across the model. UV maps consist of UV shells or UV islands, both have the same meaning. When the model is unwrapped, it is usually broken up in several pieces, in UV islands. (Polycount 2018).

There are many methods to unwrap a model on a UV map. For example, the active polygons, edges, and vertices can be projected on a UV map based on the view of the viewport. This method is called planar mapping and works best for objects that are relatively flat. Round objects tend to have distorted UV maps because the less perpendicular the surface is relative to the camera, the less of it is seen in the UV map. If the whole model is unwrapped using the planar method, even the polygons that are not visible to the camera will be unwrapped, causing overlapping UV shells. Perfectly overlapped UV shells can be used for symmetric models to save UV space, but accidental UV shell overlaps tend to cause problems in the texturing phase. If only a portion of the polygons is selected and then unwrapped with planar mapping, a new UV island is created with these polygons. (Autodesk 2018).

Another method is to mark UV seams manually in 3D software. A mesh is unwrapped at the seams, similar to how an orange would be peeled. The cuts in the peel are same as seams in meshes. Manual seam placement is preferred for complex models, as it gives more control of the unwrapping process. With this method, it is best to keep an eye on the amount of stretching in the UV map. Generally, the more seams there are, the less stretching there is. Less stretching is great, but a large number of seams can cause considerable issues during texturing, as seams can be easily seen with textures. Seams are best to place in places that cannot be seen in-game. In addition to these manual methods, there are some automatic options available as well for UV mapping. These automatic unwrapping functionalities can be found inside 3D modeling software, for example, Blender and Maya have this capability. There are also dedicated software for the job; for example Headus UVLayout and Unfold3d. Usually, automatic unwrapping requires some manual tweaking. This is especially the case with the automatic unwrapping in Blender, as it usually splits the UVs into too many islands(Polycount 2014). After the whole model is unwrapped, the scale of UV islands should be checked by applying a checkered texture on the model. This texture reveals potential stretching and unwanted seams. Even if the goal is to have similarly scaled UV islands, important meshes tend to use more UV space, as they are the main focal point and require more detail. The UV islands should be packed as tightly as possible, to maximize UV space, and symmetrical UV maps can be placed on top of each other if there is no need for unique texture detail. Too tight of a packing can cause texture bleeding, which means that the color of a UV island bleeds into another one because they are too close to each other. This is fixed by using a higher value for the UV island margin. (Blender Docs 2018).

.

2.4 Texturing

Adding surface detailing to 3D models is done with textures. Textures are raster images that are projected on top of the UV map of the object. The size of textures varies with the project, but as a rule of thumb, the resolution of the texture is kept to a power-of-two value, as it is more efficient performance-wise. Without the textures, the surface detail would have to be done with geometry, by adding more vertices, which is very performance-heavy and in many cases, impossible to run in real-time engines. This is why textures are used, as they are relatively lightweight. (Pluralsight 2014).

The performance of the textures is of great importance to real-time rendering. It is good practice to adhere to certain industry standards when it comes to texture sizes. Texture size, better known as texture resolution, is commonly kept to a value the of power of two. Power of two means that the width and height of the texture have to be divisible by 8 and/or 2, and they can be multiplied up or divided down by 2. So, the resolutions being used are often values such as 512x512, 1024x1024, 2048x2048, 4096x4096 and so on. The numbers represent the number of pixels that the textures has, the first value being the x-axis and the second the y-axis. This means that the resolutions mentioned above form an image that is shaped like a square. Modern games prefer square-shaped textures with a 1:1 ratio for width and height. Naturally, the higher the resolution of a texture, the more performance heavy it gets. Every texture that is below 1024x1024 pixels can be considered a low-resolution texture, and the textures above that resolution are called high-resolution textures. The “Power of Two” rule applies to all types of games, whether they are PC games, console games or mobile games does not really matter. Using incorrect texture sizes does not break the game per se, but it is not optimal for the performance. This happens due to the way the game engines generally work. Non-power-of-two textures(NPOT) could not be used with earlier hardware, but modern graphics cards(GPUs) can handle NPOTs, however with reduced performance. In case of an imaginary non-power of two texture size like, for example, 480x400 pixels, the engine either scales it up or down. The direction of the scaling depends on the programming of the engine. Both options are bad for performance. If a 480x400px texture is scaled up, the size goes up to 512x512 pixels. By resizing the texture, the engine basically fills in the blanks with extrapolated information from the texture. This often results in a blurry, less detailed texture, that might not even work properly due to artifacts. (KatsBits 2018).

2.4.1 Correct texture sizing

A common question among modelers is what texture resolutions should be used for different assets. Generally, the texture artist has to consider how large the object is on the screen, how close it is to the camera, and how much memory can be used for the textures. If the object is close to the camera, for example, weapon models in first-person shooter games, the texture sizes can be relatively big.

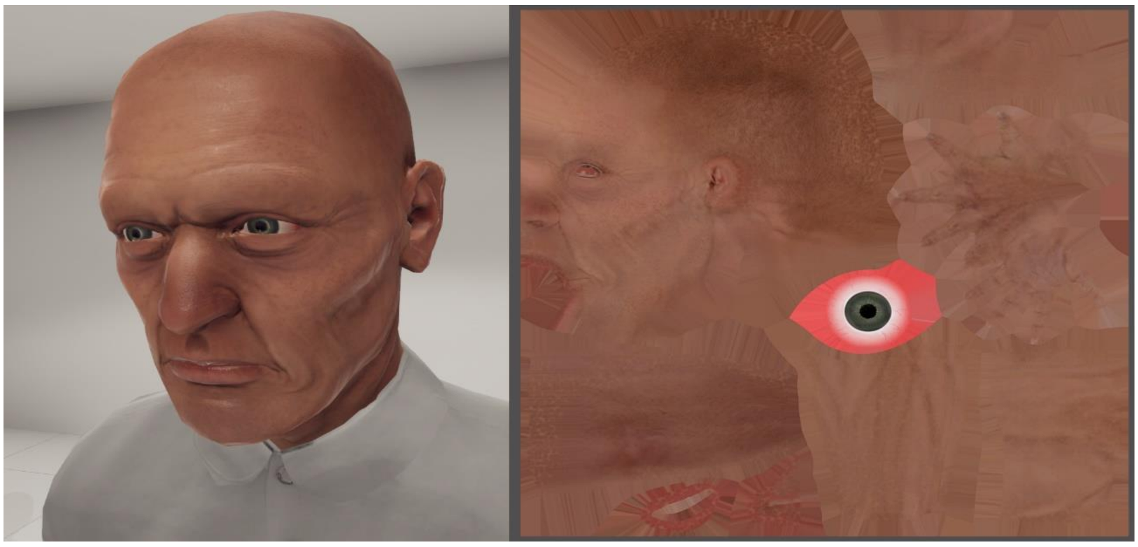
Texel density is an important aspect to consider when deciding on the size of the textures. A texel is each pixel of a texture during the time when the game engine processes the texture. The game engine works in three steps. First, it performs calculations to project the texture onto meshes. Then, the texture pixels are transformed into texels. Lastly, the engine transforms the texels into screen pixels. Texel density is the amount of texture resolution on a mesh. A good practice for deciding texel density is to first consider which type of game the asset is going to be created for. In third person games, such as Mass Effect and Witcher, since the assets are not that close to the camera, a texel density of 5.12px/cm will suffice. In games where the camera is very far from the assets, the texel density does not have to be as high. A texel density of 1,28 or 2,56px/cm is enough in these cases. The type of games requiring the most substantial texel density are First Person Games. The camera is very close to the player model, and therefore the assets require a lot detail. This quickly bumps up the need for texel density, up to 10,24px/cm. In general, the way the model is unwrapped, the resolution of the texture, the scale of the model all affect the way the texture resolution looks in-game. It is important to keep a similar texture size throughout the whole scene, otherwise some models look like they do not fit in at all and the end result will be jarring to the player. (Lezzi 2016).

2.4.2 PBR textures

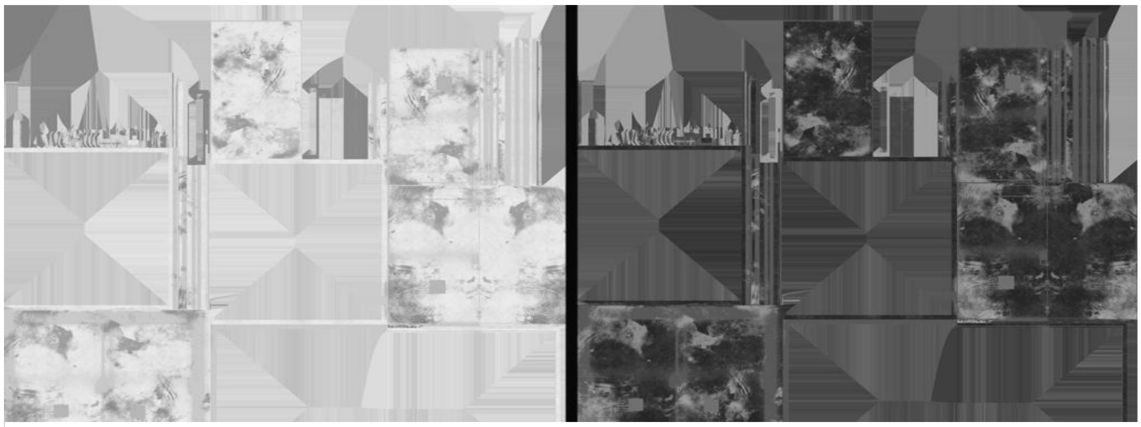
PBR(Physically-Based Rendering) is a modern way to have more accurate and realistic materials and shading in real-time rendering engines. This can be achieved by utilizing physically accurate formulas, and most importantly, by following the laws of energy conservation. Because of this, PBR is very dynamic and flexible and can be used in any lighting condition in most cases. PBR emulates how a light source interacts with the two primary material types, metals and non-metals, and how much of the light is diffused(diffusion) or reflected(specularity). Diffusion and reflectivity are the two ways for light and surface to interact with each other. If light rays hit a surface that is hard and glossy, a majority of the light will be reflected. If the surface is soft and non-shiny, the light is diffused on the surface, making it more lit without that many reflections. Another category is for semi-transparent materials, like skin and wax, where the light is actually diffused inside the object, creating a new color output. This effect is called subsurface scattering. (Allegorithmic 2016; Burke 2015).

Because of the major differences of metals and non-metals(conductors and insulators), it is helpful to think strictly in terms of metal or non-metal when creating PBR materials in the beginning. If the surface is metallic, a set of rules have to be followed, as is the case with non-metals. These rules are not usually complicated and in practice it just means that when the material is being created, certain values have to be plugged in correct channels. This can be achieved for example by looking up the correct values for albedo, the base color of the object, metallic and roughness channels in the guidelines of real-time rendering engines. This way of thinking can be too simplistic in some cases as some materials do not fall into these two categories, for example metalloids, but overall it is a good starting point. (Allegorithmic 2018).

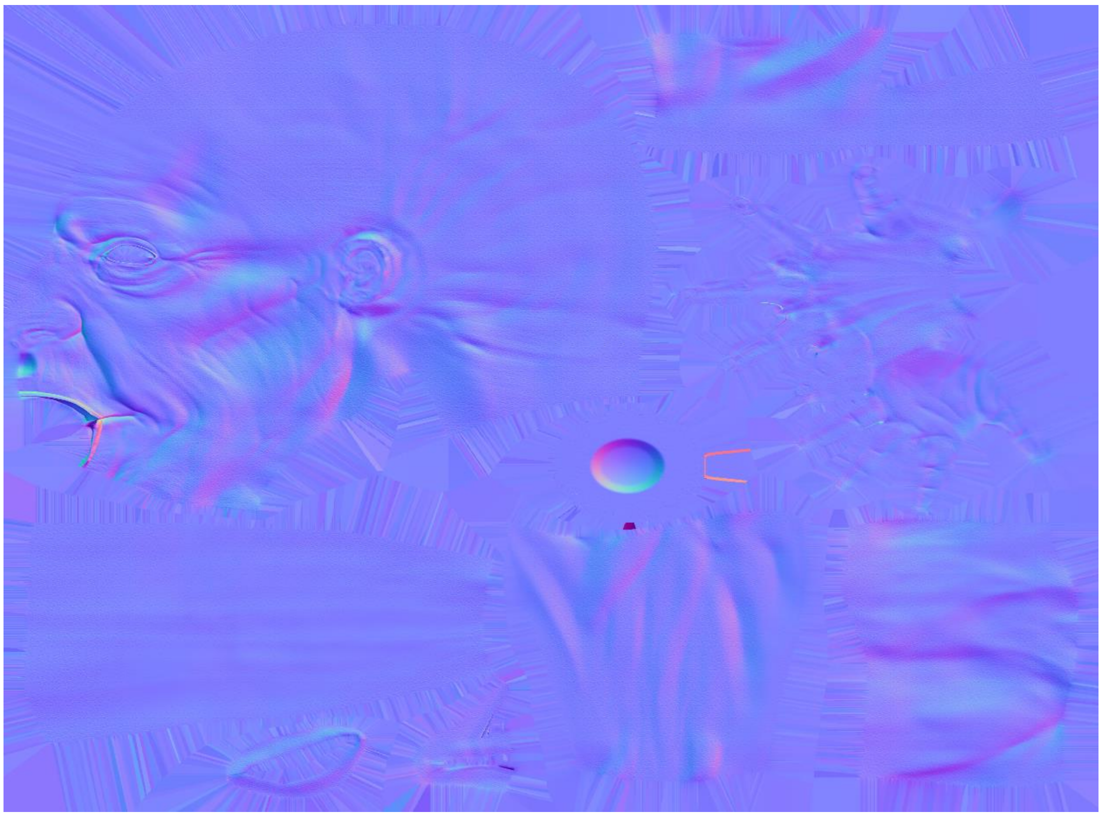
To influence the base color of an object, there are two options to choose from; a diffuse texture map or an albedo texture map. In Picture 5, the color map contains the base color of the character’s skin. A common misconception is that they can be used interchangeably. However, a great difference between using a traditional diffuse map or a PBR albedo map is the removal of ambient occlusion and directional light in albedo maps. Ambient occlusion is an effect that approximates the amount of occlusion. This means that there should be no highlights or shadows in albedo maps, which results in more accurate texturing that works more dynamically in multiple lighting scenarios. (Wilson 2015).


Picture 5. The base color of the face is created with an albedo map.

The microsurface detail, in other words, surface roughness, is commonly controlled by using roughness or glossiness maps, depending on the workflow that is being used. Both of them are grayscale maps, with values ranging between 0 and 1. As the name suggests, the roughness map dictates how rough the surface is; the rougher the surface, the less reflective it is. White areas in the map indicate a rough surface, while darker tones indicate smoother and shinier surfaces. Glossiness is practically the same as roughness, but it is only inverted, white values being glossy and black values non-glossy. The difference between the glossiness and roughness maps can be seen in Picture 6. (Maximov 2014).

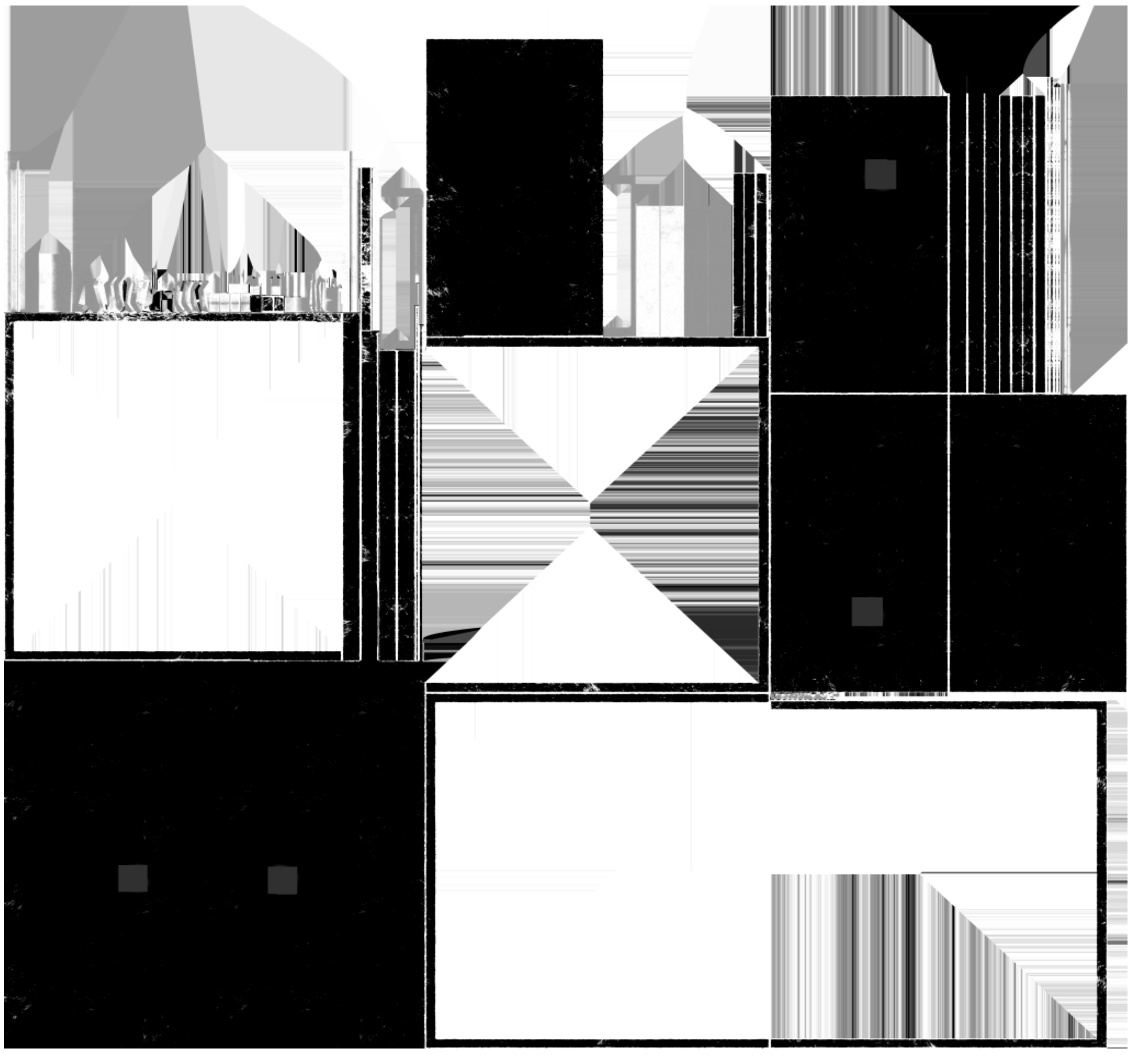

Picture 6. Differences between a glossiness (left) and a roughness map (right).

A normal map is an image that manipulates the direction at each pixel. This direction is called a normal. Normal maps are basically used to fake details on a mesh by manipulating the normals. A normal map uses RGB channels to change the normals, each channel corresponding with x,y and z-axis. In other words, different colors in normal maps mimic bumps and cavities. In Picture 7, wrinkles and creases of a character's face are faked by using a normal map. As normal maps are very difficult to create manually, they are often created by transferring details from a highly detailed model to a model with a lower amount of polygons by baking. Baking is the term that is used for saving detail on a texture map that is then transferred onto a model. Bump maps were used earlier to achieve a similar effect, but now normal maps are seen as a more modern and superior


Picture 5. Wrinkles are created by using a normal map

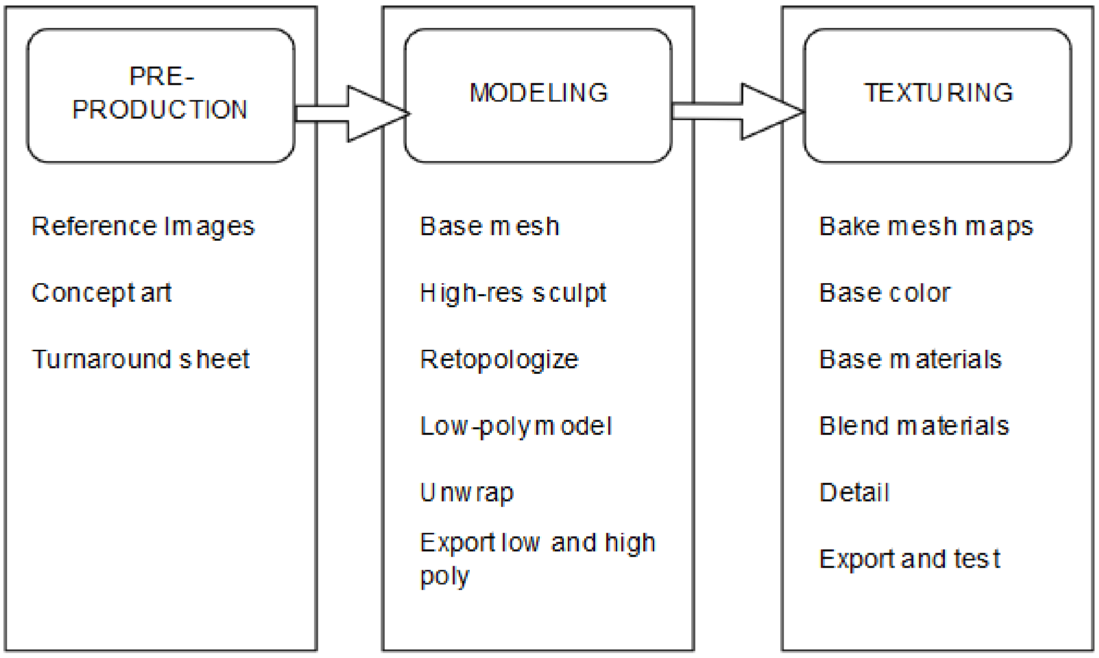
version to bump maps. (Autodesk 2018; Pluralsight 2014).

The last essential property for Roughness/Metallic workflow is metallic. A metallic map is used like a mask to differentiate metals and non-metals. A white value 1 is used to represent a metal and a black value of 0 is used for non-metals. This can be seen in Picture 8; a metal map is used to texture a metallic door. Black parts are painted metal, meaning they are dielectric, and white parts are bare metal. To achieve a realistic result, large gray values should be avoided, as they are not physically accurate. However, they can be used to smooth out transitions between metal and non-metal. (Allegorithmic 2018).


Picture 6. A Metallic map separates bare metal and painted metal from each other.

3 WORKFLOW OF 3D MODEL CREATION

The precise workflows and pipelines in the gaming industry are difficult to describe accurately because many studios have their own ways of creating 3D models and due to Non-Disclosure Agreements. However, there is a general workflow that studios follow in the production of 3D models, albeit with some differences, as shown in Picture 9. Naturally, this workflow varies depending on the size of the studio. Bigger, so-called AAA studios tend to have artists that are dedicated to a very specific part of the pipeline, while smaller indie studios employ artists who work in many stages during the production of 3D assets. There are some optimization differences between mobile, console and PC gaming pipelines, but the general workflow stays the same.



Picture 7. A general pipeline in 3D asset production

3.1 Pre-production and concepting

Generally, the pipeline starts with pre-production and concept art. Concept art stage is where the artists come up with ideas and designs together with the art director. During the early stages, the concept artists do a lot of research and exploration in order to find an interesting design that fits the theme of the game. This work is usually done digitally in Photoshop, as it is the industry standard software. The idea of the concept art stage is to generate as many ideas as possible. The goal is not to create highly rendered images. Once an appealing design is found among the other ideas, it is refined further and additional concepts are created from different angles. This eases the job of the modelers significantly. Concept art is not always purely made in 2D, as 3D software is frequently used to help with drawing the environments. First, the scene is roughly blocked in using a 3D software of choice. The software does not really matter at this point because this process is fairly simple and can be done with a minimal knowledge of 3D modeling. After the basic scene is blocked out, the artists paint over the 3D version, and by doing so the perspective stays correct. (Moore 2014).

A technique for 2D concept artists called Photo bashing has become common for generating fast environmental concepts. It is a technique that utilizes photographs and 3D models by merging them together in a 2D graphics software. Photo bashing can speed up the concept work considerably, and make concepts more realistic. It is not a substitute for artistic skill, and artists who use photo bashing and work in the concept art industry are still proficient artists with or without using photographs. (Heginbotham 2018).

In some cases the 2D phase of concept art is skipped completely and the artists jump straight into 3D concepting. This can be done because the sculpting software is getting more advanced and easier to use. Zbrush is used mainly for this task, as it is seen as the most advanced sculpting software on the market, and is the current industry standard software for sculpting. The advantages of concepting in 3D are huge. Working in 3D ensures that the model will work in-game as well. It is faster to generate ideas, as the lighting and perspective is handled by Zbrush. It is also possible to change the material of the concept sculpt with so-called Matcaps in Zbrush and usually in other sculpting software too, such as 3D Coat and Blender. Matcap stand for material capture. Material capture means that the lighting and reflections are built-in with the material. So, when camera moves around the object in 3D view, the lighting acts as if the object itself is turned. This makes it easier to see planar changes in 3D models. In addition to this, there are matcaps that simulate different materials, such as skin, metal, and clay. This advantage saves a huge amount of time, because rendering is time consuming in 2D concepts, and can essentially be done with a couple of button presses using 3D software. In addition to these advantages, sculpting software have the option to turn on symmetry, meaning that the changes made to the concept are automatically applied to both sides if the concept artist chooses so. A good example of utilizing a mix of 2D and 3D software during the concept stage can be found in The Art of Doom Book(2016). Id Software used Zbrush to help with the concepting of the monsters, and 3D software was used to block in compositions and perspective for 2D environment concepts. (BlenderArtists 2011; Pavlovich 2015; Youtube 2017).

3.2 Modeling and detailing

After the early concept phase is done, the concepts are then passed on to the 3D modelers. 3D modeling phase usually starts by blocking in the base model for the asset. Base mesh is a low-resolution model that will be refined and detailed later on. In some cases, the modelers use pre-existing base models with some modifications done as this saves time and resources. The block-in phase can be done in most of the 3D software, and there are various different approaches for this. If this step is done traditionally in software like Maya, 3ds Max or Blender, the modeler usually starts with very primitive shapes, such as subdivided cubes, and quickly builds up the basic silhouette. This step does not usually contain any detailing work, as the idea is to create a good base for digital sculpting. It is important to get the proportions right as early on as possible. A good rule of thumb for the base model is to avoid triangles, n-gons and so-called poles, meaning vertices with 5 or more edges connected to them. This convention exists because quads can be subdivided with no stretching or pinching. Triangles and poles can cause pinching when sculpting, and n-gons are generally avoided, because they work unreliably depending on the software, and can cause a lot of issues when deforming the model. In most games, the character will have animations and deformations, and for this reason, it is not recommended to use n-gons for character models. (PluralSight 2014).

If the base model is created in Zbrush, it is common to use a feature called Zspheres. Zspheres are a Zbrush tool that makes it possible to create a base mesh quickly and easily. This tool is especially useful for organic models. Zspheres work by extruding additional spheres from a root sphere. The new, extruded spheres work as new joints and can be rotated. The mesh between the spheres will be generated automatically and the radius of the spheres can be adjusted at any point. (Pixologic Documentation 2018).

When the base mesh is done, the detailing will begin. The aim of this step is to produce a highly detailed, high poly version of the character. The polygon count of the high poly version can get fairly high, but it is not used for real-time rendering so this is not a problem. The base model can be imported to any sculpting software of choice, most popular being Zbrush. When the model is imported in Zbrush, it has to be converted to polymeshes, as only polymeshes can be sculpted on. The main advantage of digital sculpting is the natural feel of organic modeling. For example, clothing and skin wrinkles are easier to create using digital sculpting methods.

The general workflow for sculpting is to move from big details to small details. This philosophy applies to almost any 3D modeling phase as well. It is recommended to move through the mesh in several detailing passes, moving on to smaller and smaller details with every pass. The number of passes depends on the model, more stylized and simplified models can be done with fewer passes than highly realistic models. However in both cases the point of the first passes is to establish major masses and planar changes. In realistic models the final pass usually includes sculpting in micro details. The final pass of detailing is convenient to do with custom alpha brushes. (Flipped Normals 2018).

3.2 Baking in the details

When the high poly model is finished, the next step is to transfer all the surface detail into the base model. Before doing this, the base model should be refitted to the high-poly version. The closer the low poly version is to the high-poly version in terms of major shapes, the easier it is to transfer the detail. This process is called baking. Baking can be done in many general 3D modeling software, but there are dedicated baking software as well, for example Knald and Xnormal. It is also possible to bake the details in Substance Painter. Regardless of the software, the baking process tend to work the same way. The baking process uses two models, the low poly and the high poly meshes. The baking tool starts at a distance out from the model and casts a ray in the direction of the other model. When the ray hits the 2nd model, it records the surface data and uses it to generate a texture map. Before the models are exported for baking, they should be triangulated, as different render engines calculate triangles differently, and if the triangulation does not match up, baking artifacts may occur. (PolyCount 2018).

While the baking itself is a simple process, it is prone to errors and sometimes there are artifacts in the baked texture maps. The artifacts happen mostly because of three possible problems; the low poly mesh does not have enough geometry, ray distance is set incorrectly in baking software or there is overlapping geometry on the model. Insufficient geometry in the low poly model causes waviness to the normal maps.

Generally waviness is not desired, but it can work if the model is only viewed in limited angles. The simplest solution to this problem is to add more geometry to the low poly mesh. Skewed detail is actually the same problem and can be fixed by adding support loops, since the added loops help in averaging the vertex normal. (Polycount 2011).

If there is overlapping geometry in the model, the meshes can be baked one by one by separating them to different models, or by “exploding” the model. The exploded view shifts the meshes in different, non-overlapping positions. The separation process of baking can be bypassed by naming the models with suffixes that differentiate high poly meshes from low poly meshes. The suffix can be named however the modeler wants, but for clarity it is best to name them logically and stick to the naming convention, for example using suffix _lp for low poly mesh and _hp for high poly mesh. By following this method, the modeler can save a lot of time in the long run and increase efficiency. Match-by-name feature can be found currently in Substance Painter. If this feature cannot be found in the modeler’s 3D software, it is best to use the exploded view method. Exploded view functions the same way as it does in technical drawings; the pieces are separated from each other by “exploding” the parts outwards. (Allegorithmic 2018; Polycount 2018).

3.3 Creating texture maps

Baking and texturing processes have overlap somewhat, as it is common to bake in the details of the high poly model in texturing software. The bake data is often used to generate additional detailing during the texture phase. Both texturing and baking can usually be done in the same software, unless the modeler uses a dedicated baking software like Knald. (Cg Cookie 2016).

3.3.1 Advantages of modern texturing methods

One of the biggest changes in the 3D modeling industry is the improvement and streamlining of the texturing process, and the emergence of PBR shaders. When the texturing is done in a modern software like Substance Painter or Substance Designer, it is faster to create iterations and modify textures. The main problem with the old way of creating textures in Photoshop is that if the texture needs to be changed, the changes have to be done in every texture type. This slows down the texturing pipeline significantly, especially since this step can be avoided by utilizing the features of modern texturing software. (Bär 2018) By generating the textures with masks using the surface data from high poly most texture maps can be modified easily without much manual work. This non-destructive way of texturing is a huge feature in a modern pipeline. The layers function similarly to 2d graphics editors, such as Photoshop and the less commonly used, open-source equivalent Gimp.

Another useful feature of Substance Designer and Substance Painter is that massive, pre-existing material libraries can be found on the Internet. These materials can be mixed together to create fast iterations and build new libraries for the project. One platform for such services is Substance Share that can be accessed by having an Allegoritmic account. There are currently 1013 materials and 529 smart materials in the library. Premade materials are a great way to create a base for the textures that will be tweaked later. However it is not recommended to rely only on premade materials, as many require some manual work to really look realistic. (Substance Share 2018).

3.3.2 Texturing stages

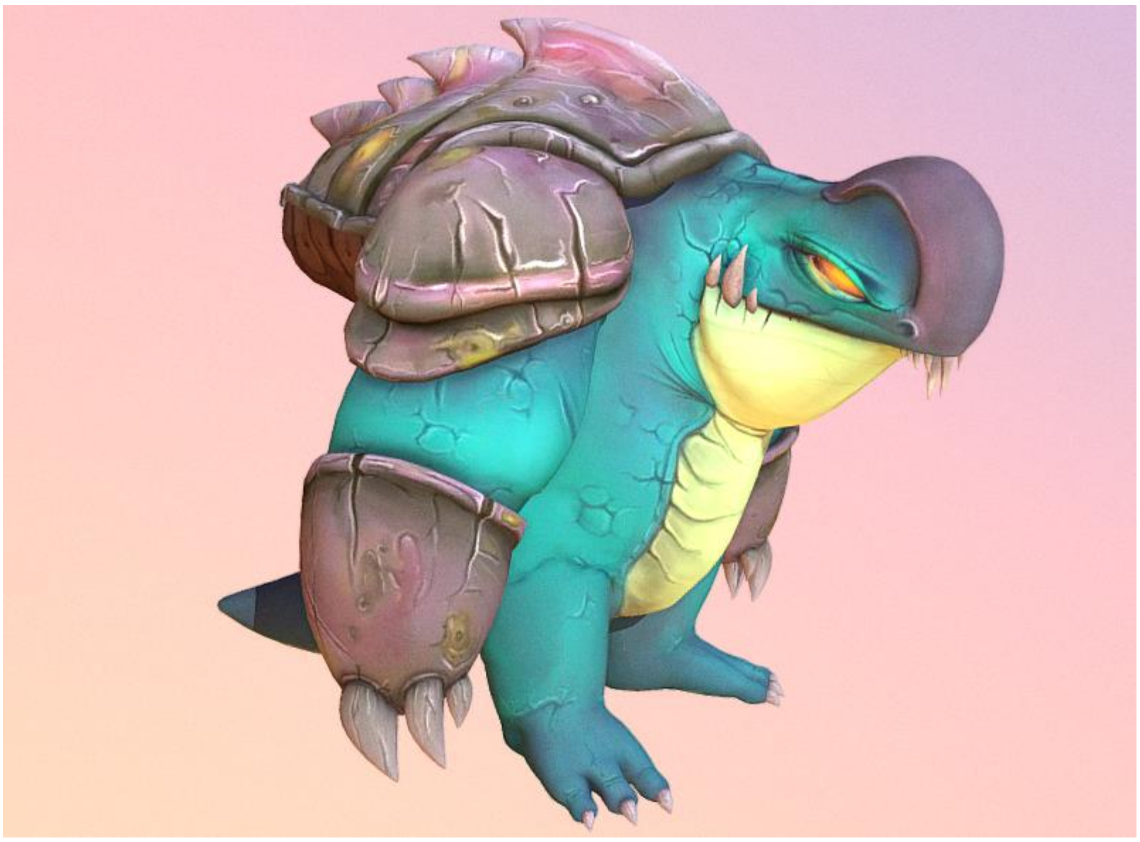
The process of character texturing begins by referencing the concept art and laying down base colors. As always, it is important to gather relevant reference pictures to make the process easier. Different parts of a model are usually separated by using black masks, and then are simply filled with flat colors to give a sense of the color palette. Many texturing software can utilize ID masks that are done in 3D modeling software by changing vertex colors. At this point it is still possible to explore different color palettes, as it can be done by simply changing the color of fill layers. To ease the process of coming up with a color palette, it is important to get familiar with at least the basics of color theory. As it is a rather extensive area, it will not be covered in this thesis. After the base colors are established, it is time to add physical properties to the model. This means adjusting metallic and roughness values to a roughly accurate number, depending on the material. At the start, metallic value for metals should be set to 1 and non-metals to 0. With just these basic fill layers, the block-in is done. After block-in, layering of different materials begins. By layering materials it is easy to make variation to the base material, making it look more realistic. Lastly, after the model has some color and roughness variation, it is time to add finer detail. These details can be added manually by painting them with custom brushes or by using the bake data and fill layers. Bake data like curvature maps are especially great for generating dirt or wear to the model. A great feature of texturing software is the stamp brush which can be used to literally stamp textures on the model. The stamp function is useful for adding unique detail and finishing touches to a model. At this point, some sort of testing scene should be created to check if the materials work in the intended real-time rendering engine. The lighting of the scene should resemble the one used in the game environment. Usually it is recommended to create at least a bright and a dimly lit scene for testing, as both are quite common in game environments. The steps of an ideal material pipeline with modern texturing software should look like follows; Identify material types, create or load material presets, blend the materials together, apply wear and tear, and lastly, apply decals and unique details if needed. (Johns 2016).

After all of the steps mentioned above, the model should be ready to handle forward to a rigger and then to an animator. As this thesis only cover the creation of the model itself, rigging and animating will not be discussed further.

3.4 Realism vs. stylization

The goal of realistic texturing is naturally to make the assets as lifelike as possible, and at the same time to enhance reality, to depict it in a way that is interesting to the player. Realism does not necessarily mean that the game environment has to look exactly like the real world, just that it would be plausible to exist in an imaginary world. Realistic textures rely heavily on PBR workflow, procedural texturing and photograph based texturing. The disadvantage of realistic texturing is that, while it’s bringing a cinematic feel to the game, it tends to age poorly. Games that were considered photorealistic 10 years ago have the infamous uncanny look. This same phenomenon has happened many times, and it is most likely that the current generation’s hyper realistic look is no exception. A great advantage to realistic texturing is that the software are improving every year, making it easier to produce believable textures with simple equipment.

Stylized texturing is the artists depiction of simplified reality. Textures in this style tend to have fewer details and exaggerate shapes that are characteristic to the asset that is created. Microsurface details are often toned down while focusing on small detail, medium detail and big detail, bigger details being more important. Microsurface details refer to surface noise like skin pores, tiny grains in fabrics or directional fibres in wood. Stylized textures have huge variance; some styles use only diffuse color textures, while others utilize the modern PBR shaders. The biggest variation tend to come from the way how the diffuse textures are done. The artist might opt for simple flat colors, gradients or for the most labour intensive, painterly look. The advantage of using only diffuse color is that it is possible to create very appealing texturing with minimal performance drops, making this method suitable for all platforms. For example in Picture 9, all of the details of the heavily stylized model are done with a diffuse map. It is possible to fake lighting information and irregularities on the surface by hand-painting them. Disadvantage being that handpainted textures tend to take a long time to create. (Aava 2017).



Picture 8. A stylized model using only a hand-painted diffuse map

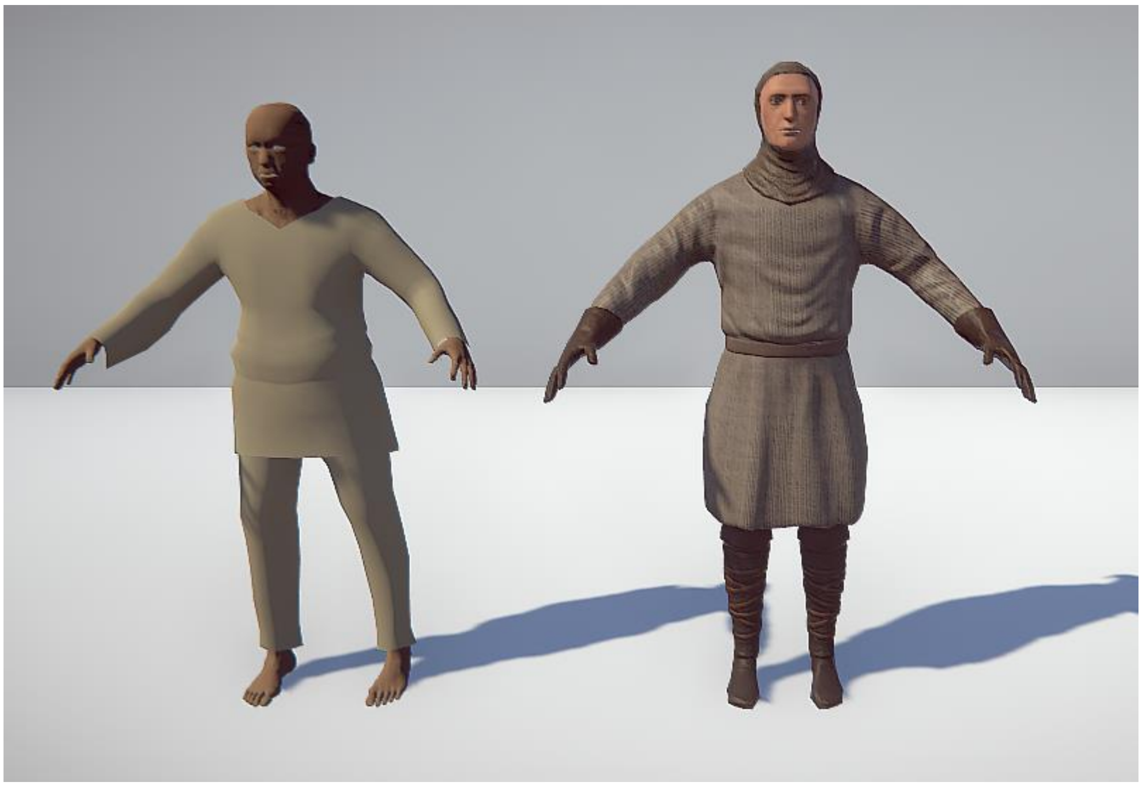
4 COMPARING MODELS WITH DIFFERENT WORKFLOWS

In this chapter some models will be compared to each other that were done as a part of traineeship by either the author of the thesis or other interns. The character models were created for projects called Lights on! and Virpa. The methods used for the models had quite a lot of variance, depending on the project, and the results could be seen in the models.

4.1 Lights on!

Lights on! is a mobile game that uses AR technology to unlock game content. For this project, the customer requested a lot of character models for the game. The characters, their clothing and accessories needed to be historically correct, as the game was loosely based on real life events. It was requested that the models should look realistic, so creating stylized low-poly assets was not an option. A contact person sent various reference pictures with the goal of creating an accurate representation of the people during the game's time. Usually reference pictures are taken from various angles, giving the modeler a sense of the overall model that is to be created. However, because it is difficult to find accurate pictures of this particular subject, some models had to be redone for historical accuracy via some rounds of feedback, which caused slight delays in the pipeline.

As the modelers had to create almost 100 models in a very small time frame, there was not enough time to go through proper modeling workflow, and crucial steps for realistic models had to be skipped. The most crucial steps being the creation of a detailed high polygonal mesh for baking purposes and lack of realistic textures. Instead the models were created by using only low-poly models, on top of a base human mesh. The base meshes were created in HumanMaker without modifications, so many facial features of the models looked very similar. HumanMaker models were then exported to Blender, and very simple low poly clothing was modeled on top of them. The clothing was then UV unwrapped very quickly, with some seams placed incorrectly. Next, the models had to be textured. Because high poly models were not used in the pipeline, a lot of detail was lost that could've been baked in from a high poly model. The first character models that were created by an inexperienced modeler looked very rough. Later on the models were created with a slightly more modern workflow, by using Substance Painter to create very basic textures with smart materials, and by faking textures by hand-painting variation to the normal map. This upgrade in the workflow made the models look better, as shown in Picture 9.


Picture 9. Comparing the models created by another artist (left), the new ones made by author (right).

Overall, when compared to the guideline in Chapter 3, the old workflow used in this project followed very few of the steps. From the Modeling column, only base mesh was created, and for textures the old models only had base colors applied with no variance to Albedo, Roughness of Metallic values. Naturally this way of working saved some time, but this style of working might be better suited for a more stylized look, as realism is not an issue. The new workflow went through the previously established workflow more accurately, and as a result the models looked more realistic. The main difference was in the usage of the textures. Instead of just flat colors, the models used smart materials that have Albedo, Roughness and Metallic variance to them. This step cost essentially no more time than in the old workflow, as the smart materials functioned as a drag-and-drop feature, meaning that the smart materials can be applied instantly to the meshes. The improvement of quality with minimal time investment is noticeable and should be kept in mind when there is a need for a big number of models in a small timeframe. With this kind of workflow, the models can be done roughly in one or two days, which is a quite fast pipeline for semi-realistic character models.

4.2 Virpa

Virpa is a VR project for PC, with the goal of making realistic character models. The timeframe of the modeling process was roughly two months, and the goal was to create at least five character models. As the modeling phase had to be started immediately to meet the deadline, the amount of time for concepting was limited, which actually was not very problematic, as it was easy to find relevant reference pictures. For example there are a lot of reference pictures on the Internet for the fireman character, which helped with the early production.

During this project it was possible to follow a better workflow for 3D characters. First, a low polygon character was created in Blender. This phase was fastest one, as the goal was to just block in very basic shapes, without a lot of detail. The low polygon version was saved and moved onto another layer. Later on it would be used to speed up the retopology phase. With the low poly clothing and accessories done it was easy to start working on the high poly sculpt. The detailing phase was done in Blender by utilizing the Dyntopo feature, that adds resolution automatically to the model where needed. Dyntopo is great for big changes, because the stretching of polygons is limited to minimum. Multiresolution modifier could have also been an option to add resolution to the model, but the changes to the base mesh were quite big, so there would have been some polygonal strectching with it. With the fireman character, there was no need to detail the face, because it was hidden under the mask, which gave more time to put details into the clothing. All of the high poly sculpting was done by sculpting the wrinkles manually, and a lot of reference pictures was used during this phase. For more realistic results, cloth simulations could have been used to generate high poly clothing for the character, but software like Marvelous Designer is not used in Turku Game Lab. Also, the computer that was used could not handle super dense meshes in Blender so very fine detail, like fibers and grain, could not be added to the clothes. The accessories, helmet, oxygen tank and mask were created by cage modeling with subdivision surface modifiers. This meant that there was no low poly model at the start, and these parts would have to be retopologized altogether.

High poly meshes were then retopologized in Blender. As the game was built for VR usage, the goal was to keep polygon count as low as possible. For the fireman the goal was to keep the count under 10 000 triangles. Retopologizing the model was fairly straightforward and easy to do by using parts of the previously created low poly base mesh and a retopology plugin for Blender called RetopoFlow. RetopoFlow speeded up the process by automatically filling in some parts of the mesh with guidelines.

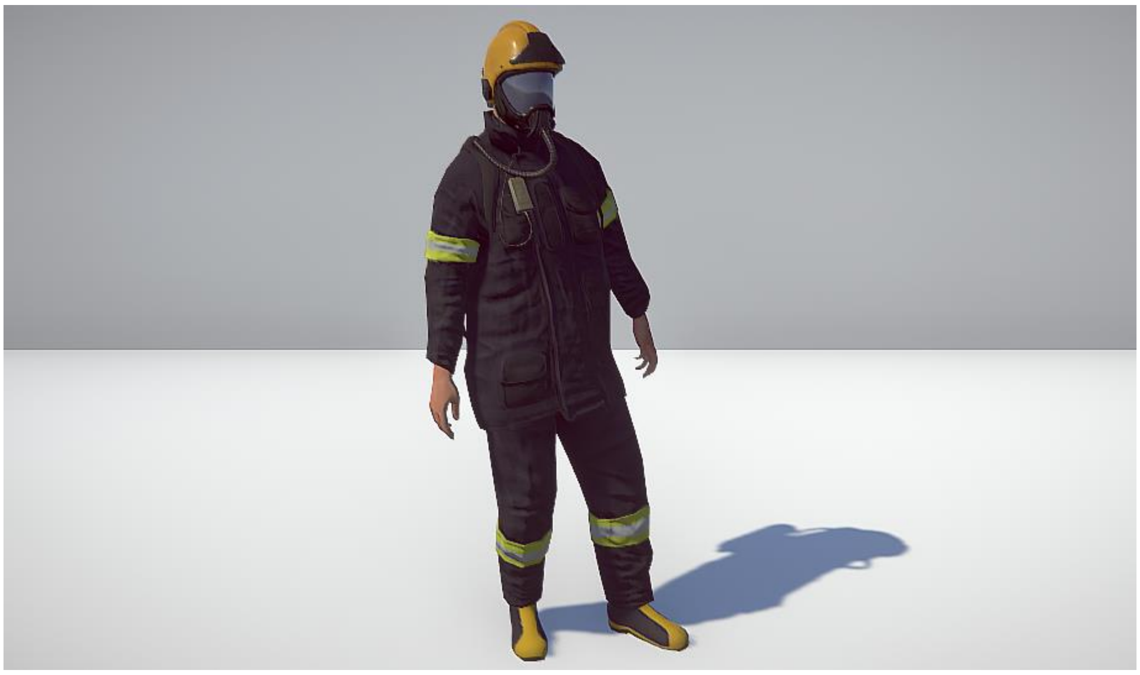
UV unwrapping of the clothing was done by marking seams on places that could not be seen very easily, like armpits, inner thighs and on the sides of the jacket. Hardsurface meshes were unwrapped on the same map by using planar projection, and cutting small seams in places where stretching occurred. UVs were then unwrapped on a grid to keep the texel density similar across the mesh. All of the accessoried had UVs mirrored because unique detailing was not important for these meshes, and clothing was unwrapped without mirroring because the creases and wrinkles were unique in each side. Lastly, in order to pack the islands as tightly as possible, a plugin called Packmaster was used to pack everything tightly together with a small margin. Usually it is better to organize the UV islands logically, but this was the faster approach.

Low poly and high poly meshes were then imported to Substance Painter for baking and texturing. The meshes had suffixes to differentiate low poly and high poly meshes from each other, so baking would be easier as there is no need to isolate meshes manually. The most important baked mesh maps for this model were Normal map, Curvature map and Ambient Occlusion map. Mesh maps were baked in 2048x2048 resolution and with 4x antialiasing to soften out artifacts. It is quite normal to see artifacts in mesh maps due to incorrect settings or cage object, but this time the details baked out perfectly on the first try.

Texturing started with filling in the base colours of the meshes. By looking at reference pictures, the most common color for fireman jacket seemed to be blue, and the yellow for the helmet. Everything expect the oxygen tank was dielectric, so only the tank had a metallic value of 1, and 0 for the other meshes. Some fireman clothing seemed have quite a low roughness value, but in the end the roughness value was set fairly high as it looked better in the test scene. Small surface irregularities like fibres were done with procedural textures. Cavity map was used to bump up the contrast between creases.

With this part done, the only thing missing was unique detailing, like dusty spots on the jacket and pants. By looking at reference pictures, dusty spots seemed to have high roughness value and slight desaturated discoloration. Simplest solution was to stamp the details with a custom alpha brush on a layer that affected roughness and color values. With this last step the texturing was done, and the mesh and the textures were exported to the Unity test scene, and the final result can be seen in Picture 10.

The modeling process of the fireman utilized many steps from the workflow that was introduced in Chapter 3, and the results were superior to that of the Lights on! project, as the model looked more realistic. The character creation followed all of the steps that were mentioned in Chapter 3, except pre-production, like concept art and turnaround sheets. All of the modeling and texturing was done by following the guidelines in Picture 7. Creating the high poly version of the model takes time, but the results are usually better than relying on painting height details manually. For example, the folds of the clothes are difficult to create without a high poly mesh. The texturing phase utilized some premade smart materials that were also tweaked by hand to create more realistic results. For a project like Lights on!, this workflow could not be used as there were too many models to be created, and following the guidelines more accurately takes more time. In this case the character models took about three to five days to create from start to finish.



Picture 10. One of the character models created for the project.

5 CONCLUSION

The aim of the thesis was to introduce the reader to the modern methods of 3D modeling by researching various concepts that are essential for a modeler to know. In addition to pure modeling theory, the thesis also explained the theory behind texture images, and their function in modern real-time engines. The focus was on PBR-based textures, because their usage has become imperative for achieving appealing models, both stylized and realistic. The intention was not to go into technical details, such the maths and physics behind PBR, but rather give a simple explanation on how to use specific maps in a modern pipeline.

In the last chapter of the theoretical part, a general workflow for creating modern models was introduced. It was challenging to find concrete information for this chapter, as there is variance in the pipeline of 3D asset production, and major companies are not willing to share their exact workflows. However, after researching the topic, a general workflow could be found, even though the specific software that the companies use was difficult to pinpoint. In the end this was not a problem, as the workflow that was introduced can be followed in most of the 3D modeling software. The initial aim was to make the workflow chapter contain many separate techniques that can be used only with specific software, such as cloth simulation in Marvelous Designer or hair modeling in Zbrush. This part was omitted because it would have given the thesis too wide of a scope. Additionally, the rigging and animation parts were not included even though it is a part of character creation. These topics are very extensive, and could not be covered in the scope of the thesis.

The practical modeling part was interesting to carry out, as it showed the differences of character models that were done with an outdated workflow and ones that followed the modern workflow more accurately. Both projects had very tight deadlines, and especially in the case of Light On!, the results could be seen clearly. Many necessary steps had to be skipped to stay in schedule and the sheer amount of models seemed to affect on the quality as well. In Virpa, the models looked much better because a more modern pipeline was followed and fewer models had to be made. However, there were some problems that were not present in Lights on!, such as problems with exporting the assets to Unity with correct animations. This caused delays in an already tight schedule. It is important to note that the models created for Virpa took a couple of days longer than the ones in

Lights on!. If visual quality is not a priority or the models are seen from far away, the modeler can get away by taking the easier route and skipping some steps. If the player can not see the details that were carefully created by the modeler, it might be a good idea to invest resources elsewhere.

The thesis could have been implemented better by planning the scope better and narrowing it down to a singular topic. This would have made it possible to go more in-depth with, for example the general workflow, instead of trying to cover every aspect of 3D modeling. Additionally, the practical chapter could have been more detailed and included more steps of the process to give a better and more detailed explanation of the workflow. Unfortunately not many screenshots were taken during modeling, so step-by-step pictures were not possible after the models were finished.

REFERENCES

Aava, K. 2017 Realistic vs. Stylized: Technique Overview Referenced 18.9.2018 https://80.lv/articles/realistic-vs-stylized-technique-overview/

Adams, T. 2018 Announcing MudBox 2018.2 Update and Introducing Dynamic Tesselation. Referenced 5.9.2018 https://area.autodesk.com/blogs/thebuzz/announcing-mudbox-20182-update-and-introducing-dynamic-tessellation/

Allegorithmic. 2016 Understanding PBR Referenced 05.10.2018 https://www.youtube.com/watch?v=ueC2qGzWrgQ

Allegorithmic. 2018 The PBR guide – Part 1 https://academy.allegorithmic.com/courses/b6377358ad36c444f45e2deaa0626e65 Referenced 04.10.2018

Autodesk. 2018 Planar UV mapping. Referenced 15.9.2018 https://knowledge.autodesk.com/support/maya/learn-explore/caas/CloudHelp/cloudhelp/2018/ENU/Maya-Modeling/files/GUID-B6519472-C0ED-4C07-99C6-12107A3509D9-htm.html

Blender Docs 2018 Seams. Referenced 15.9.2018 https://docs.blender.org/manual/de/dev/editors/uv_image/uv/editing/unwrapping/seams.html

BlenderArtists. 2011 What exactly is a “matcap” and how does it aid sculpting? Referenced 15.9.2018 https://blenderartists.org/t/solved-what-exactly-is-a-matcap-and-how-does-it-aid-sculpting/496810

Burke, S. 2015 Defining PBR (Physically-Based Rendering) Graphics with Crytek Developers Referenced 04.10.2018 https://www.gamersnexus.net/gg/1866-what-is-pbr-cryengine-star-citizen

Bär, S. 2018 The advantages of Substance as PBR texturing tool Referenced 24.9.2018 https://www.artstation.com/sebastianbaer/blog/pn9e/the-advantages-of-substance-designer-as-pbr-texturing-tool

Cadsourcing. 2016 What’s the difference: CAD vs. 3D modeling vs. BIM Referenced 1.9.2018 http://cadsourcing.com/whats-the-difference-cad-vs-3d-modeling-vs-bim/

CG Cookie 2016 Big idea: Baking Referenced 21.9.2018 https://cgcookie.com/articles/big-idea-baking

Flipped Normals. Top Tips for Improving Your Sculpts in Zbrush Referenced 15.9.2018 https://www.youtube.com/watch?v=3boCwp9uLsY

Heginbotham, C. What is Photobashing? Referenced 12.9.2018 https://conceptartempire.com/photobashing/

Johns, M. 2016 Create a texture set for games in 10 steps Referenced 20.9.2018 https://www.creativebloq.com/3d/create-texture-set-games-10-steps-51620492

KatsBits 2018 Make Better Textures, The ‘Power Of Two’ Rule & Proper Image Dimensions. Referenced 19.9.2018 https://www.katsbits.com/tutorials/textures/make-better-textures-correct-size-and-power-of-two.php

Lezzi, L. 2016 Figuring out Texel Density. Referenced 20.9.2018 https://80.lv/articles/textel-density-tutorial/

Pavlovich, M. Introduction to Zbrush Part 1. Referenced 12.9.2018

Maximov, A 2014 Physically Based Texturing for Artists Referenced 8.10.2018 https://artisaverb.info/PBT.html

Moore, 2014 Creating Whole New Worlds Referenced 18.9.2018 https://www.craftsy.com/art/article/what-is-concept-art/

Pixologic Documentation. 2018 Zspheres. Referenced 18.9.2018 http://docs.pixologic.com/user-guide/3d-modeling/modeling-basics/creating-meshes/zspheres/

PluralSight. 2014 Eliminate Texture Confusion: Bump, Normal and Displacement Maps Referenced 8.10.2018 https://www.pluralsight.com/blog/film-games/bump-normal-and-displacement-maps

PluralSight. 2014 Texturing for Games Referenced 16.9.2018 https://www.pluralsight.com/blog/film-games/texturing-games-maintain-high-level-detail-without-extra-geometry

PluralSight. 2014 Why are Ngons and Triangles so Bad? Referenced 15.9.2018 https://www.pluralsight.com/blog/film-games/ngons-triangles-bad

PolyCount. 2014 Polycounts in next gen games thread! Referenced 10.10.2018 https://polycount.com/discussion/141061/polycounts-in-next-gen-games-thread

PolyCount. 2014 Best auto UV unwrap? Referenced 17.9.2018 https://polycount.com/discussion/137137/best-auto-uv-unwrap

Polycount. 2018 Texture Coordinates. Referenced 15.9.2018 http://wiki.polycount.com/wiki/Texture_Coordinates

Polycount. 2018 Texture Baking. Referenced 16.9.2018 http://wiki.polycount.com/wiki/Texture_Baking

PolyCount. 2014 Understanding averaged normal and ray projection Referenced 16.9.2018 https://polycount.com/discussion/81154/understanding-averaged-normals-and-ray-projection-who-put-waviness-in-my-normal-map%2019.09.2018

Polycount. 2017 Subdivision Surface Modeling. Referenced 18.9.2018 http://wiki.polycount.com/wiki/Subdivision_Surface_Modeling

PolyCount. 2018 Blender question regarding soft/flat shading on. Referenced 5.9.2018 https://polycount.com/discussion/197162/blender-question-regarding-soft-flat-shading-on-this-low-poly-model-i-am-making

Rouse, M. 2018 3D model Referenced 10.9.2018 https://whatis.techtarget.com/definition/3D-model

Slick, J. 2018 What is 3d modeling Referenced 12.9.2018 https://www.lifewire.com/what-is-3d-modeling-2164

Sosak, Y. 2018 Should You Get A Graphics Tablet For Sculpting? Referenced 14.10.2018 https://www.youtube.com/watch?v=R1CaEph3DTU

Substance Share. 2018 https://share.allegorithmic.com/ Referenced 14.10.2018

Taylor, J. 2015 Why are Triangles bad when modeling? Referenced 15.10.2018 https://www.methodj.com/why-are-triangles-bad-when-modeling/

Taylor, J. 2016 Modeling vs Sculpting: How do you know which to use? Referenced 5.9.2018 https://www.methodj.com/modeling-vs-sculpting/

Thilakanathan Studios 2016 Why do we need Topology in 3D Modeling Referenced 15.10.2018 http://thilakanathanstudios.com/2016/09/why-do-we-need-topology-in-3d-modeling/

TurboSquid. 2017 Tris, Quads & N-Gons. Referenced 3.9.2018 https://www.turbosquid.com/3d-modeling/training/modeling/tris-quads-n-gons/

Topology Guides 2018 Moving and Manipulating Edge Poles Referenced 15.10.2018 https://topologyguides.com/

Youtube. 2017 3D Vehicle Design: Concepts and Workflow. https://www.youtube.com/watch?v=QhaWh_MQDTo

Wilson, J 2015 Physically-based rendering, and you can too! Referenced 8.10.2018 https://marmoset.co/posts/physically-based-rendering-and-you-can-to