

Elisabet Heikkilä

A Guide to Building a 3D Game Character



Bachelor of Business
Administration

Business Information
Technology

Spring 2017



KAJAANIN
AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES

TIIVISTELMÄ

Tekijä(t): Heikkilä Elisabet

Työn nimi: Ohjeistus 3D-pelihakmon tekoon Blenderissä

Tutkintonimike: Tradenomi, Tietojenkäsittely

Asiasanat: 3D-mallinnus, peligrafiikka, pelihakmo

Opinnäytetyö käsittelee 3D-hahmon luomisen kaikkia vaiheita mallikuvien teosta aina valmiin 3D-mallin riggaukseen. Työn eteneminen selostetaan niin tarkasti, että 3D-mallinnuksesta kiinnostunut lukija voisi luoda oman hahmonsa kuvauksen perusteella. Opinnäytetyön päämääränä ei siis ole luoda viimeisteltyä pelihakmoa, vaan ohjeistaa 3D-hahmon luomisen jokainen vaihe esimerkin avulla.

Opinnäytetyö on jaettu neljään osioon. Ensimmäinen osio kertoo 3D-hahmon luomisprosessin teoriasta. Toinen osio selostaa teoriaa hyvän 3D-hahmon rakenteesta ja hyvistä käytänteistä mallinnusprosessin aikana. Kolmas osio sisältää perusteet Blender-ohjelmiston käytöstä, jotta ohjelmaan tutustumatonkin lukija voisi halutessaan opetella käyttämään Blenderiä. Neljäs osio puolestaan sisältää itse käytännön osuuden, eli 3D-hahmon luomisen ohjeistuksen.

Käytännön osio alkaa hahmon mallikuvien, eli 3D model sheetin, luomisesta ja antaa vinkkejä mallikuvien luomiseen. Kappale keskittyy eri tapoihin pitää mallikuvat yhtenäisinä. Mallikuvien teon ohjeistuksen jälkeen aiheeksi nousee itse 3D-mallin kokoaminen. Työssä kuvaillaan, missä järjestyksessä hahmon ruumiinosat luodaan, mitä työkaluja käytetään ja miten osat lopulta kiinnitetään toisiinsa. Jokainen vaihe on havainnollistettu kuvin ja kuvatekstein, jotta ohjeistusta olisi helpompi seurata. Lisäksi käytännön osuus kertoo pikaisesti, miten valmiiseen 3D-hahmoon lisätään tekstuurit. Lopuksi käytännön osuudessa käydään läpi 3D-hahmon rigin teko, jotta hahmo voitaisiin animoida myöhemmin.

ABSTRACT

Author(s): Heikkilä Elisabet

Title of the Publication: A Guide to Building a 3D Game Character

Degree Title: Bachelor of Business Administration, Business Information Technology

Keywords: 3D modeling, video game art, game character, game development

The thesis describes the entire process of creating a 3D game character from the creation of the reference material, the 3D model sheet, all the way to the creation of a rig for the final 3D model. The process is described in such detail that a reader interested in 3D modeling could create their own character alongside the thesis. The goal of the thesis is not the creation of a polished character model but rather to walk the reader through every step of the way the help of an example character.

The thesis has been split into four main parts. The first part describes the process of creating a 3D game character in theory. The second part focuses on the structure of a good character model along with some tips on good practices to make the work more streamlined. The third part is a brief guide to the basics of the Blender program so even a reader who has never worked with the program could make use of this guide. The fourth part includes the final guide to creating a 3D character.

The guide begins with the creation of the 3D model sheet and gives tips to keeping the reference images consistent with each other. After the reference images are done, the next phase is the creation of the 3D model itself. The chapter describes in what order the character's body parts are created, what tools are used and how the finished parts are combined. Each step is visualized with pictures from the example character. The guide also quickly shows how textures are added to the finished model. Lastly, the chapter walks through the creation of a rig for the 3D model so the character could later be animated.

TABLE OF CONTENTS

1 INTRODUCTION.....	1
2 THE PROCESS OF 3D CHARACTER CREATION	2
2.1 3D Model Sheet.....	2
2.2 3D Modeling	4
2.3 UV Unwrapping and Texturing	5
2.4 Rigging a Character for Animation	6
3 THE STRUCTURE OF A GOOD CHARACTER MODEL.....	8
3.1 Polygon Count.....	8
3.2 The Default Position	11
3.3 Clean topology	11
3.4 The structure of the joints.....	13
3.5 Good practices	14
3.5.1 World Scale.....	15
3.5.2 Naming Conventions.....	15
3.5.3 Version control	16
4 THE BASICS OF BLENDER	17
4.1 Setup.....	17
4.2 The interface	19
4.3 Hotkeys	24
4.4 Modifiers.....	25
5 3D CHARACTER CREATION STEP BY STEP.....	27
5.1 Creating the 3D model sheet.....	27
5.1.1 The front and back views	28
5.1.2 The sides and other views.....	32
5.1.3 Bringing the references to Blender	34
5.2 Creating the 3D Model	36
5.2.1 The torso	38
5.2.2 The limbs and joints	49
5.2.3 The feet and hands	56

5.2.4 The hands	59
5.2.5 The head and face	68
5.2.6 Clothes	88
5.2.7 Hair	96
5.3 Unwrapping	104
5.4 Texturing	109
5.5 Rigging	115
6 CONCLUSIONS	128
REFERENCES	130
ATTACHMENTS	

LIST OF SYMBOLS

Vertex: A point where two or more lines or edges meet. For example the corners of a polygon.

Edge: The line between two vertices.

Polygon: In 3D modeling, the word polygon refers to the individual parts that form the surface of a 3D object (mesh). In games, 3D models should be formed of three or four sided polygons.

Edge loop: A line of edges that circle around the model as an intact loop.

Optimization: In 3D modeling, optimization refers to creating a model as light as possible for the game engine to run. Optimization includes for example keeping the polygon count of a model as low as possible.

Layer: In 2D programs layers are like transparent sheets which can be drawn on without it affecting the contents of the other layers. Layers can be hidden, unhidden and their order can be changed. 3D programs can also make use of layers.

Rigging: The art of turning into a static 3D mesh into something that can be animated.

Rig / Armature / Skeleton: A system for animating 3D meshes. A rig is made by creating bones, linking them together and attaching the final product to an 3D object. The bones can then be moved to change the shape of the 3D mesh.

BLENDER-SPECIFIC TERMINOLOGY:

Object mode: A mode where new objects can be created or moved around.

Edit mode: A mode where the created objects can be edited in more detail. Only object at a time can be opened in edit mode. If a new cube is created in Edit mode, it will be considered a part of the same object even if it would be separate in the 3D space.

Scene: The 3D space where a model is created.

3D Cursor: A tool which determines where new objects and shapes are created. 3D cursor can be moved wherever it would be most useful. The cursor can be brought back to the middle of the scene with the “Shift + S → “Cursor to Center” command.

Origin: An object’s root point. The origin functions as for example as an object’s middle point when using mirroring tools and as the object’s axis when rotating. By using the “Shift + Ctrl + Alt + C” combination while in Object mode, the active object’s origin can be moved either to the object’s center of mass or to the 3D Cursor’s current location.

Pivot Point: The scene’s pivot center which determines how an object behaves when scaled or rotated. Changing the Pivot Point can for example change whether the object rotates around its own Origin or around the 3D Cursor.

1 INTRODUCTION

The goal of this thesis is to create a step-by-step guide for creating a 3D character for a video game. The guide describes each step from the creation of the reference images to 3D modeling the character in Blender. The guide also explains how the finished 3D model can be unwrapped, textured and prepared for animation. Each step is explained in detail so the reader could create their own 3D character alongside the guide. Each action and quick key used in Blender is explained so the reader can replicate the process without external help. The guide is aimed at people who have at least some knowledge of how Blender works but a chapter on the very basics of Blender is included for those who have never tried the program before and only have background in 3D programs like 3DS Max or Maya.

A character model will be created as an example but the focus of the thesis is not on creating a polished character. Instead, the goal is to explain the process so someone else could make their own character using the thesis as a guide. The thesis will explain how this one character model was created while pointing out areas where problems arose and how the said problems could possibly be avoided in future characters.

The topic of the thesis is quite extensive, but the guide will be based on my topical seminar paper from late 2016. The seminar paper focused on the steps of creating a simple 3D character model based on a pre-existing 3D model sheet while this thesis builds upon and extends far beyond that. The information and images in the thesis will also be used to create compact study material for a 3D character creation course for the Kajaani University of Applied Sciences.

2 THE PROCESS OF 3D CHARACTER CREATION

Creating a 3D character is a greater undertaking than people often realize. In large studios, the job is usually divided among several artists due to the time it would take and the wide range of skills that the creation of a single character requires. In smaller projects, the character designs, 3D models and animation might be created by one person but this greatly limits the amount of characters that can be added to the game. Usually it's better to have a different artist for each part of the pipeline. This way the job gets done faster as the workload is shared. For example, the character designer can work on a new character design while the modeler still works on the 3D version of the first character.

In some cases, a team might try an approach where several artists create model sheets and create their own characters for the game from start to finish, but generally it is better to have artists focus on a specific area of the pipeline. This way the team can ensure the art style of the production remains consistent. (Masters 2015) While smaller teams may not have the luxury of having enough artists to share the workload this way, allowing an artist to focus on one thing instead of having them work as a jack-of-all-trades makes the process more streamlined.

As an example, we could use the character models of the game Overwatch by Blizzard Entertainment. According to the Overwatch team's lead character artist's ArtStation submissions, the work is usually shared in the following manner: one person created character concepts, another one the models, third artist handles the rigging and animation work is left for a fourth artist. Weapons were also often created by a separate artist. (Galand 2016)

2.1 3D Model Sheet

While gorgeous concept paintings of characters work well for marketing purposes, they are less helpful from a 3D modeler's perspective. When the character designs and 3D models are made by two different people, a single painting of a character

in a heroic pose leaves many questions about the design unanswered. If the modeler has not familiarized themselves with the reasons behind the character's design, they may accidentally misinterpret certain areas and thus the feel the character gives might change. (Pluralsight 1, 2015)

3D model sheet is a character designer's tool for conveying their vision to the modeler efficiently. A 3D model sheet displays the character in a neutral pose from several angles. (Image 1) Usually the character is shown at least from the front, back and side. There may also be other information on the sheet. For example, drawings that highlight certain details of the character or even photographs of different materials that the character's design would need. (Zagobelna, 2014)

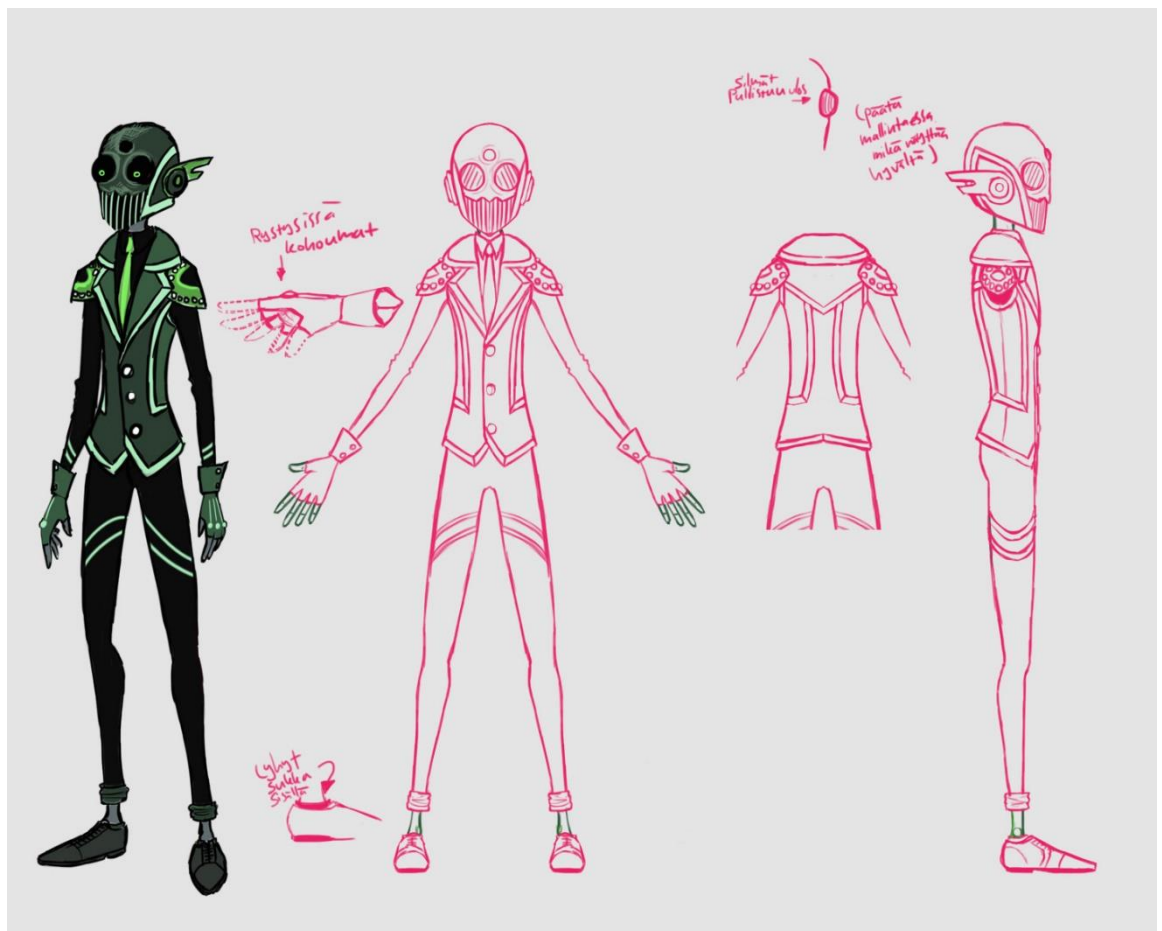


Image 1. An example of a 3D model sheet with rough concept art alongside a front view, side view and a partial back view. A few extra details are shown with added images.

One thing that makes 3D model sheet different from usual concept art is the fact the subject is drawn in an orthographic view (Pluralsight 1, 2015). This means that a character's each body part is pictured as it would appear without the distortion caused by perspective. In a picture with orthographic projection every part appears their actual size regardless of their distance from the viewer. Image 2 demonstrates the difference between perspective and orthographic projections.

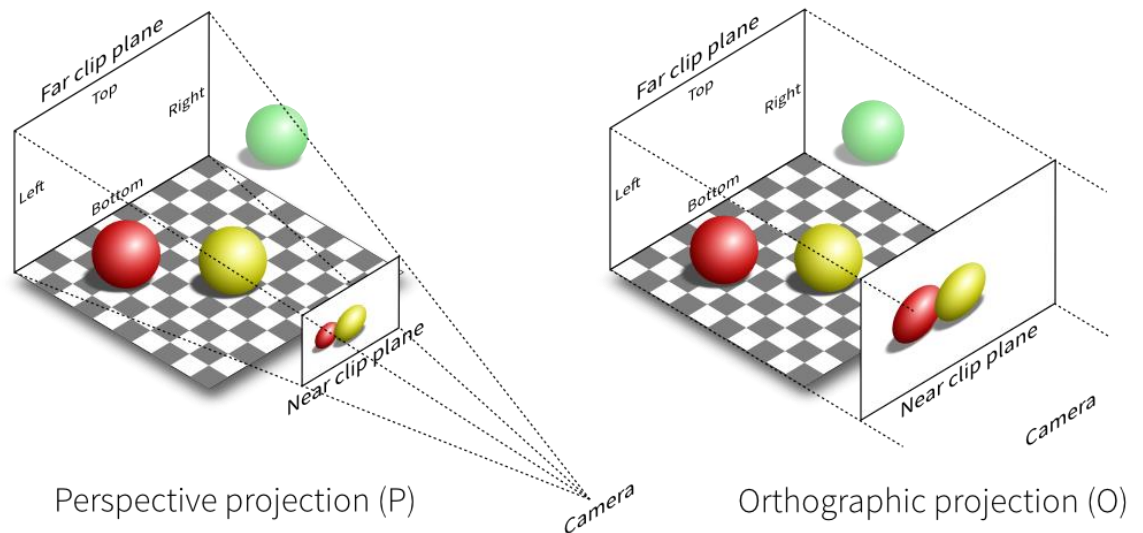


Image 2. The difference between perspective and orthographic projection. (Script Tutorials)

The references can be imported to a 3D modeling software and used as a blueprint when modeling the character. This makes the job much easier for the modeler, especially if they are not too familiar with the anatomy of the creature being modeled. It also allows the modeler to focus on creating a model with good topology without needing to worry about getting the character's body proportions right. In a way, a 3D model sheet is a 3D artist's sketch for creating the final model. (Pluralsight 1, 2015)

2.2 3D Modeling

After the 3D model sheet is done, the images can be imported to a 3D modeling program. The images can be displayed as background images over which the modeler can begin shaping the character. The model itself usually starts from a

simple shape like a cube. By adding loop cuts and moving the vertices around, the 3D modeler can create the character's body. Limbs can be modified from separate cylinders that are finally attached to the body once they are ready. The head can either start from a cube or be crafted polygon-by-polygon per the provided reference pictures.

If the character or object is symmetrical, the modeler can use mirroring tools to create a perfectly symmetrical model. When a mirror modifier is active, every change done to one side of the model will be automatically done on both sides of the model's middle line. This greatly speeds up the modeling process. Once the symmetrical version is done, the mirror modifier can be collapsed onto the model so both sides of the model can be edited individually if there are parts that need to be asymmetrical on the model. Any symmetrical parts are though best kept under the influence of the mirror modifier to ensure all future steps like texturing and rigging can be finished quickly and painlessly.

2.3 UV Unwrapping and Texturing

Game characters rarely are made of a single material like clay or glass. In order to give a character features like skin, clothing and hair colour, texture maps are essential. But before these maps can be created and applied, the modeler needs to unwrap the UVs of the model. UV, or UVW, represent the different axes where the 3D model exists. UVW represent the XYZ axes respectively. (Autodesk, 2016)

The first step of UV unwrapping a finished model is placing the seams. Like in real clothes and map projections, the seams are a way to break down a 3D object into 2D shapes. The modeler will choose edges on the model, mark them as seams and have the program place the cut-out shapes onto a UV layout. The UV layout can be exported from the program into a painting program like Photoshop and a 2D artist can paint the textures there. The finished textures will be brought back to the 3D program and applied to the character model.

Still, 3D game characters and objects rarely only use one texture map that gives the object colour, also known as a diffuse map. The model's surface can be made

more realistic by using different texture maps that can for example alter the model's transparency (transparency map), shininess (specular map) or shape (normal, bump and displacement map). (Slick 2, 2016)

When it comes to diffuse maps, they may be either made from photographs or painted from scratch. It depends on the game's art style which of these methods should be used. A game with stylized characters, like World of Warcraft, work well with hand-painted textures while a more realistic game, like Call of Duty, would look more comical if the otherwise realistic people and environments would have painted textures. In image 3, the normally-realistic graphics of Witcher 3 have been changed to simple painted textures.



Image 3: Giving simple textures to a game with realistic models can lead to interesting results. (Gamespot)

2.4 Rigging a Character for Animation

Usually, a 3D model by itself is a static mesh which cannot be animated effectively. Especially when working with the model of a creature, creating a system for moving its limbs is important. The task of creating a system for moving a character by

creating bones and joints is referred to as a rigging while the structure itself can be referred to as a rig, armature or skeleton. (Slick 3, 2016)

An armature is built of bones that form chains and are placed inside the character model. There are various kinds of bones and joints that can be used in rigging. The two main types of bones are the deforming bones and the control bones. Deforming bones are the bones that are directly connected to certain parts of the 3D mesh. Moving, rotating and scaling a deforming bone will also alter the character model accordingly. Meanwhile, control bones are used to control other bones, directly or indirectly. Moving a control bone will not directly affect the surface of the model, but it may move other bones which again deform the mesh. (Blender 2.78 Manual)

When it comes to creating a system to move the joints, the two main types are Forward Kinematics (FK) and Inverse Kinematics (IK). Forward Kinematics refer to a system where the bones are moved one by one to create a pose. For example, posing hands is usually done by first bending the bones closest to the palm and moving outwards. Inverse Kinematics are a system where the bone at the end of the chain is moved and the program moves the bones below it accordingly. For example, if used in a hand, the fingers could be pose by only moving and rotating the tip of the finger. IK systems are though somewhat tricky to create and can move in unpredictable ways if not properly executed, thus requiring the artist to do some manual cleanup later. IK systems work well when used in arms and legs while FK can be used for fingers and spines. (Slick 3, 2016)

Once the armature is finished, the model is attached to it in a process called weight painting. Some programs like Blender have the option to automatically assign the character mesh to the bones but the automatic weights usually need to be adjusted. A common example of a problem that can occur with automated weight painting is the character's clothes or hair moving at a different pace than the body part it should cover when a limb is moved, thus causing the body to clip through and leave the clothing or hair behind. (Pluralsight 2015) Depending on the complexity of the rig and the model, the process of creating a working rig can take anything from a few hours to several days or even weeks. (Slick 3, 2016)

3 THE STRUCTURE OF A GOOD CHARACTER MODEL

There are several things that can go wrong when creating a 3D character. The model's topology can be a mess, the UV layout can be a pain to work with or the rig can be prone to breaking the model when animated. When a character is crafted poorly, it can be quite an eye-catching sight, if not a disturbing one, while a well-made one can greatly help the player focus on the game itself. It is good to know what the good practices of 3D modeling are before spending too much time trying to create a complex character for a game.

3.1 Polygon Count

In the case of video game characters, topology and optimization are important. Because in games the player chooses what their character does, everything needs to be rendered in real time which means every image you see is created as you play. Creating these images takes processing power and time depending on how much detail the character models and environments have. If the models would be too complex, the player would have to wait to see the results of their actions because rendering the images would take so much time. (Silverman, 2013)

But what is a good amount of detail for a character model to have? It all depends on the platform the game is aimed for. The Unity User Manual states that the ideal polygon count for mobile games is 300 to 1500 polygons, while desktop platforms can handle character models of 1500 to 4000 polygons. If there are multiple characters on the screen at the same time, the polygon count may need to be reduced to lessen the load (Unity User Manual (5.5), 2017).

Movies are not interactive like games which means they can be pre-rendered. This means rendering time is not an issue and the characters and environment can be as detailed as the creators wish. (Silverman, 2013) In video games, cut scenes can be pre-rendered and some games really do use cut scenes to show off their artists' skills. For example, the Nintendo DS version of Final Fantasy IV includes an opening cinematic with character models so detailed the console would have

no hope of rendering them in real time. The game itself is played with heavily stylized and simplified character models. (Image 4)



Image 4: Same game, same characters, different priorities. Screenshots from Final Fantasy IV's Nintendo DS release.

It is important to take into consideration the game's needs when creating the character models. In the right picture above, we can see the in-game character models' hands are very simple to the point of resembling hooves more than human hands. The characters in the game simply do not need individual fingers so the modeler decided to leave them out. Image 5 shows two different kinds of hands a game character might have. Another point to consider is whether the character needs a three-dimensional mouth or if talking can be animated with a 2D texture. It all depends on how close the characters will be to the viewer and what the character will do. Small simplifications like these can quickly reduce the character model's polygon count.



Image 5: A hand with individual fingers and a hand with 3 fingers fused together. (Ward 2011)

There is though one thing where a character model with a high polygon count is often used in games and that is in the creation of texture maps. A certain texture map is used to create the illusion of detail that responds to light without giving the actual model these shapes with polygons. This map is called the normal map. A normal map uses three different colours to store the direction of the normal of a pixel. These maps are created by creating a low polygon and high polygon version of a character and using a 3D program to in a way project the high polygon model's details onto the low polygon model. When the normal map is applied to the low polygon model, its surface will respond to light the same way as the high polygon model would while still remaining quick to render. (Polycount, 2016)

3.2 The Default Position

When a character model sheet is made for a 2D production, the character is often drawn in a pose that shows how the character's limbs are drawn in different positions. (Masters 2015) In a 3D model sheet this is not advisable as the character's pose will be defined by it. In 3D games, characters are usually modeled in a pose where the arms are extended straight to the sides. This is referred to as the T-pose. The point of the pose is to make the character easier to model, rig and animate. After all, if the character's arms were down and relaxed, it would make it difficult for the modeler to work on the character's sides, armpits and the underside of their arms. Especially weight painting in the rigging phase can get messy if the arms are close to the torso. Still, the traditional T-pose has its problems as well. In T-pose, it is harder for the animation software to figure out which way the arm should bend and the stretching on the shoulders is more prominent when the character lowers their arms. (Engländer, 2015)

In newer games, you can see the characters' arms at a slight downward angle. The change is quite recent and there haven't been many official explanations for it. One theorized reason for this is the fact game characters rarely need to raise their arms straight above their head which means the midpoint of the usual range of motion is lower. Having the arms slightly down in the base pose makes the topology of the shoulders more natural and reduces stretching of the shoulders when the arms are being moved. (Polycount, 2011)

3.3 Clean topology

When a model is brought into a game engine, it will be triangulated. This means every polygon will be converted into tris, triangle-shaped polygons. If the model already consists of tris or quads everything should be fine. After all, tris need not be triangulated further and quads can only be triangulated in two different ways. Though the artist should not mix tris and quads too much. They should stick to one of the two polygon types. Usually quads are a better choice as they can be triangulated for game engines or subdivided for sculpting as needed. (Slick, 2016)

Problems may though arise if there are polygons with more than four edges edges, so-called n-gons. The more edges a polygon has, the more different ways there are for the engine to triangulate it and the results are unpredictable. (Silverman, 2013) Image 6 shows all the ways a pentagon-shaped polygon could be triangulated.

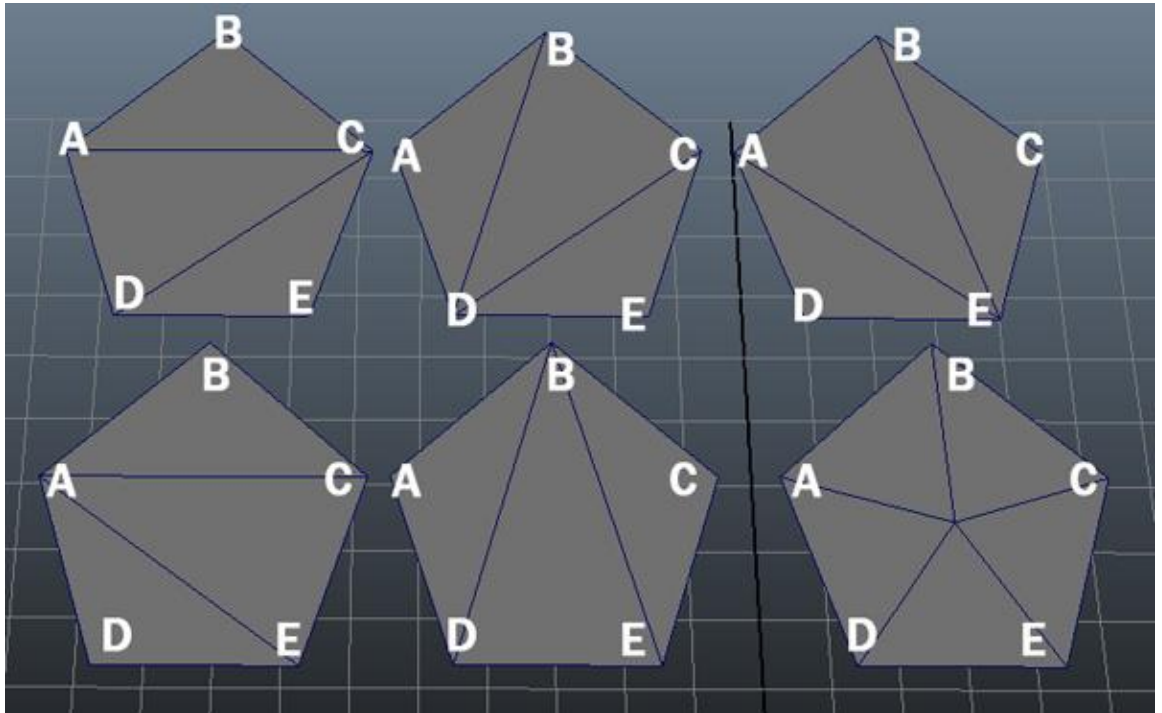
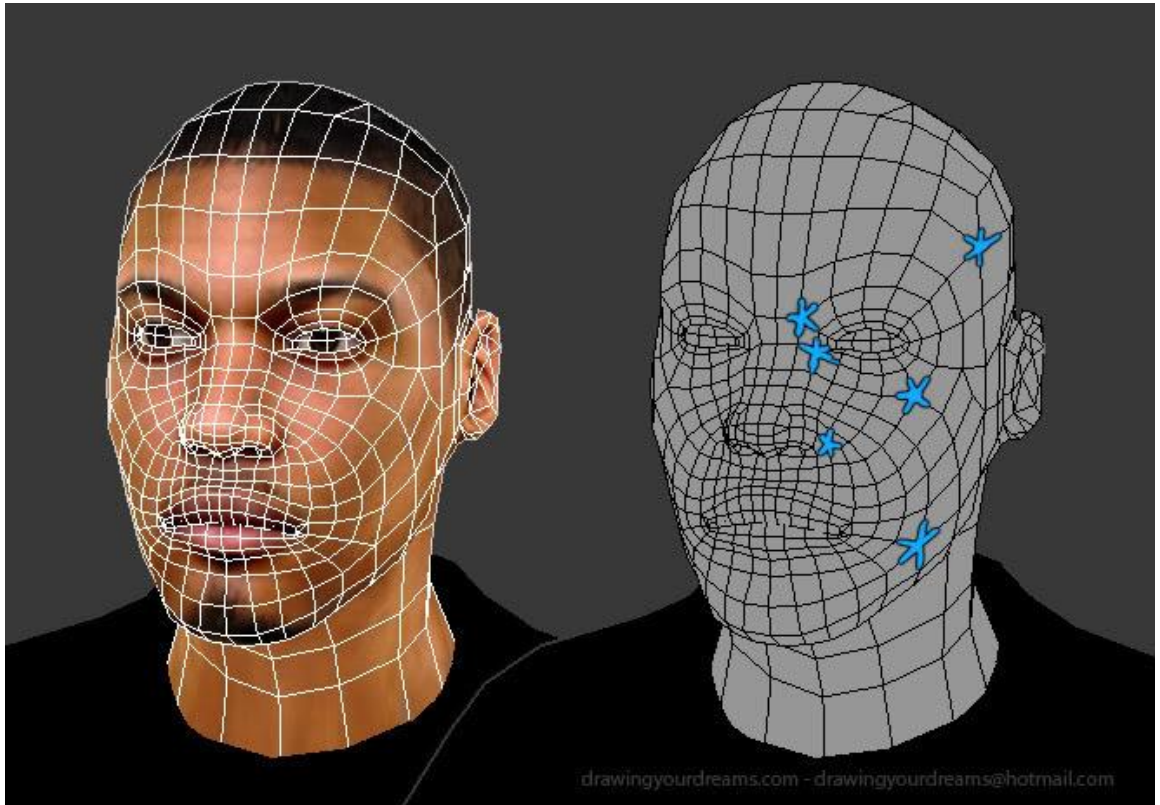


Image 6: Triangulating a pentagon. (Silverman, 2013)

When creating a character model, it is a good idea to try to match the topology with natural muscle lines. This ensures the character's body will behave more naturally when animated. Another point to remember is keeping the topology clean and even. The modeler should aim to have the body's polygons form a neat grid (Ward, 2011). One thing that can cause problems in topology is having poles in the mesh. A pole is a vertex where 5 or more vertices connect. Poles can be used in certain areas of the model, as shown in image 7, but they generally can cause problems if the modeler does not know where to place them. (Polycount Del, 2011) Poles disturb the flow of edge loops and might even cause an edge loop that seemed fine to suddenly spiral around the model several times, making it difficult to later on add edge loops to the model.



(Image 7: Examples of where poles can be placed on a character's face to help avoid triangles among quads. (Polycount Del, 2011))

When creating any 3D model, the modeler should remember to remove any polygons that cannot be seen. In a first-person shooter, they might want to delete the polygons on the underside of the player character's gun. Similarly, if a character has their body and clothes modeled separately, any parts covered by clothes should be deleted from the nude model of the body. When it comes to edge loops, any loops that do not alter the model's shape should also be removed (Ward, 2011).

3.4 The structure of the joints

When it comes to characters that need to be animated, the structure of joints needs to be planned well. There are several ways to handle the structure of a 3D character's joints, some more natural-looking than others. When a 3D character is animated, its body parts will deform. Some polygons will stretch while others will shrink. Problems though will arise if the areas that move will cause the character's

body structure to collapse. Image 8 shows 3 different joint types and how they look when bent to a 90-degree angle. The third type demonstrates a joint that collapses when bent.

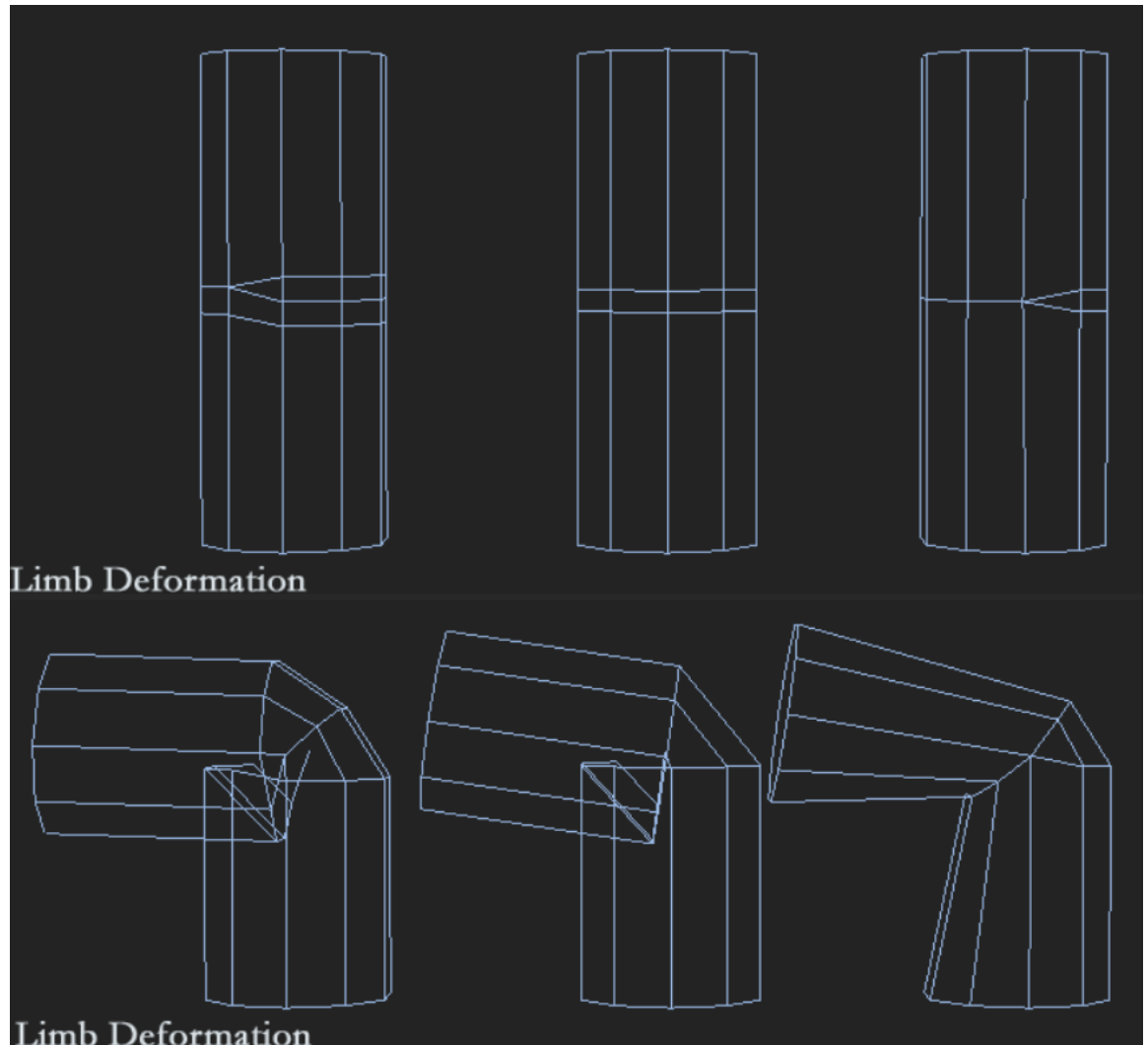


Image 8: Different kinds of joints. (Polycount, 2015)

3.5 Good practices

When creating assets for a video game, there are certain things the artists should do to make each other's work easier. After all, video game development is a team effort and all the work needs to be combined eventually. The artists should together decide on things like world scale and naming conventions. There is also always the chance that someone must pick up someone else's unfinished work

and finish it. If this happens, it is especially important that the 3D files are well organized and easy to navigate.

3.5.1 World Scale

World scale means setting the measuring system and grid size of every artist's 3D program to be accurate to the game world. Centimeters are a commonly-used system in most studios. Setting the grid size to be the same on every artist's computer makes it easier to keep all the assets in scale to each other and helps the technical artists to create shaders that work in the chosen scale. Without a world scale system like this, the environment assets may turn out massive compared to the player character or the lighting may turn out too strong or weak. (Rinaldi)

3.5.2 Naming Conventions

regardless of whether the artist is working on 3D models or 2D textures, naming everything is important. Naming the files is a given, but also every separate object or layer should be named in case the piece needs to be later on reworked either by the original artist or by someone else. If every object has generic names like "Sphere1" or "Cube3", finding the object that is a specific piece of the character's armour wastes time. Every object and material should have a name that is easy to recognize. This also goes for creating 2D textures; trying to find the layer that has the character's shirt could take a while if every layer is named "Layer#".

Naming everything is especially important when rigging. Because bones rarely are too unique in appearance, every bone should have a name that is easy to recognize even when the character has been pulled into a pose that is hard to read. For example, a bone that controls the rotation of a quadrupedal character's left front knee could be named "FrontKnee_L".

One commonly used naming convention is the CamelCase. In this naming convention, everything is written without spaces and with every new word capitalized.

In programming the first word is generally not capitalized. For example, “redBrick-Wall” could be a material named in CamelCase. Using the same naming style as the programmers makes it easier for them to work with the art assets too. (Rinaldi)

3.5.3 Version control

There is always the chance that recent changes to a model need to be rolled back in one way or another. The modeler may have removed a part of the model without noticing or maybe they accidentally applied a modifier and are unable to trace back the changes it made. This is where version control proves useful. Whenever the modeler is comfortable with their progress, they can save their current work as a separate file, like save states in games. If the artist realizes they made a mistake and are unable to undo enough steps, they can always load an earlier version of the model and take a different path. (Van Gumster, 2016)

The artist can use either version saving system that are already included in the program, Blender for example has the option to allow version saving, or they can use external programs. Different versions are usually numbered incrementally, for example, “mainCharacter_02”. Some people may also name different versions by adding the date it was saved on to the end of the file name.

4 THE BASICS OF BLENDER

Blender is an open source 3D modeling program which can be downloaded for free at www.Blender.org. The program is free to use even for commercial projects and can be used for every aspect of 3D modeling from the actual modeling to sculpting, texturing and rigging. Blender is though quite different from most other 3D programs due to its high customizability and reliance on hotkeys for most tasks. This means the program can have a steep learning curve when someone with history with other programs decides to give it a try. This chapter will very briefly explain the very basics of Blender for those who are unfamiliar with its functions.

4.1 Setup

Before beginning the actual 3D modeling process, there are a few things that one might want to change in the Blender user preferences window. The user preferences can be accessed from the File-tab or by using the “Ctrl Alt U” shortcut. The first thing to change is the history states, or global undo. These are the amount of times the artist can undo their steps and by default this number is quite low. The global undo can be altered in the bottom-left of the Editing-tab of Blender User Preferences window. (Image 9)

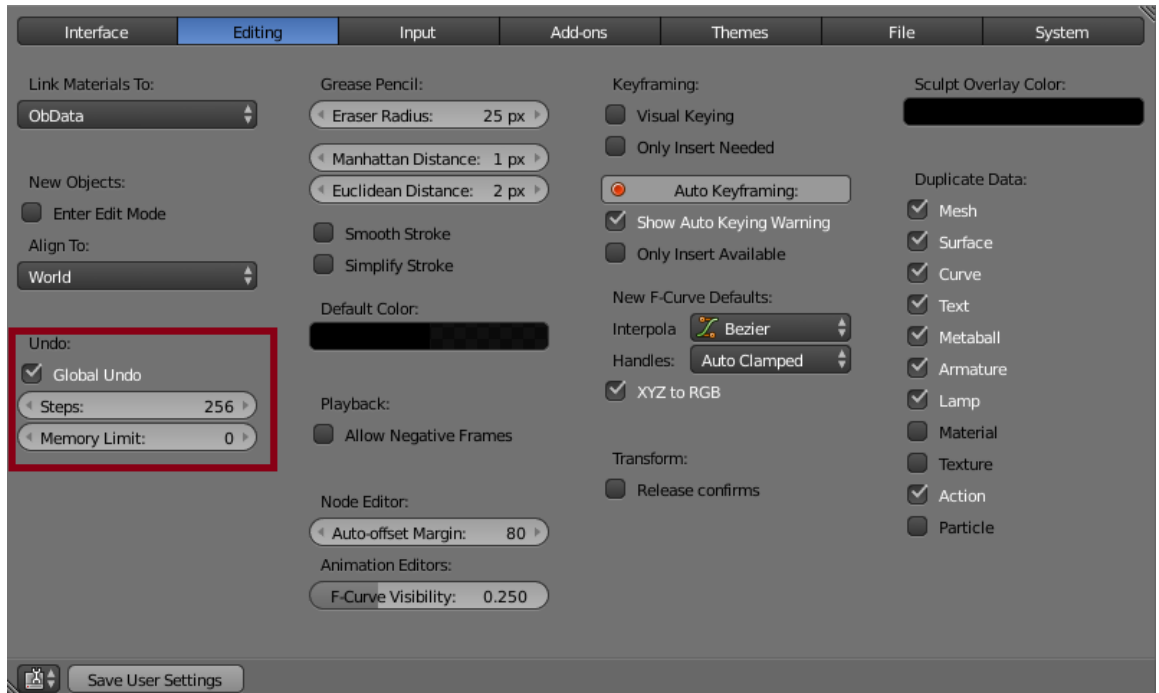


Image 9: The Editing-tab of Blender User Preferences. The Global Undo slider is highlighted in red.

For someone whose first experience with 3D modeling was 3DS Max, Blender's default controls soon proved quite frustrating in how different they were. Luckily, it is possible to customize them in the User Preferences window. Whether one wishes to use Blender's own controls or change them to match 3DS Max's controls is their own choice. One should though remember, that if they decide to change any settings, they will need to make the same changes on every computer they work on from here on out. Writing down these guidelines somewhere where they can always find them easily is not a bad idea.

First, by default Blender uses the right mouse key to select things instead of the left mouse key which is used in most other programs, including 3DS Max. This can be changed in the Input tab of the Blender User Preferences. Secondly, Blender uses different controls to rotate and move the 3D view. These can also be accessed in the Input tab by selecting the following route in the drop-down menus on the right: 3D View → 3D View (Global). In the 3D View (Global) menu, the settings to change are titled "Rotate View" and "Move View". The key bindings can be changed by clicking the current key combination on the right side and pressing the desired keys on the keyboard. Image 10 highlights all the changes and shows

how the Input tab should look afterwards. The changes need to be saved by pressing the “Save User Preferences” button in the bottom-left or else none of them will be applied.

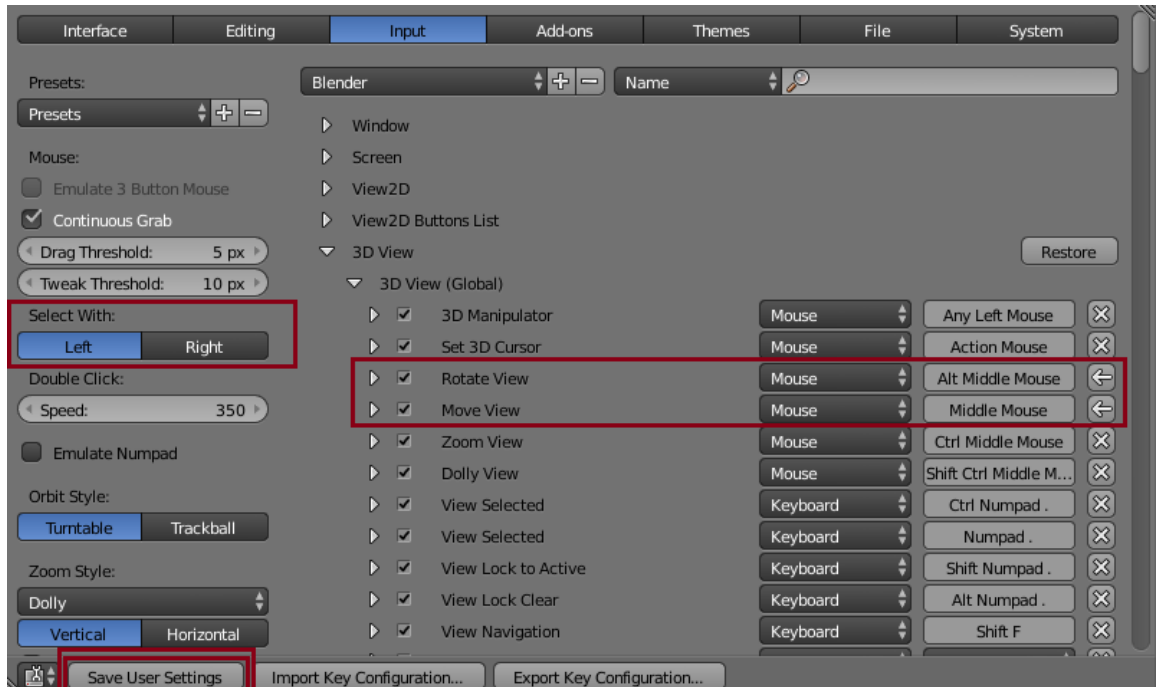


Image 10: The changes needed to make Blender controls more similar to 3DS Max.

4.2 The interface

It is recommended to open Blender and examine it first-hand instead of only relying on the images of this guide. There are several tabs, menus, windows and buttons for a modeler to examine which makes showing everything in still images difficult and redundant. Blender is also a highly customizable program which can look very different depending on who is using it. When Blender is first opened, the screen will look something like what can be seen in image 11.

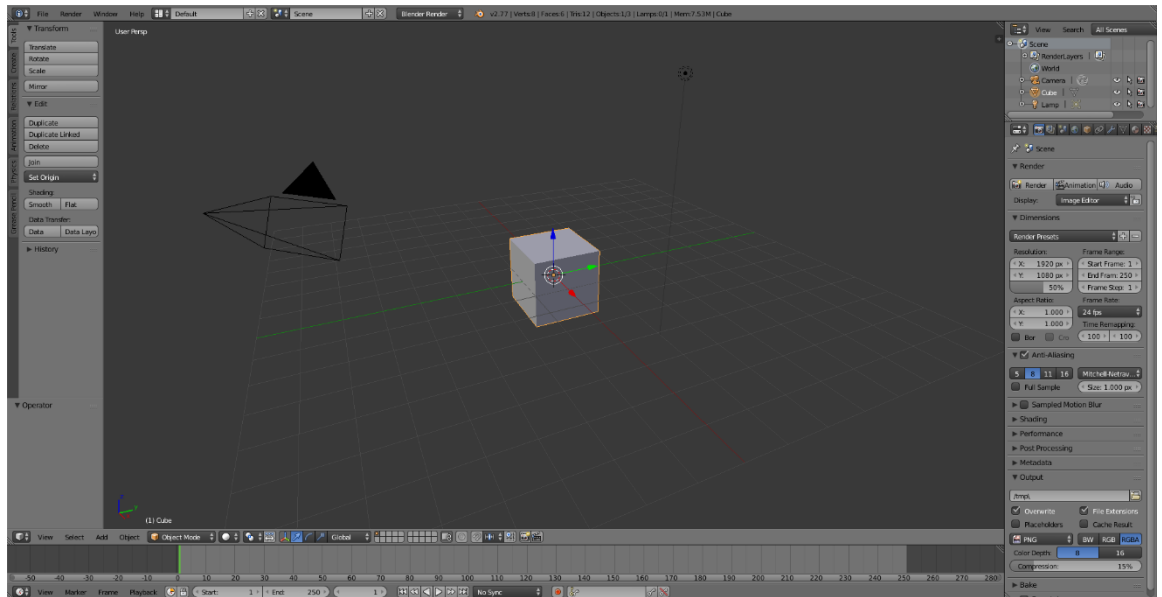






Image 11: The first look at the Blender interface.

On the left side, there are several tabs that can be used to create and edit objects. The tabs will be different depending on what mode the window, or viewport, is currently in. When first opened, the first viewport is in Object mode, as seen from the small menu at the bottom of the window (). In Object mode, the most important tabs usually are the “Create” and “Tools” tabs. In the Create tab, new objects can be created while in the Tools tab these objects can be edited, duplicated and joined together. By clicking on the button with the Object Mode text, the mode can be switched to Edit mode. While simple objects can be altered in Object mode, Edit mode allows a wider range of actions than can be performed on the object.

It is important to note that shapes created in Object mode will be separate objects with individual settings and names. Meanwhile, creating new shapes in Edit mode will add the shapes to an existing object and apply any settings and modifiers from that object to the new shape. For example, if an object has a Mirror modifier, creating a new sphere on one side of the scene in Edit mode will create an identical sphere on the other side, while creating a new sphere in Object mode will only create one sphere wherever the modeler wishes.

In Edit mode, the Tools tab has a wide range of tools to edit the initial shape with. Some of the tools are very self-explanatory while others can sound alien to a beginning 3D artist. The Extrude tool can be used to extend a selected face or edge to a chosen direction, the Inset tool can be used to in a way copy the shape of the selected faces inside the selection, while the subdivide tool can be used to split edges and faces in half. The Loop Cut and Slide tool can be used to add edge loops to wherever the modeler wishes. While the Loop Cut and Slide tool is active, the amount of created edge loops can be adjusted with the mouse scroll.

The various buttons around the Object Mode button () at the bottom also depend on what mode the window is currently in. The three first buttons will always be there while the rest will change. The  menu can be used to change the model's appearance on the screen. By default the model will be shown as a gray shape, but here the model can be shown to be displayed as a wireframe, textured model or even a rendered image. It should though be noted that this render is a preview and takes a while to load properly. Even slightly moving the view in Render view will cause the scene to be recalculated which will show as an eye-straining storm of pixel noise.

The  menu is where the modeler can select the current pivot point. By default, all changes will be done by the current selection's median point, but here the pivot can be set to, for example, individual origins which allows several connected polygons to be scaled individually rather than in relation to each other. To demonstrate this, let us look at a cube with three subdivisions. Subdivisions can be added with the "Subdivide" tool and the top faces can be extruded out with the "Extrude Individual" tool in the left toolbar. This will extrude the faces so they are separate towers even if they appear to be an even face. Scaling these faces by the median point or by individual origins will create very different results as can be seen in image 12.

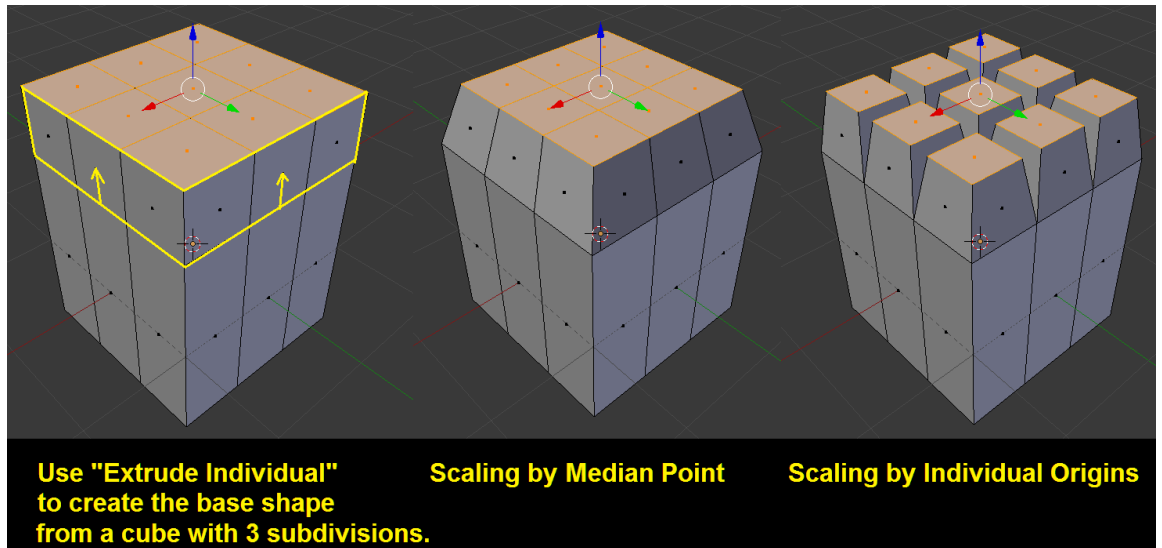







Image 12: The difference between scaling a group of extruded surfaces by median point and by individual origins.


The next buttons () will rarely be used in this guide as most of their features can be used with the use of hotkeys, but they can be used to change the Manipulator's (the Gizmo in 3DS Max) mode. The arrow lets the manipulator to be used to move the object, the arc lets the object be moved and the club-looking shape can be used to scale the object. The last drop-down menu offers different choices for the orientation of the Manipulator.



In Object mode, next to the Manipulator settings are two sets of squares ()). These squares represent different layers where objects can be placed. A layer with any sort of content will have an orange ball inside it. Several layers can be selected at the same time by holding Shift while selecting them. In Edit mode, the layer boxes are replaced with the following buttons, . These represent different ways of selecting parts of the 3D object. The first one will allow individual vertices to be selected, the second one will select edges between the vertices and the last one will select faces.

A more peculiar option though is the  button. This button will allow vertices, edges and faces to be seen and selected through the model. This button can be an invaluable tool when selecting areas with a lot of detail as it allows even faces hidden small crevices like a mouth or behind an ear be selected with ease.

Similarly, the tool can be a nuisance if it is left active when trying to focus on a specific area of a 3D model.

The last set of buttons to look at is the  group. These buttons represent the edge snap tools. When the magnet button is active, parts of the object can be moved in even increments. The menu next to the magnet where the constraints of the snap can be selected. For example, when the snap target is set to vertex, moving the selected vertices will always snap onto the vertex closest to wherever the selection is moved.

Moving away from the buttons of the toolbar, in the top-right corner is the Outliner which shows the structure of the scene. It will list out all objects created in Object mode ranging from cameras to actual shapes. The names of each object can be changed by double-clicking the name. Pressing the  button next to the name of an object will open that object in Edit mode. This is a quick way to switch between objects without needing to visit the Object mode in between. The eye symbol will hide and unhide the object, the arrow symbol will toggle whether the object can be selected and the camera symbol determines whether or not the object can be seen when rendering the scene. If an object suddenly disappears without reason in rendering, having accidentally clicked the camera symbol can be the reason.

The list of tabs under the Outliner is the Properties panel. Here the settings of a single object can be altered. Explaining all the tabs would take a very long time but luckily only two of them need to be explained for now. The  tab allows the addition of modifiers. When creating a 3D character, some notable modifiers are the Mirror modifier which allows the easy creation of symmetrical creatures and the Solidify modifier which can be used to give even thickness to pieces like clothes and armour. Meanwhile, the  tab is where materials can be created and assigned to an object. In the Material tab, the objects colour and texture can be changed as needed.

As mentioned, Blender's interface is very customizable. All of the windows explained can be changed into any mode from the small drop-down button at the left side of each window. The Outliner can be made into a 3D viewport, the Timeline can be made into a 3D viewport, even the Info tab at the top can be turned into a

3D viewport. The original 3D viewport can also be split into ten more viewports by pulling on the top-right corner (■). The options are nearly limitless, if not actually limitless. Extra viewports can be merged together by grabbing the corner again and pulling it towards a window that is the same width or height as the current one. Image 13 shows all these described viewports in action.

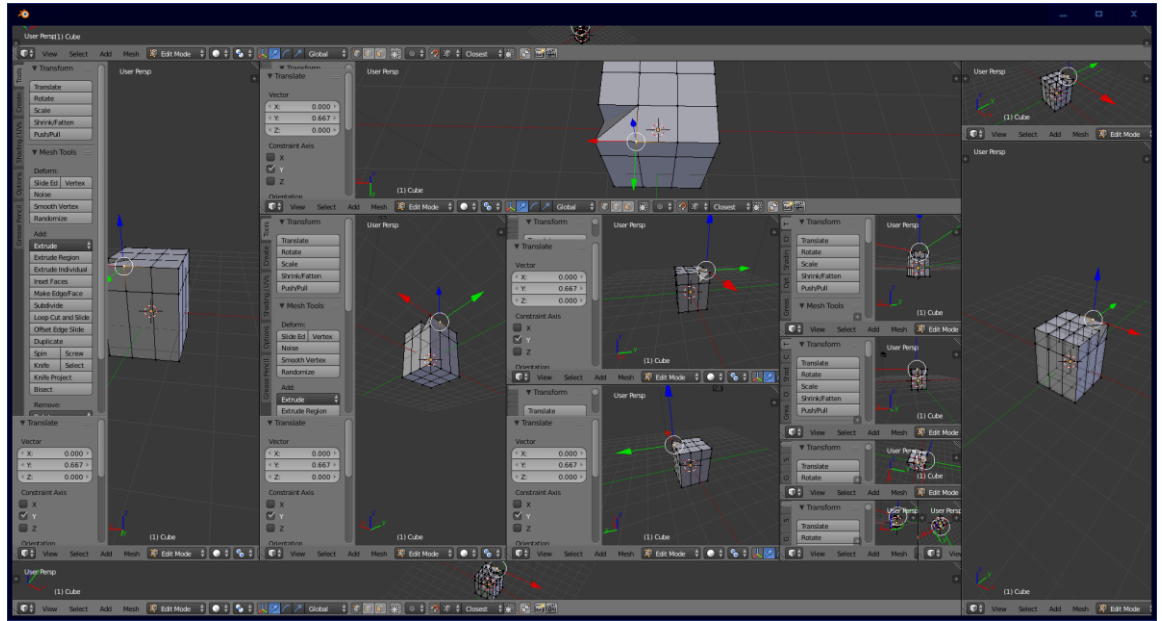


Image 13: Blender's interface split into a plethora of 3D Viewports.

As a last notion, the + symbol at the top-right of some windows can be used to open an extra settings menu. Depending on the type of window currently open, the contents of this menu vary.

4.3 Hotkeys

Blender is a program with a massive number of hotkeys. Many of the actions can be found as buttons, but most actions do not have a button. For example, creating a face between two edges or merging selected vertices are actions that do not have a button on screen. An alternate way to finding these actions is searching for it by pressing the space bar and writing the name of the command. Learning all the hotkeys by heart may seem like a daunting task at first, but the most used

commands will quickly become a second nature. Having a list of the hotkeys though can be useful, especially if taking a break from using Blender.

All the hotkeys used in the creation of the character model later in this thesis will be explained in the tutorial itself. Attachment 1 has a list and short description of all the hotkeys used in this project along with a few other hotkeys that could be useful elsewhere.

4.4 Modifiers

Modifiers are tools similar to filters in photography. Modifiers can be added and removed from objects as needed and they can be used to add different effects to the piece. When modeling characters, one of the most useful modifiers is the Mirror modifier. The Mirror modifier will copy all the topology on one side of the object's origin point along whichever axes the modeler wishes. The Mirror modifier makes it easy to create symmetrical characters and speeds up the process greatly.

Another useful modifier is the Solidify modifier. This modifier works best when creating objects like armour or clothing. The Solidify modifier will add thickness to any 2-dimensional plane and give it thickness. For example, the modeler can first create a long coat for the character out of flat planes and later use the Solidify modifier to make the jacket thicker. The Solidify modifier also ensures the piece is the same thickness everywhere. (Image 14)

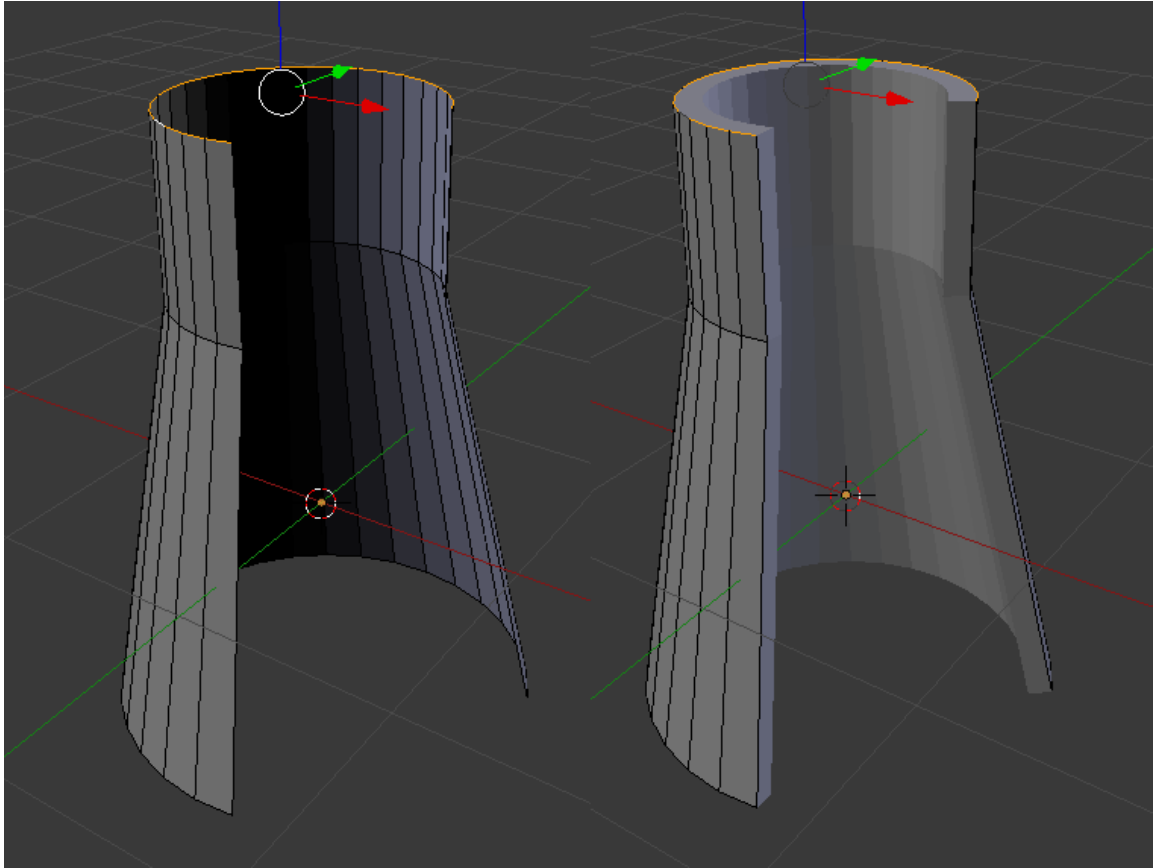


Image 14: How the Solidify modifier gives thickness to a flat object.

Some characters may have parts that need to be copied and pasted in an even line. For example, a fluffy tail could be created out of cones that are copied inside one another or the ridges of a dragon could be copied in a line along the creature's back. For such pieces, the Array modifier can prove useful. The Array modifier will copy the object as many times as needed into a set direction. It is also possible to have the Array follow a set shape like a curve.

There are several other modifiers, but their effects may be less useful when creating a simple game character. The Multiresolution and Subdivision Surface modifiers can be useful when sculpting a very detailed 3D character but they hardly are beginner-level features.

5 3D CHARACTER CREATION STEP BY STEP

This chapter will show the process of 3D character creation step by step. The first part shows how to draw the character's 3D model sheet and gives tips for keeping all the views consistent. The latter parts will focus on different parts of creating and preparing the 3D character model for use in video games.

The character featured as an example is not going to be used in a game but will be created in such a way that it could be used in a game like *The Elder Scrolls V: Skyrim*. The character will have all pieces of clothing modeled separately so they can be equipped and un-equipped as necessary. Overlapping pieces of clothing will have some polygons removed to make the model lighter. If it cannot be seen, it will not be rendered. The aim is to have the character's body consist of about 5,000 polygons while wearing any combination of clothing while the character's head should have about 1,000 polygons. This way the whole character model should have about 6,000 polygons.

The example character will be referred to by his name, Aroleir, through the course of this guide.

5.1 Creating the 3D model sheet

Before the artist begins creating a 3D model sheet, they should already know what kind of character they are creating. This goes both for the character's clothes and the level of realism the model would have. The T-pose used for a 3D model sheet strips the character of their personality which is why designing a character in this stage is not ideal. Polishing the character's design before creating the 3D model sheet also ensures that the artist will not need to share their focus between creating an interesting design and keeping all angles of the model sheet consistent.

One can also use a pre-made model sheet but the artist should remember to make sure they have permission to do so if they plan on publishing the model. It is also good to note that a pre-made 3D model sheet may have the exact same mistakes

that were mentioned in the earlier chapters and thus using them to make a 3D character might be more difficult than anticipated.

5.1.1 The front and back views

The first stage of creating a 3D model sheet is drawing the nude character's front view without anything extra. This makes figuring out anatomy easier. Even if the character would never be shown without their clothes in the game, the nude references make figuring out topology and joints easier in the 3D modeling phase. The front view is a good starting point as the back view can later be traced over it and the side views can use the front and back views as points of reference.

It is up to the artist to decide whether they wish to have an unclothed version of the character from all directions or not. Front and side views can be enough reference for creating the naked character model for a humanoid character. Aroleir's front and side references were first drawn nude, but the back view was drawn only after all the clothes of the front view were done as time was limited.

If the character is meant to be heavily stylized, the artist can make their own decisions on the character's proportions. Meanwhile when creating a realistic character, finding reference material is more crucial. One should remember that in full body photos you rarely see an orthographic projection of the person. Especially feet often are shown slightly from above in drawings and photos of people. In a 3D model sheet, the feet should though be drawn directly from the front as shown in image 15.



Image 15: Orthographic view of feet and shoes.

The character's pose in a 3D model sheet is very neutral and there are several graphs that show the proportions of a human body. The first step of drawing the character would be figuring out their height in comparison to their head. This height varies greatly depending on the style and species of the character, but in Aroleir's case the character's height was set to be $7\frac{1}{2}$ times the size of his head. Anything less than this would give the character a cartoon-like feeling while anything taller might take him towards an idealized style like in superhero comics. Image 16 shows a few examples of men with different body proportions.

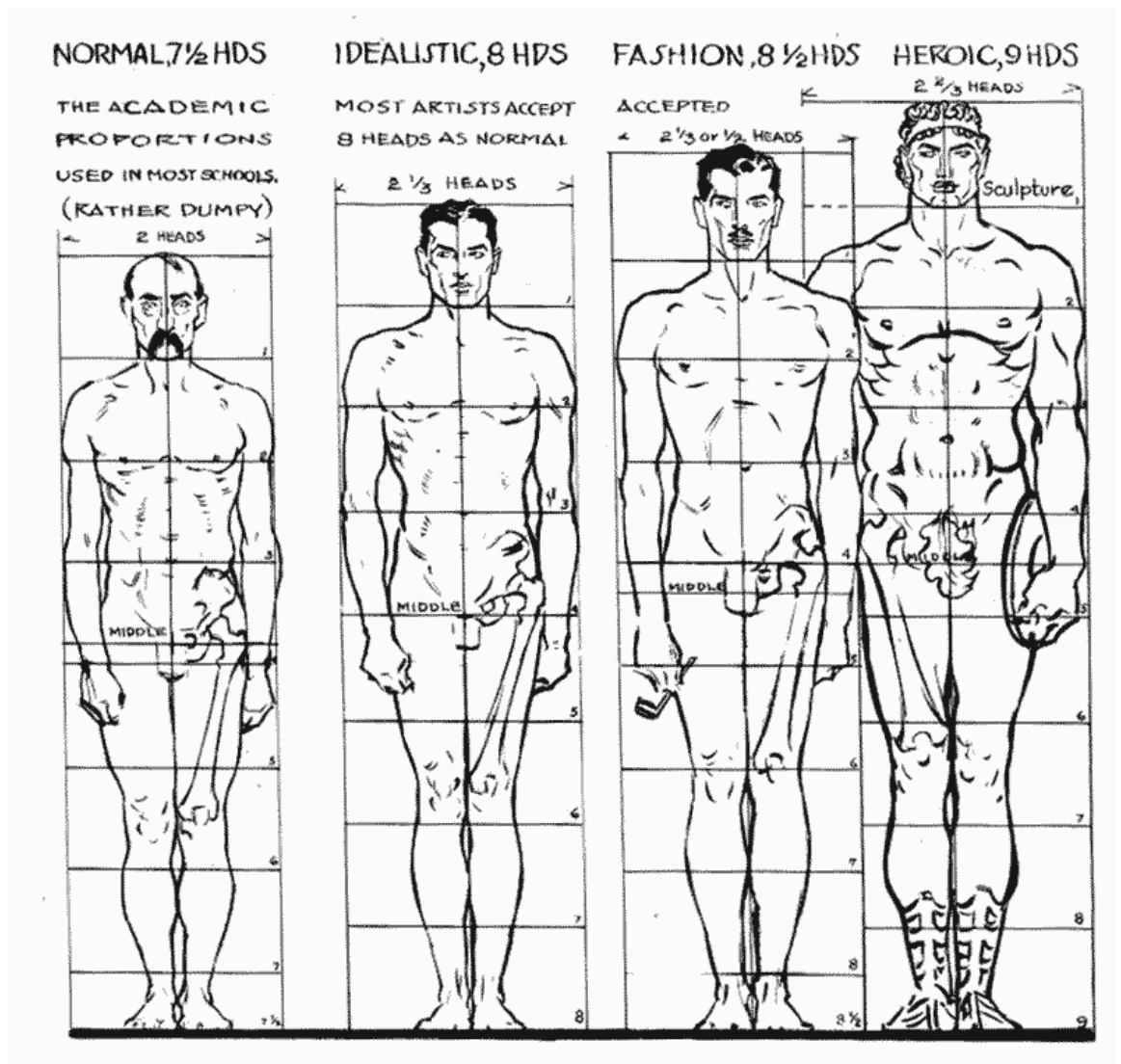


Image 16: Different ways to draw a man. (Andrew Loomis)

A scale for drawing the character can be created quickly by drawing an egg shape that represents the character's head and copying it as many times as needed. This

scale can then be used to block out the basic shape of the body per the artist's vision for the character. Aroleir is meant to be fit and toned, but he does not need the hourglass figure and pronounced muscles of a superhero. Image 17 shows the very rough sketch of his proportions alongside the finished lines.

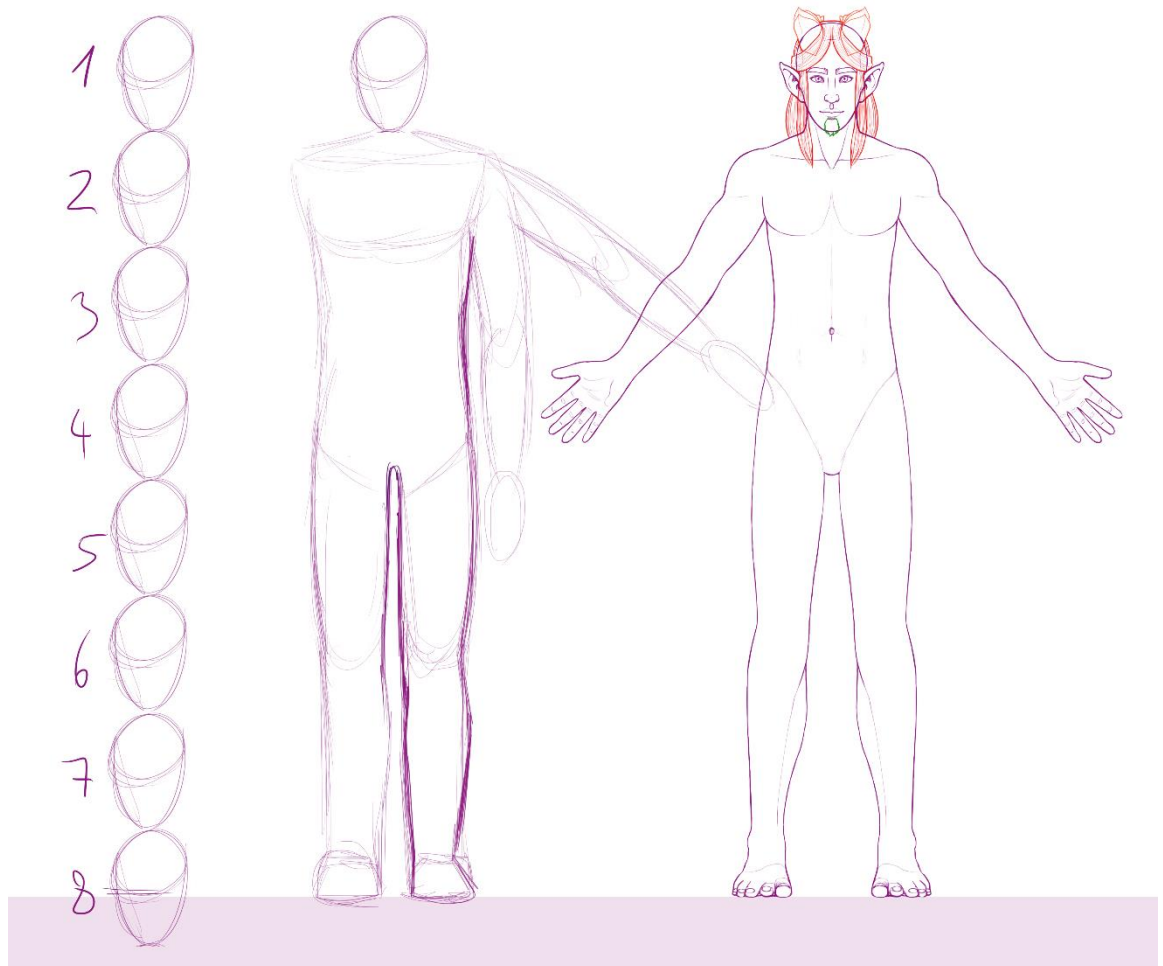


Image 17: Body sketch and finished lines.

The artist can and should speed up the drawing process by first drawing only half of the character and mirroring it if the character has parts that are symmetrical. After all, the 3D artist will most likely also use mirroring tools to speed up their work if the character's design allows it. If the references would not be mirrored like this, it would be highly likely that the halves would not be perfectly aligned. This would be visible in the 3D modeling phase, as one of the reference's halves would be slightly off. While this would not be much of a problem, it would still be an annoyance that could easily be avoided. Another point to this is the fact that perfectly

symmetrical faces can easily look strange. If the references are mirrored, the artist can in that phase already try to make the face's proportions look pleasant despite the perfectly symmetrical features and thus make the 3D artist's work a bit easier.

When the clothes are being drawn, it is a good idea to place them on a separate layer so the nude version can also be given to the 3D artist. The nude version helps the modeler tell where the character's joints should be even if their clothes would be loose-fitting. It also usually is easier to add the clothes after the base topology of the model is done. If the game has several characters, the nude character model can also be edited to make different characters quickly. In Aroleir's case, every piece of clothing was drawn on a separate layer in the front and side views due to the way he should be able to wear different clothing combinations. The back view though was drawn fully clothed as the side view gives enough information about how his clothes work on the back. Attachment 2 shows the finished reference sheet with different combinations of clothing.

Once the front view of the character is finished, with or without clothes, creating the backside is a straight-forward process. First, the line art of the front view is copied and mirrored. By deleting all the lines within the outline, we get a base to draw the back view around. This way the silhouette remains the same in both front and back views. Certain details that continue from the front to the back, like edges of clothes, belts and such, can be marked on the back view and Image 18 shows the stages of this method.

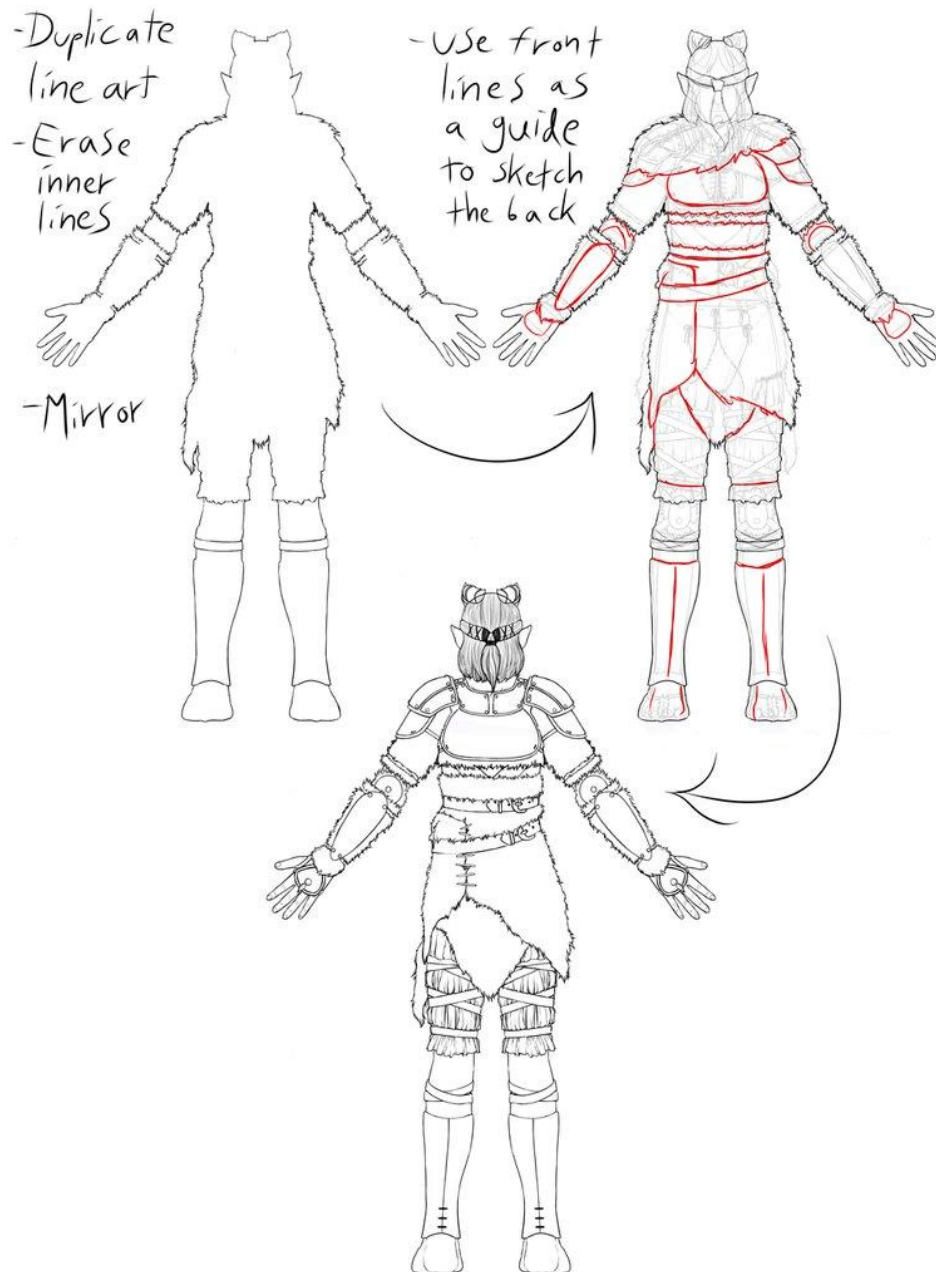


Image 18: Creating the back view with the help of the front view.

5.1.2 The sides and other views

After the front and back views are done, creating the side view is just a matter of combining the information of the two references. The front and back views are to be placed on opposite sides of the canvas with enough space in between for the side view. To keep the side view consistent with the other two, straight horizontal

lines are to be used to connect matching parts in the front and back views. In certain drawing programs, like Photoshop, these drawn lines can be created by using the built-in ruler tools. If the used program does not have rulers, another way to do this is to use the rectangle selection tool to mark the area where a certain part is to fit. It is also possible to draw straight horizontal lines on a separate layer. Regardless of which method is used, it is easier to make sure every detail lines up with the other views when using these guidelines. Image 19 shows an example of how the rectangle selection tool was used in drawing Aroleir's side reference.

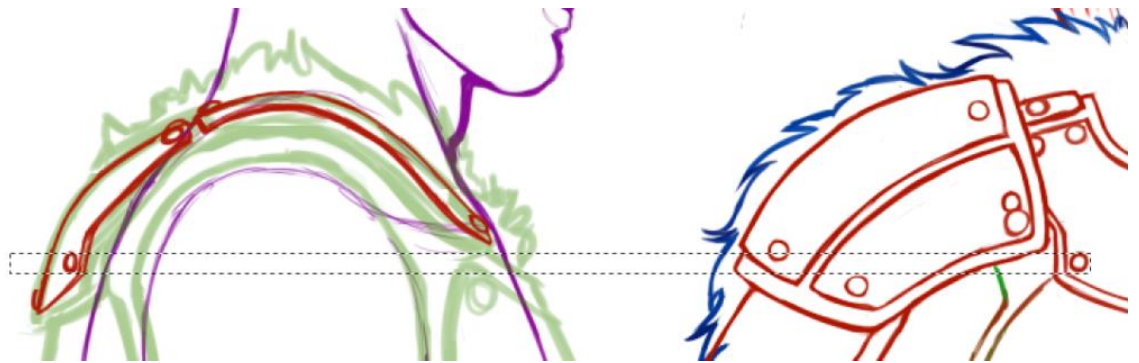


Image 19: The use of rectangle selection tool in positioning the studs of his armour.

When creating the side view, it is a good idea to either draw the arm separately or leave it out completely. Depending on whether the character's arms are extended straight to the sides or at a slight angle, drawing them on the side view is either useless or harmful. If the arms are at a slight downward angle, the arm will hide a part of the character's sides. The downward angle in tandem with the orthographic view also makes the hand's size and the arm's proportions quite tricky to figure out. This is because the artist cannot simply use the character's head's height to measure the arm's length as they could in the front view. The artist could though draw the arm straight from the side and the 3D artist could model the arm separately and attach it to the body when the limb is ready. In Aroleir's case, the arm was not drawn from the side at all as the front and back views showed the details of his clothes well enough.

One of the greatest challenges of creating the side view is staying true to the orthographic view. If the character has for example a belt that hangs loosely at an angle, it can be difficult to figure out how it would look from the side. This is where using the horizontal guidelines is especially useful even if tedious. In Aroleir's

case, this problem was apparent when creating the cut pieces of fur around his waist. From the front and back the shape was nothing special, but in the side view it took a few tries to find a shape that looked good. Image 20 shows a few examples of how the furs could have been drawn while keeping the reference true to the other images.



Image 20: Different shapes that are all plausible. The middle one is the final shape.

5.1.3 Bringing the references to Blender

Before the character references can be brought to Blender, they need to be saved on separate files. All the references should be placed in the middle of the file and aligned to each other to make bringing them to Blender as easy as possible. This can be done by first placing all the references on top of each other in a 1:1 square shaped file, saving a copy of it for each view and deleting the extra layers from each file. This way there will be a separate file for each view while also having all the views be on the same spot on the square file.

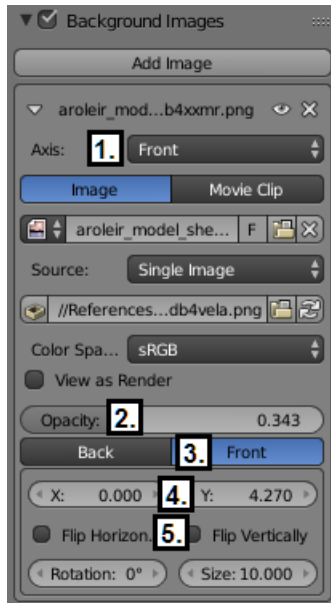
Once the references are saved as PNG files, they can be brought to Blender by using the Background Image feature. On the very bottom of the right sidebar of

the 3D Viewport there is a tab titled “Background images”. By ticking the box and clicking the “Add image” button, you can select a reference picture from your computer. After the image has been imported, by default it will show in all main views (Front, back, left, right, top, bottom). The images are automatically hidden when the character is rotated so they will not clutter the viewport. It should be noted that the background images only show in orthographic view (Numpad 5) as in perspective view they would be inaccurate. A single view can be selected in the drop-down menu titled “Axis” (Image 21, point 1).

By default, the images are displayed behind the model. This means they cannot be seen through the model which defeats the purpose of the reference images. To see the images through the model, the images should be set to be transparent (Image 21, point 2.) and displayed on the front of the viewport rather than the back (Image 21, point 3). This way the model can be seen through the reference images.

The images are placed in the middle of the scene so the base grid crosses over the character. This can make creating the character more difficult as then the center of the grid cannot be used as a point of reference when positioning different parts of the character’s body. The reference images can be moved higher in the scene by adjusting the Y value near the bottom of the background image settings (Image 21, point 4.). The reference should be placed so that the character’s feet rest on the grid. This way the character’s origin point will be between their feet which makes adding him to the game easier. The value slider changes in increments of 0.1 which can be too inaccurate. The value can be set manually in more detail by clicking on the number.

In Aroleir’s case, his side reference was only drawn of the left side but it would make matters easier to be able to see the reference from both sides. To do this, the same image was applied twice and was set to be displayed on both left and right side. In the right side, though, the reference had to be mirrored so he would face the same way as on the left side. (Image 21, point 5.)



1. Choose what angles the images are shown from.
- Change the reference file.
- Update the reference file if there have been changes.
2. Change the opacity of the reference image.
3. Set the image to be in front of the model so the references can be seen through it.
4. Adjust the position of the image. X for horizontal alignment and Y for vertical.
5. One of the side views needs to be flipped so the character on it will look in the same direction as all the other views.

Image 21: The settings for the front background image. Here the settings are shown as they were when making the model for Aroleir.

5.2 Creating the 3D Model

The modeling process begins from the character's torso as it gives the base for the model's level of detail and overall structure. Creating a cube is a good start as it is easy to add edge loops to such a simple shape. The cube is resized so it is about the size of the character's chest and in Edit-mode it is moved to the height where the character's chest should be. The move is done in Edit-mode because if the object would be moved in Object-mode, the model's origin point would move with it. In Edit-mode, the object's origin remains at the middle of scene. If the origin point would move elsewhere, it would later cause problems in the use of mirroring tools and in bringing the character to a game.

If the origin has moved from its place by accident, it can be returned back to its intended place by first moving the 3D Cursor to the middle of the scene (Shift + S → Cursor to Center) and then moving the Origin to the 3D Cursor (Shift + Ctrl + Alt + C → Origin to 3D Cursor). The origin can only be moved in Object-mode.

Before beginning to edit the cube, the “Limit Selection to Visible” (📐) setting should be turned off. The button can be found in the bottom toolbar in Edit-mode. When this button is turned off, vertices can be seen and selected without the surface of the model obscuring the view. Basically, this shows the model’s full wireframe through the surface polygons regardless of what angle the model is viewed from.

The first change to the cube is setting it up for the Mirror modifier. A vertical edge loop is added to the middle of the cube with the “Loop Cut and Slide” tool. This edge loop marks the character’s middle line. Next, all the polygons on one side of the middle line are deleted, leaving one half of the cube. This is when a Mirror modifier can be added. This modifier copies all changes done to the model in real time, which means only one half of the character needs to be modeled by hand. The Axis of the modifier should be set to X which is the default setting. (Image 22)

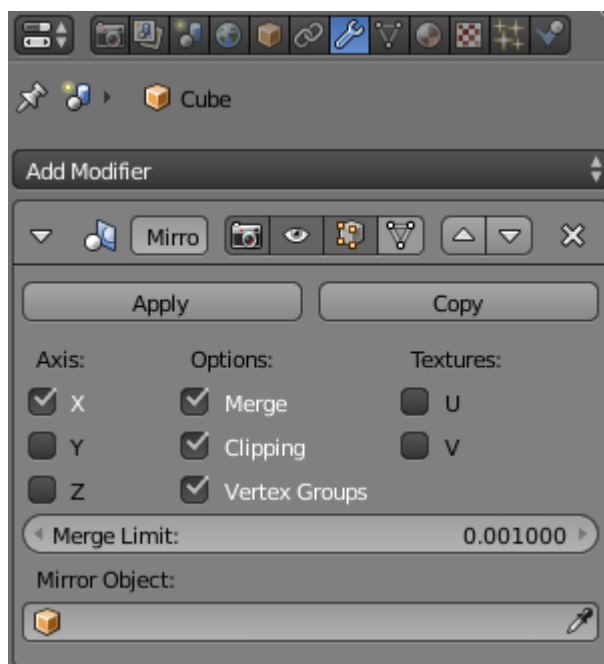


Image 22: The final settings of the Mirror modifier.

The Clipping option is disabled by default but it should be enabled most of the time when creating the character. The clipping option makes sure the middle line created before stays where it should and prevents vertices from moving to the other side. If vertices would end up on the wrong side, the mirror modifier would copy

them twice and create overlapping faces. (Image 23) The Clipping option can occasionally be disabled for a moment if the character model requires separating vertices on the middle line.

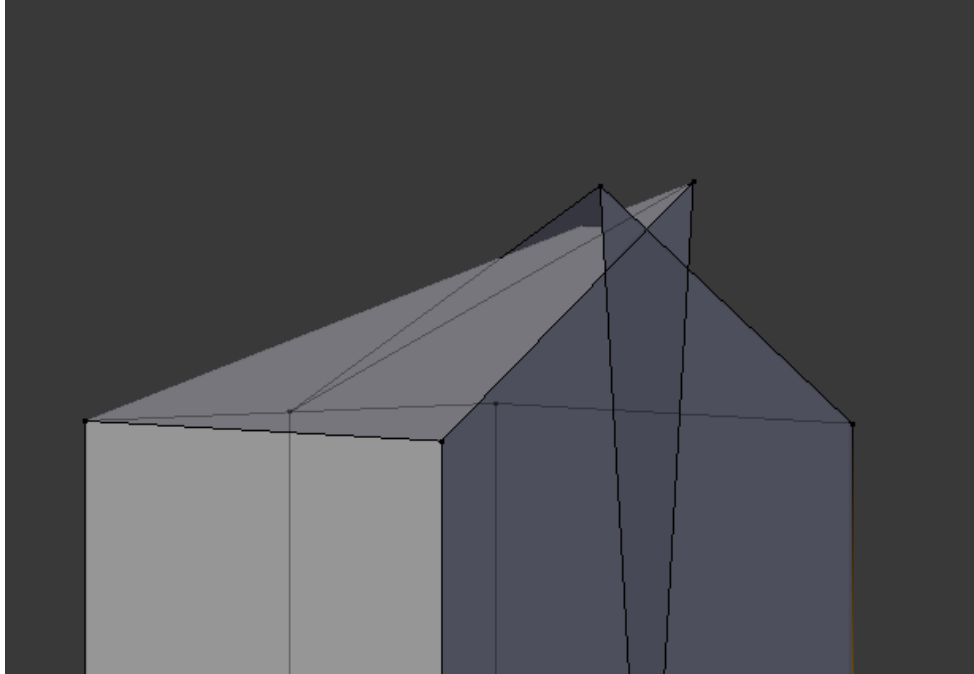


Image 23: An example of troublesome topology that may occur if the Clipping setting is turned off when modeling. The highest vertex has slipped to the wrong side and thus created overlapping faces and the clean middle line is lost.

5.2.1 The torso

After the Mirror modifier has been prepared, the process of turning the cube to the character's body begins. The Box Selection tool (B) is used to select and move the cube's top vertices where the character's shoulders will be and the lower vertices are taken to where the character's hips will be. This is done from both the front and the side views. Next, horizontal edge loops are added to the shape and the created vertices are scaled so they line up with the character's waist's silhouette in the reference images. Once the horizontal edge loops are in good places, vertical edge loops are added to both the model's front and sides. Image 24 shows the added edge loops without any changes to the model's silhouette to show how

many edge loops were added on each side. When crating the actual model, it is advised to not add these loops all at once as they clutter up the views fast.

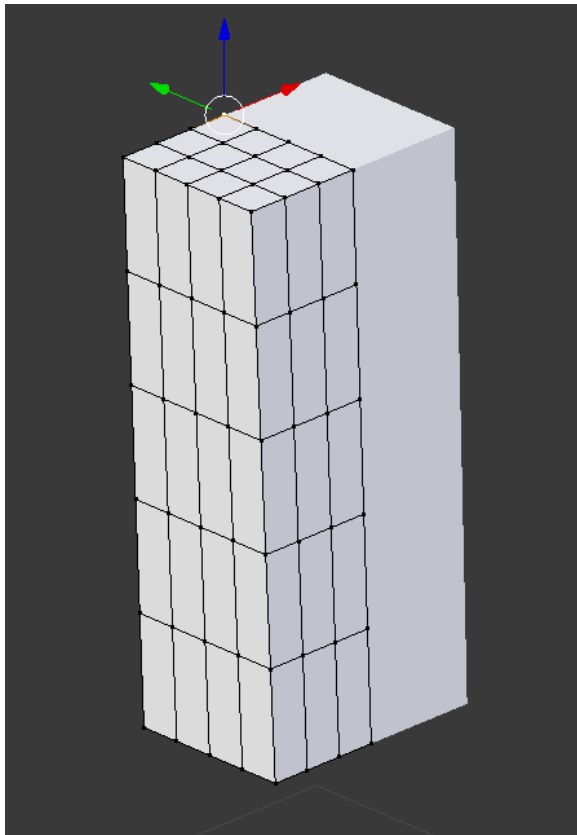


Image 24: In the beginning, four horizontal edge loops were added. After them, two edge loops were added to the Y axis (front-back) and three were added to the X axis (right-left). More edge loops can be added as the need arises.

This phase is mostly about moving vertices around and trying to create the silhouette for the character's body from the front and the side. The views can be changed by pressing the Numpad keys 1 and 3. The aim is to find balance between softening the character's features and being mindful of the game engine's limitations. This job can be made easier by selecting several vertices and using the scaling tool (S) to round out the shapes, and by having the Edge Slide tool move vertices along the existing edges by pressing the G-key twice after selecting the vertices.

Once the shapes have been matched to the front and side references, the model seems quite competent already. However, when the view is rotated to any other view, the model turns out to have right angles instead of rounded shapes. Before rounding the corners, this is a good place to create holes for the character's arms

and neck. The main thing to remember here is that the arm holes should have an even number of vertices and the removed polygons should be chosen so the vertices around the hole are easy to move into a round formation. In Aroleir's case, four polygons were removed from the side (arms) and the top (neck). The arm and neck holes were then rounded to match the references. At this point the holes can be very rough. Image 25 and 26 show two ways of creating the arm hole. The first one shows the arm hole on Aroleir, while the second one features an earlier character project.

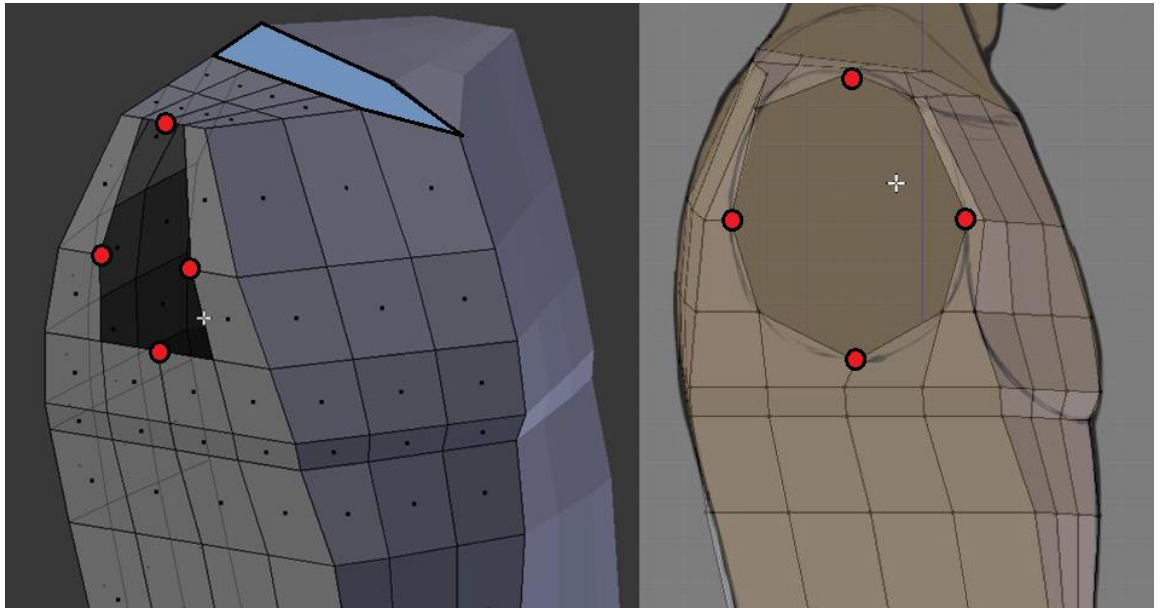


Image 25: The creation of the arm hole on Aroleir. The faces marked with blue were also deleted to create the hole for his neck.

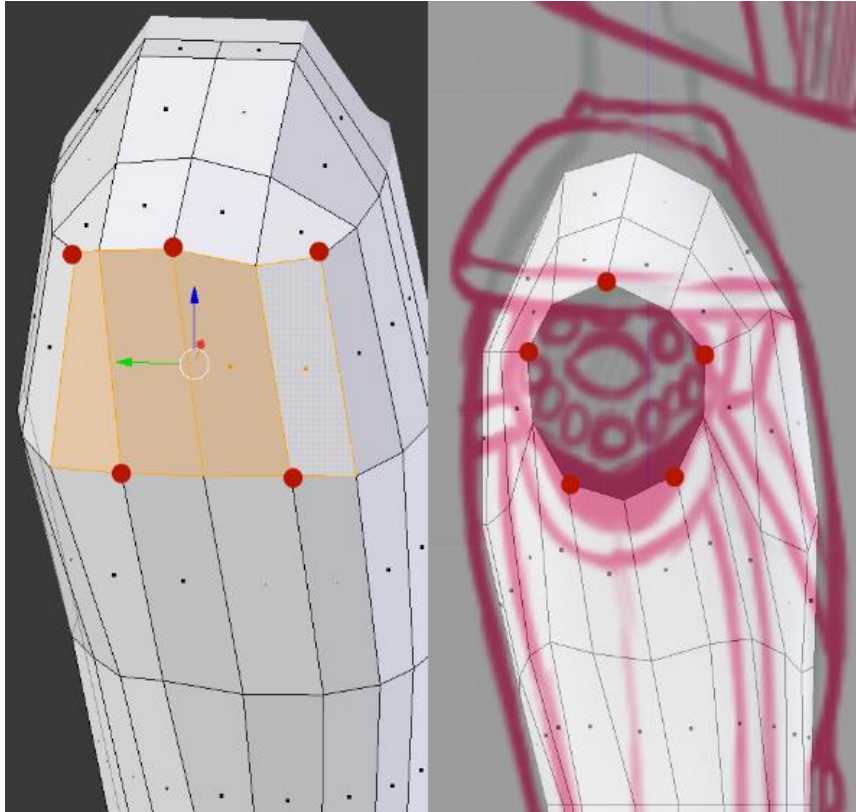


Image 26: This character was much lower poly than Aroleir and even a peculiar solution like this worked well.

When it comes to rounding out the character's sides, there should be one edge loop on both the front and the side that won't be touched at all. This will be the loop that determines the character's silhouette as shaped before. When creating a realistic character, these silhouette-determining loops are usually the middle loop or the one behind it, while on the front and back views this may be the loop next to the middle line. After all, the human body usually has a slight dip between the shoulder blades and chest muscles. On a more cartoony character, the silhouette-determining loops may be the very middle loops as a small detail like the dip between shoulder blades is not necessary.

Once the silhouette-determining loops have been selected, all the other edge loops will be moved inwards to get rid of any angles that stick out. The first edge to change should be the very corner of the shape. All the vertices of that one edge should be selected and moved along the Y and X axis so they would make a smooth curve with the two edge loops next to them. Next, one edge on both sides of the currently selected edge are selected and moved again to make a smooth

curve. This will be repeated until every vertex except the silhouette-determining ones have been moved and the shapes have been smoothed out. This goes also for the top vertices that would become the character's shoulders. (Image 27)

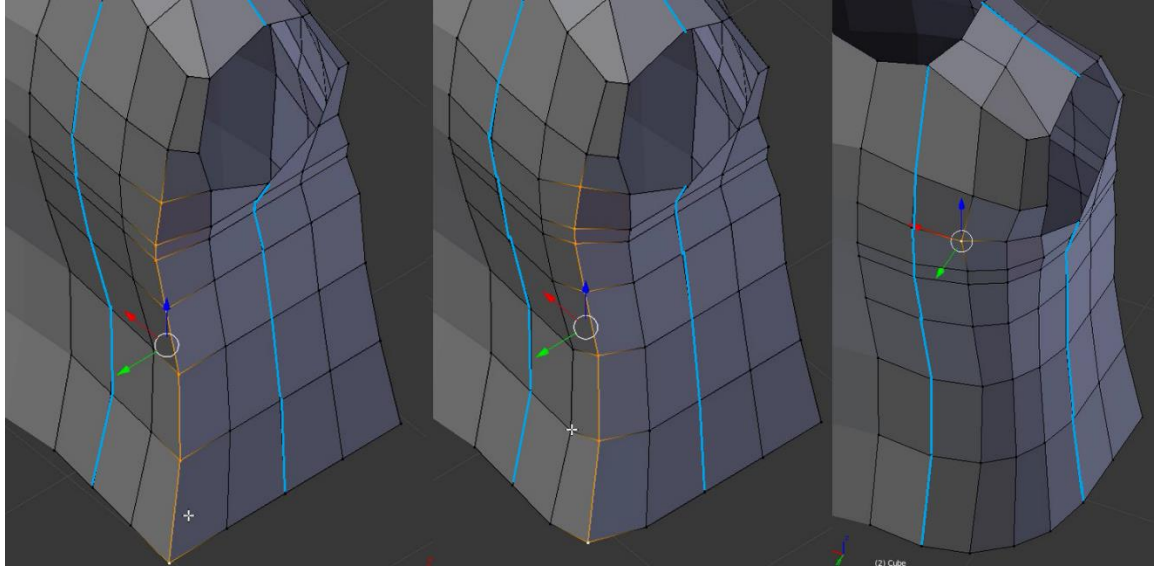


Image 27: Moving the vertices of the corner to smooth out the shape. The silhouette-determining edges that were not moved in this phase are marked with blue.

When creating the front of the torso of a more realistic character, the loops can be edited so they follow natural muscle lines of the body. Namely, this means curving the top loops so they form the shape of chest muscles. If done right, this will ensure the chest muscles will move naturally when the character moves their arms and the shape of the chest will be obvious even without textures. (Image 28)

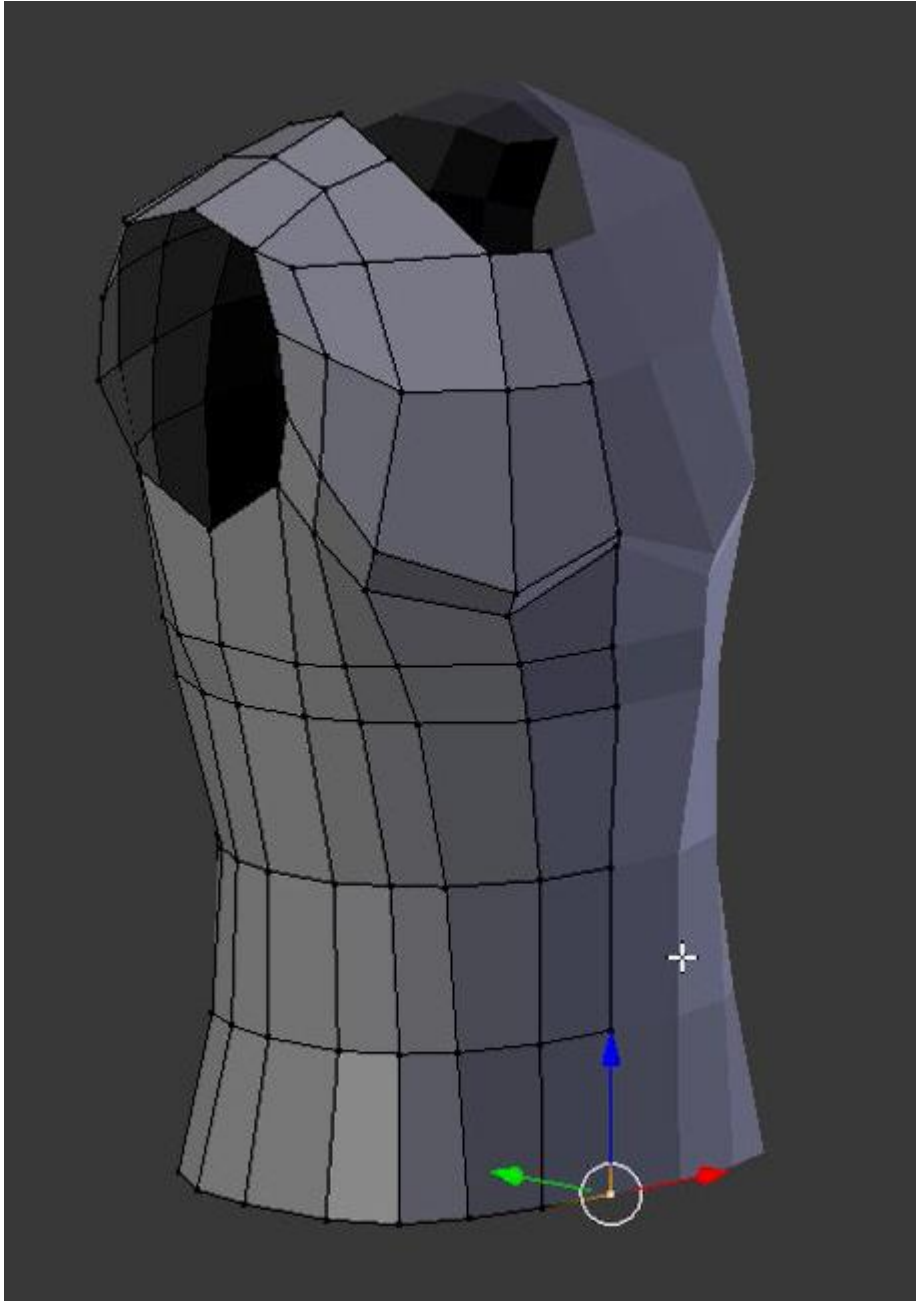


Image 28: The finished torso. It should be noted that one of the edge loops around the rib cage was removed later in the project as it was not needed after all.

Because a character's arms are usually down and in movement the shoulders are under a lot of stress, adding extra polygons to the shoulders will prove useful. The shoulders can be given a more natural shape by selecting the second highest edges of the arm hole and creating faces between them with the Face tool (F). The created face will be split into four with the "Loop Cut and Slide" tool. This will leave behind a triangle-shaped hole with 6 vertices. The edges of this hole are

selected and a face is created in the hole's place again with the Face tool. The new polygon will then be split into two quads by choosing the middle vertices and creating an edge between them with the Join tool (J). Image 29 demonstrates the steps of this action.

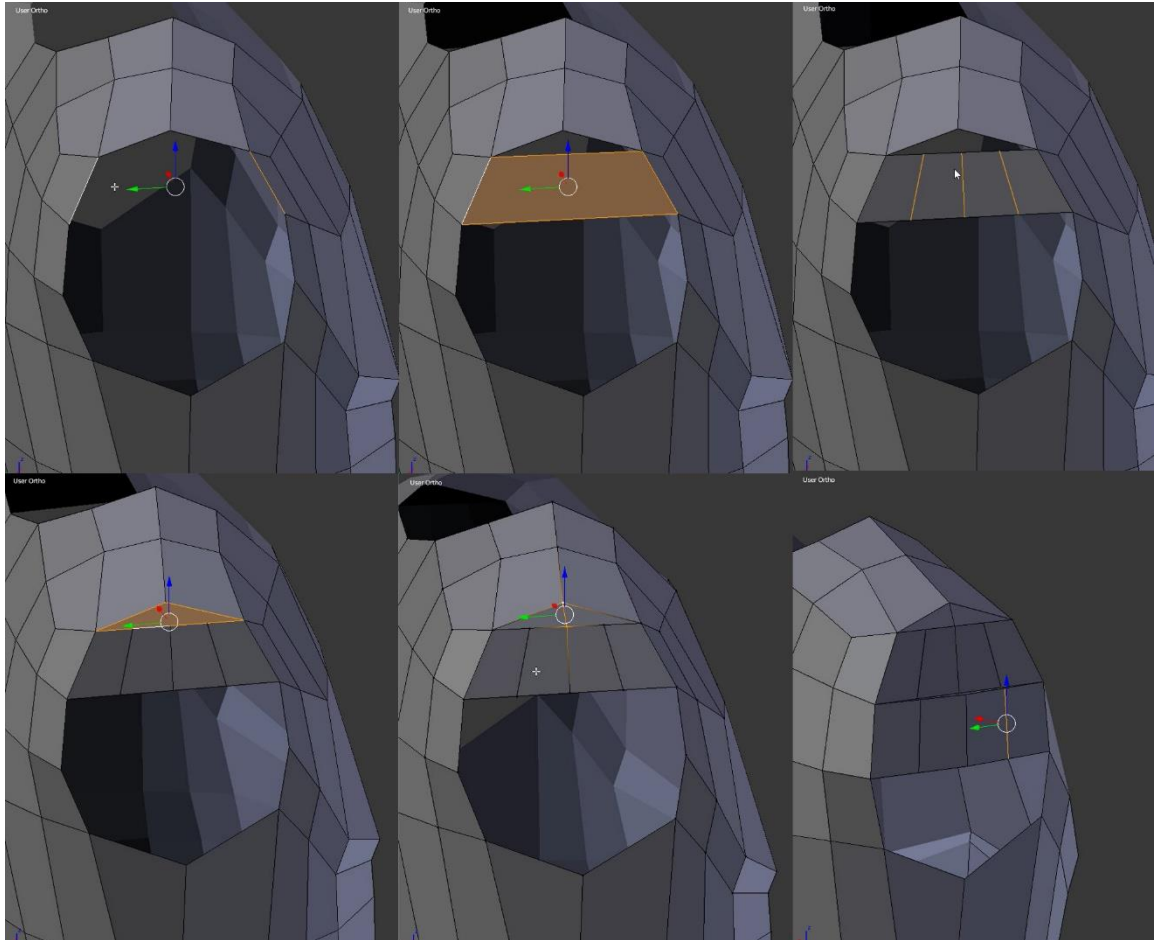


Image 29: Adding more polygons for the shoulders. In the last image, an extra set of polygons was added as it lined up with the muscles on the reference images better.

Once the new polygons have been created, their vertices need to be moved so they match the references. The vertices of the middle line are selected and moved along the X axis to match the character's silhouette from the front. After that, the vertices next to them are moved so they form a smooth, round shape. The vertices on both side of the middle line can then be moved around at the same time and scaled together to get the curve even. If needed, more rows of polygons and edge loops can be added to the shoulder. (Image 30)

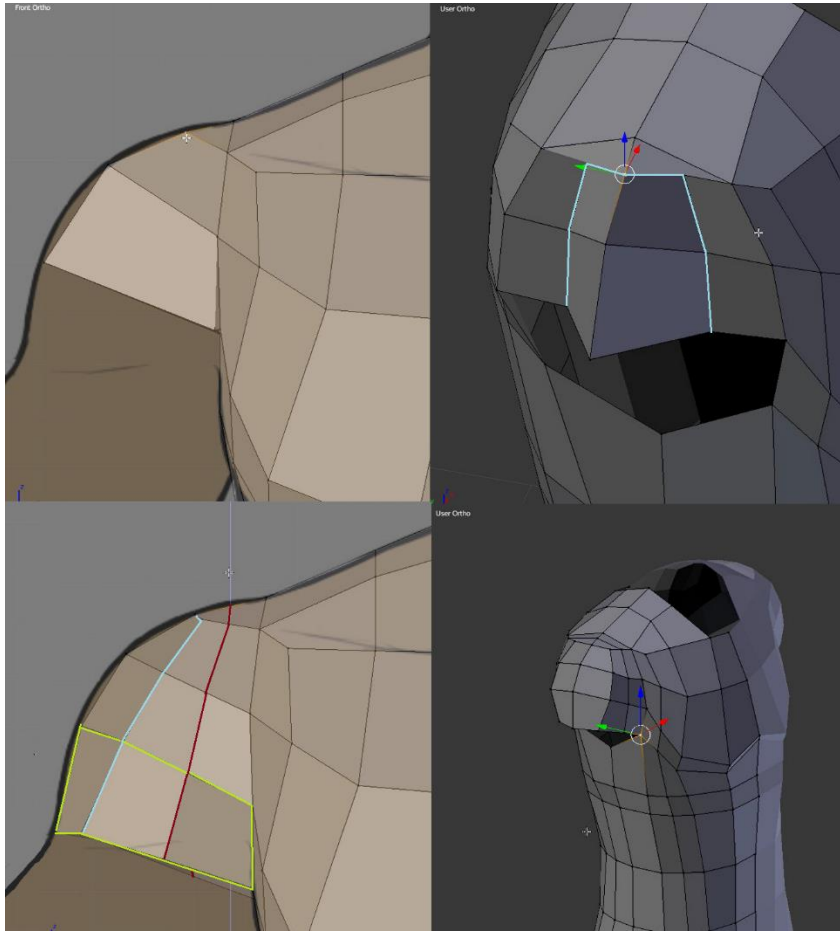


Image 30: Shaping the shoulders. The green lines show an extra row of polygons that was added after shaping the top of the shoulder. Red lines show an extra edge loop that was added to help make the shape smoother.

Next, the character's lower body is prepared the same way the shoulders were. An equal amount of edges are selected from the character's front and back and faces are created between them with the Face tool (F). The amount of edges chosen depends on how detailed the character's crotch needs to be. In Aroleir's case, one edge was selected. This edge was later on split into two but at this point just one edge was easier to work with. The face created between the edges are then Subdivided (S) into two and the new edge is pulled down to the level of the character's crotch. Image 31 shows the steps of this process.

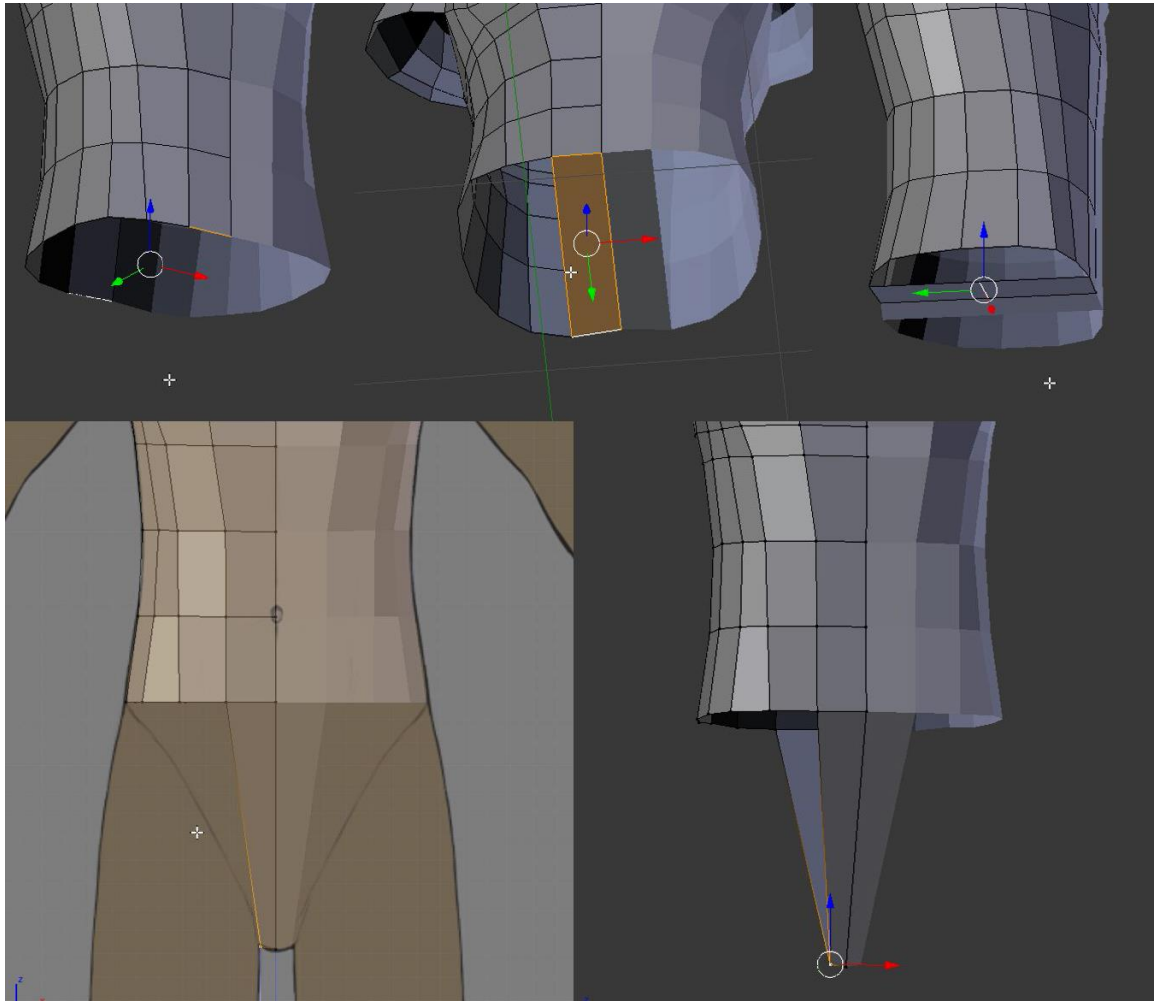


Image 31: The beginnings of creating the character's crotch.

The lower body at this time resembles more a spike than a crotch. The created edges need to be subdivided further and the new vertices to be moved into a more rounded shape. It should be remembered that this middle line will be the dip between the character's buttocks, not the highest point of them. For Aroleir, two edge loops were added to both the front and the back of the crotch strip. (Image 32)

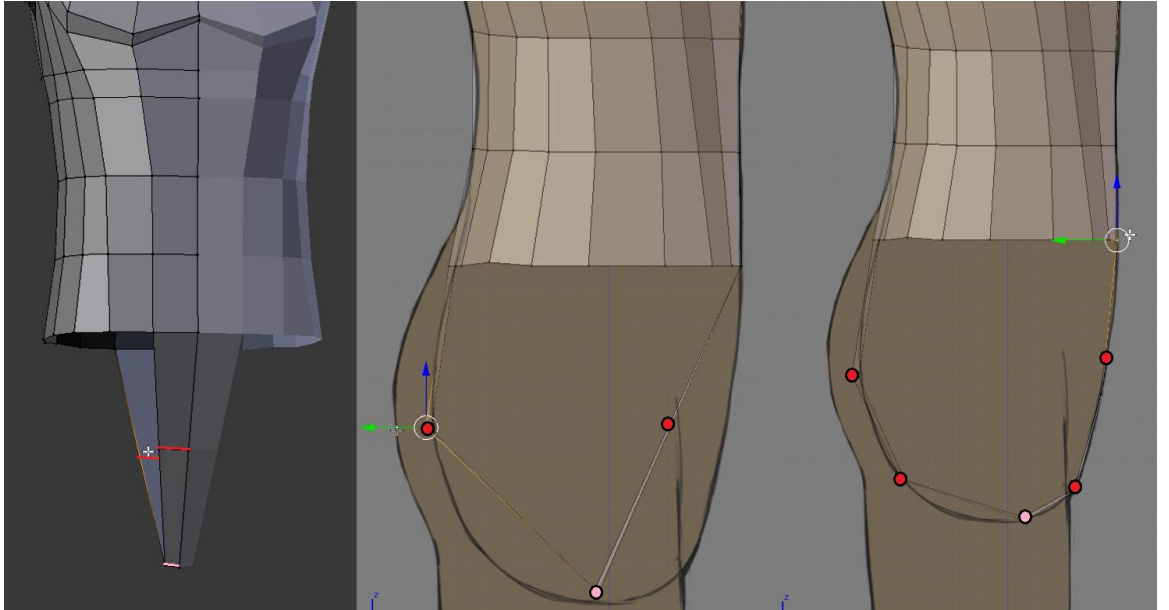


Image 32: Refining the crotch's silhouette. Red lines and dots represent newly added edges while the pink ones represent the edge added in the last step.

In the next phase the modeler needs to decide how many vertices they want the character's legs to have. Like with the arms, the legs should have an even number of vertices and thus the leg holes here should be the same. At this point Aroleir's leg holes had 14 vertices which I found to be a bit too high. The number of vertices can be lessened without removing any edge loops by extending down the character's torso like demonstrated in image 33. The two highest edges are joined together with a face. The face is then subdivided by the number of vertices that the character's side has and the created vertices are joined to the side with the "Alt + M" hotkey. The silhouette will then be adjusted to match the references. This method lowers the amount of vertices in the character's leg hole by two every time a new row of polygons is created.

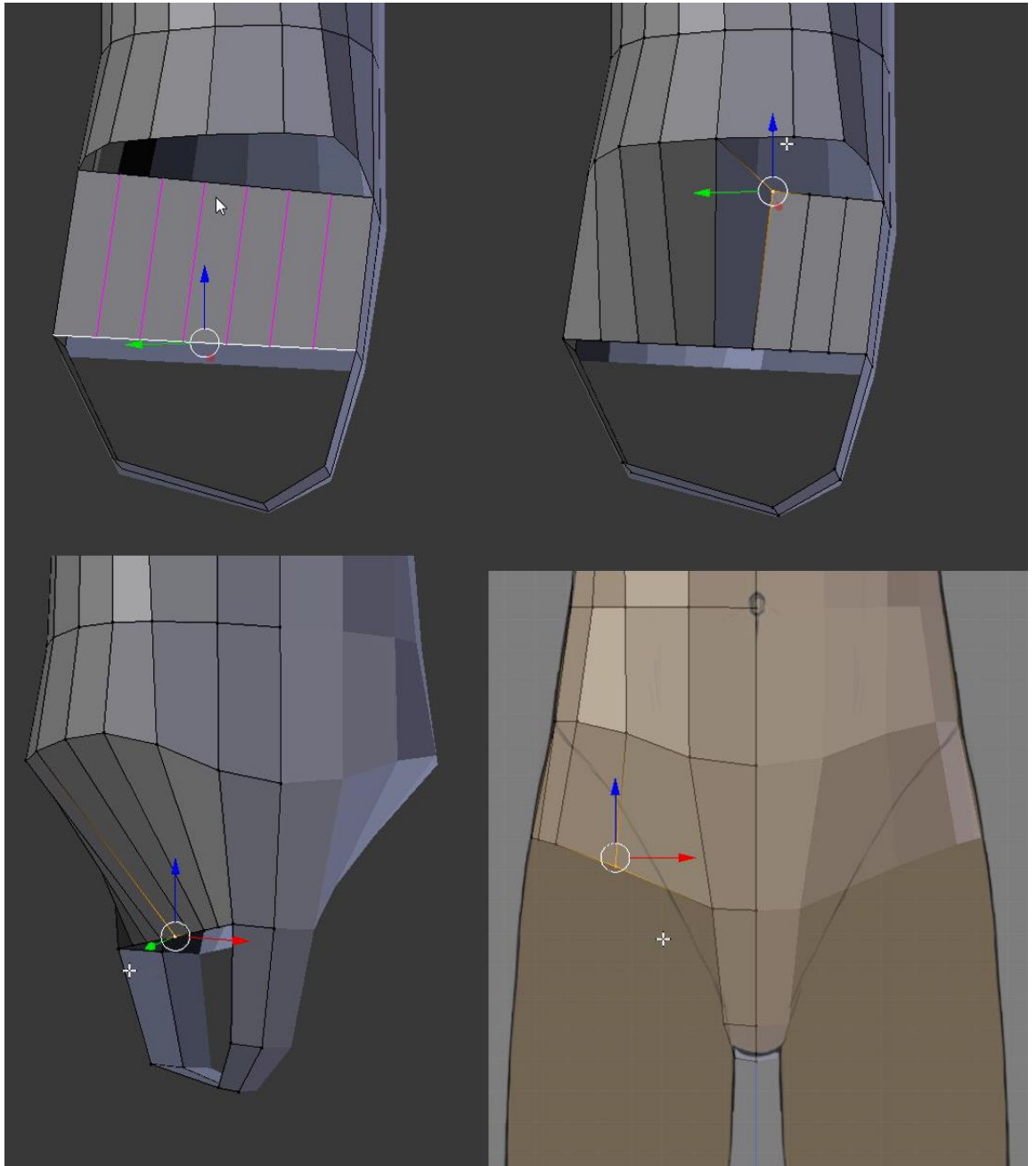


Image 33: Lowering the number of vertices in the leg holes.

In Aroleir's case, it was enough to do this process only once. If the number of vertices still feels too large but the modeler doesn't want to bring the character's hip any lower, the modeler can add these extra faces to the crotch area. This though means the modeler will need to re-adjust the silhouette to match the references. Once the number of vertices feels manageable and the leg holes have been rounded to feel natural, it is time to begin creating limbs for the character. Image 34 shows how Aroleir's finished torso looked like at this point.

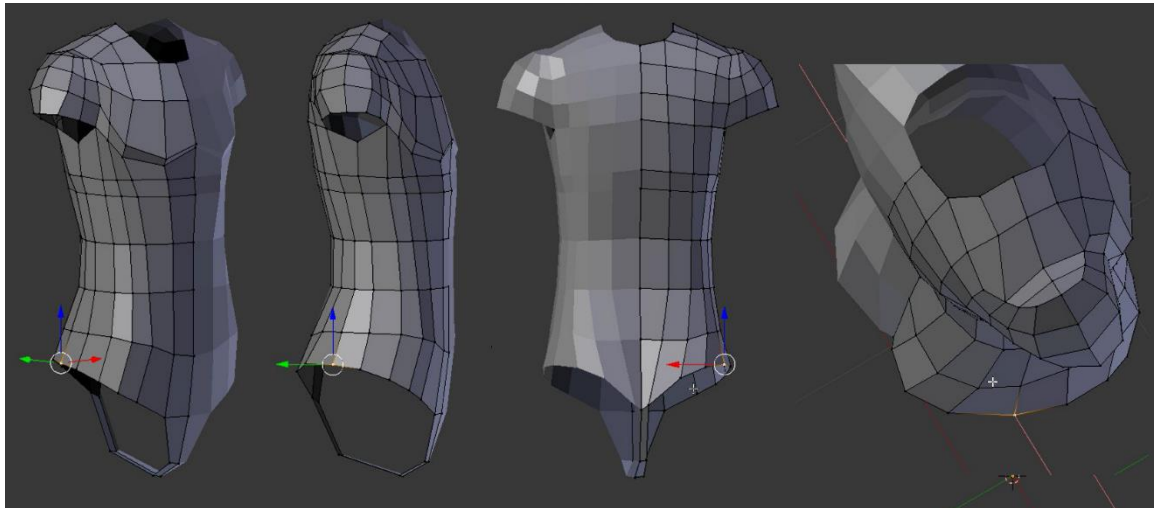


Image 34: The finished torso.

5.2.2 The limbs and joints

A character's arms and legs are created separately from the torso. They will be made as new mesh in the same object. While in Edit mode, a new mesh is created. Cylinder is a good starting shape for limbs as the amount of edges can be set from the start. As Aroleir's leg hole had 12 vertices at this point, a 12-sided cylinder was created. This cylinder needs to be re-sized so it is about the thickness and length of the character's leg and then moved to where the character's legs will be. The cylinder can also be rotated to match the reference images. (Image 35)

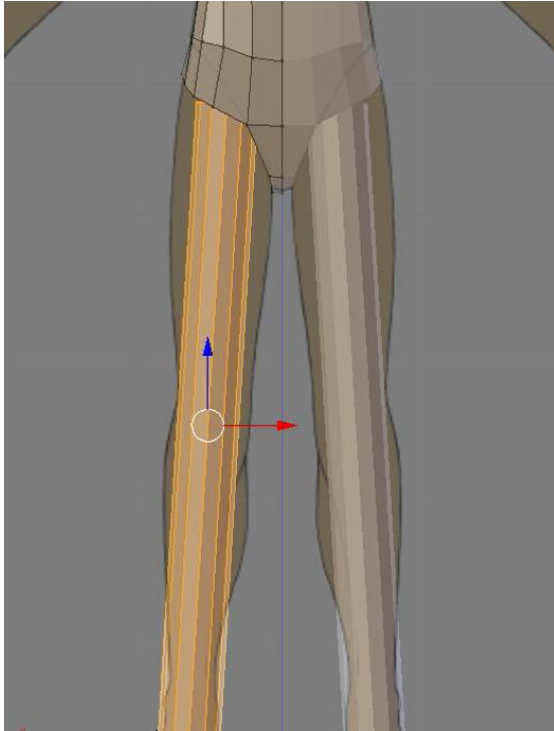


Image 35: The cylinder that is the base of the character's legs.

When created, the cylinder is placed wherever the 3D Cursor is. If the cursor is in the center of the scene, the cylinder will be placed so a part of it overlaps the middle line of the Mirror modifier. In this case, the modifier's Clipping option needs to be temporarily disabled so the cylinder can be moved to its intended spot. Otherwise the Mirror modifier will prevent the cylinder's vertices from going past the middle line and stretches the shape sideways.

Once the cylinder is in its place, it is time to begin shaping it. First of all, the cap polygons on the top and bottom of the cylinder need to be removed. After this, the vertices on the edges of the hole are selected and scaled to match the references. The bottom of the cylinder should match the character's ankles and the top should be roughly the size of the leg hole on the torso. The edge will likely need to be rotated slightly to make connecting the cylinder to the torso easier. In side view, the cylinder should be placed so the ankles are in the right spot.

The modeler needs to decide which vertices on the cylinder and the torso will be merged together. When the ankle is in the right spot, the modeler will pick whichever two vertices are the closest to being aligned. These vertices are merged together with the "Alt + M" command. The vertex on the cylinder should be merged

into the one on the torso so that the vertex on the torso does not move. This can be done by first selecting the vertex on the cylinder, selecting the one on the torso second and then picking the “At Last” command in the “Alt + M” menu. All the vertices on the cylinder should be merged to their corresponding vertex on the torso. (Image 36)

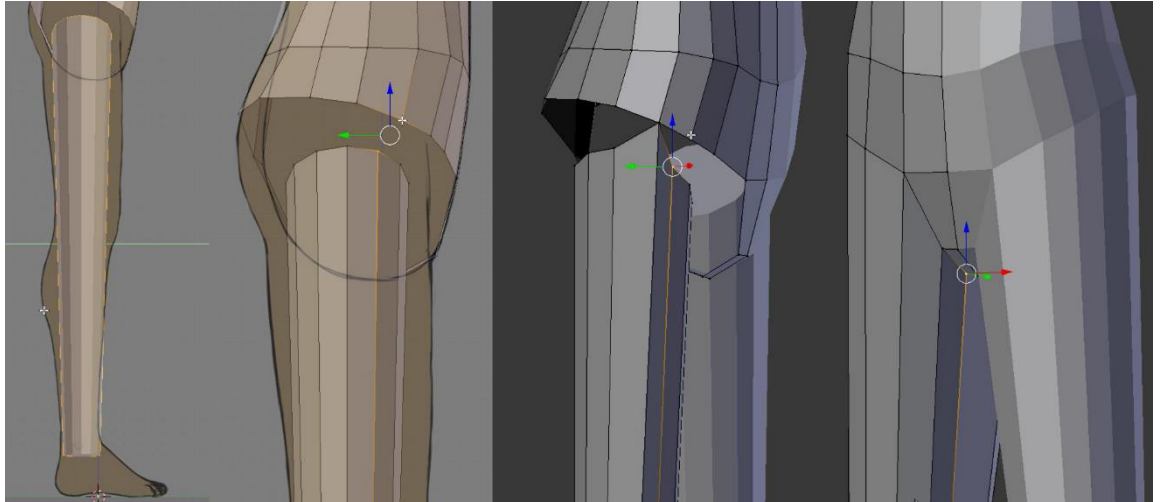


Image 36: Merging the leg cylinder to the torso.

Shaping the character’s buttocks can be quite tricky as the polygons easily distort when creating such rounded shapes. In this project, the challenge was approached by creating edge loops to the top of the leg cylinder and shaping them according to the reference images one at a time. It may be helpful to enable smooth shading in this phase as it makes any sharp edges and distorted polygons more obvious. Smooth shading can be enabled by selecting all polygons of the model (A), opening the “Shading / UVs” tab from the left side of the 3D Viewport and clicking on the “Smooth” button under the “Faces:” option. Image 37 shows different phases of shaping Aroleir’s buttocks.

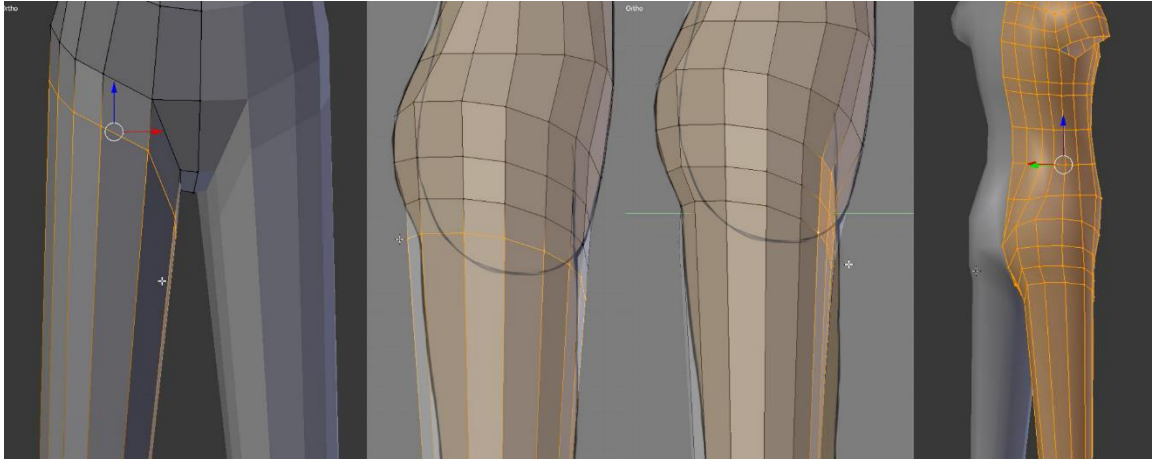


Image 37: Matching the upper leg to the references. The last picture shows how the model looks with smooth shading enabled.

It is up to the modeler to decide what kind of joints the character will have. Aroleir's knees and elbows got joints based on one of the examples on Polycount's Limb Topology page (Polycount, 2015). The first step was to make a horizontal edge loops where the knee needed to be. Because the edge loops from creating the character's buttocks are at an angle, the loops created under them will also be at an angle. They can quickly be rotated to be horizontal by selecting the vertices of the created loop and scaling them to 0 by the Z-axis (S + Z + 0). This needs to be done both to the knee loop and the ankle loop at the bottom. (Image 38)

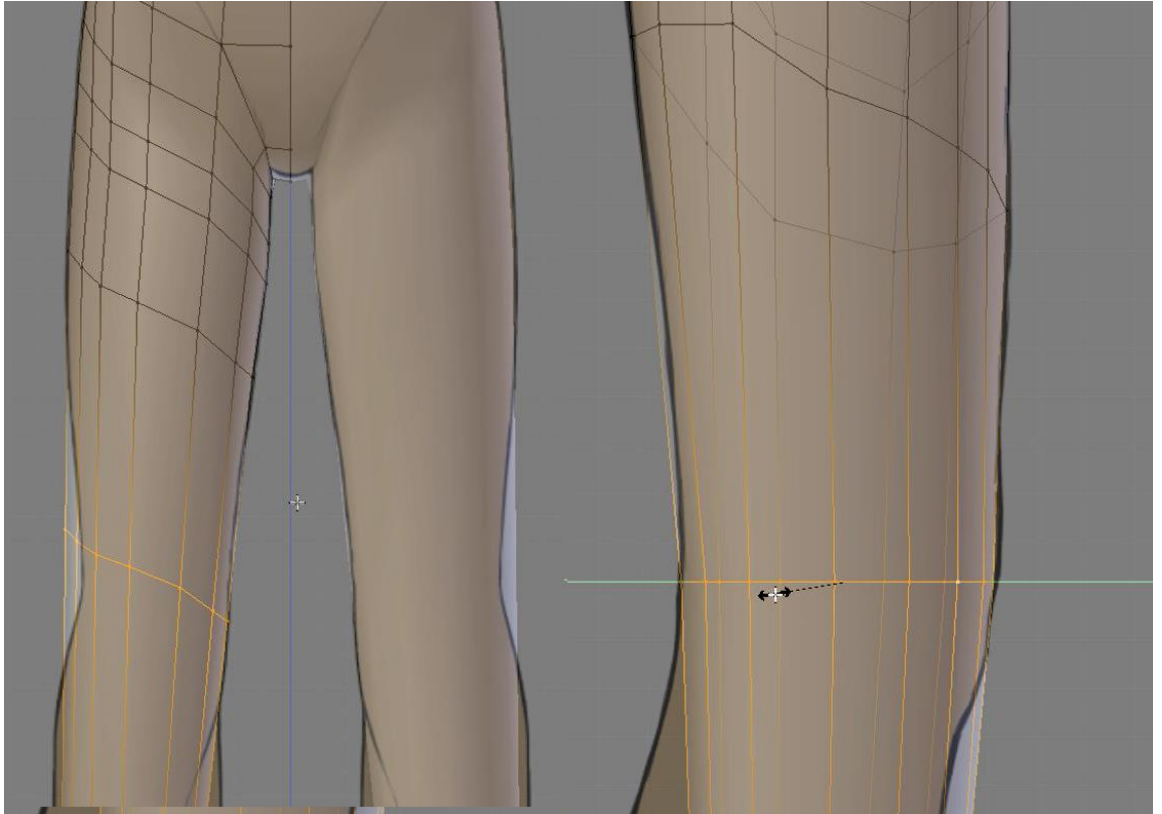


Image 38: Scaling an edge loop to 0 by the Z-axis creates a horizontal edge loop. This image also shows an extra edge loop added to the thigh. This edge loop was used to make the character's things look more rounded.

To make the joint itself, a second edge loop is created under the one made before. About half of the vertices of these edge loops will be merged together one by one behind the knee, leaving two triangles on the sides of the knee. To make the joint so it will distort less when animated, another edge loop is added below the knee. The polygons on the front will limit the area that will stretch during animation while the two loops behind the knee will prevent the joint from collapsing. The steps described can be seen in image 39.

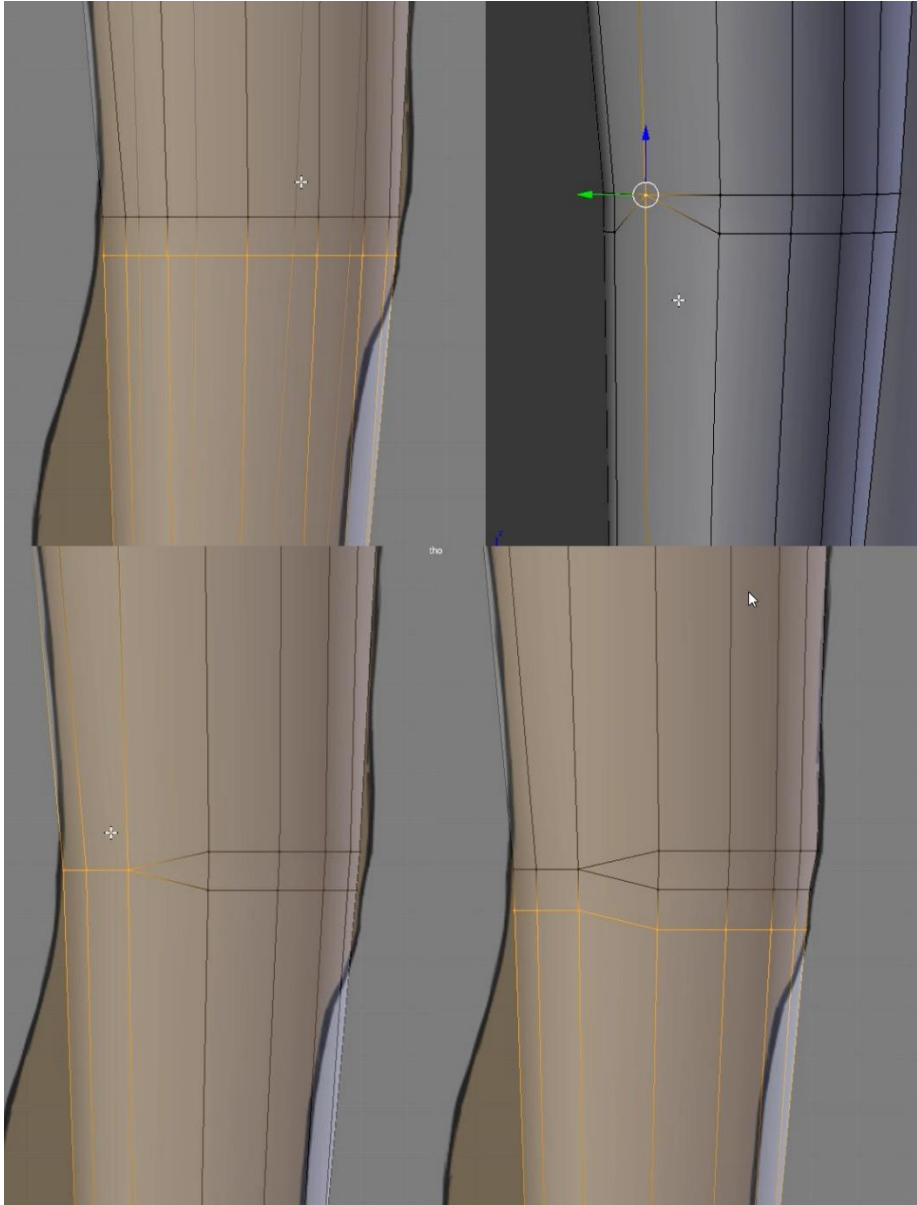


Image 39: Creating a knee joint.

After the joint is done, the rest is merely shaping the leg to match the references. Since Aroleir is quite realistic, his legs were given four extra edge loops to reach a natural shape. The loops were though added one by one, starting from near the knee so all changes done to the shape would also work as a point of reference to the future loops. The vertices on the inner side of his leg were also moved a bit inwards to create a shape that mimics the shape of a muscular leg. Image 40 shows how the edge loops were added and edited.

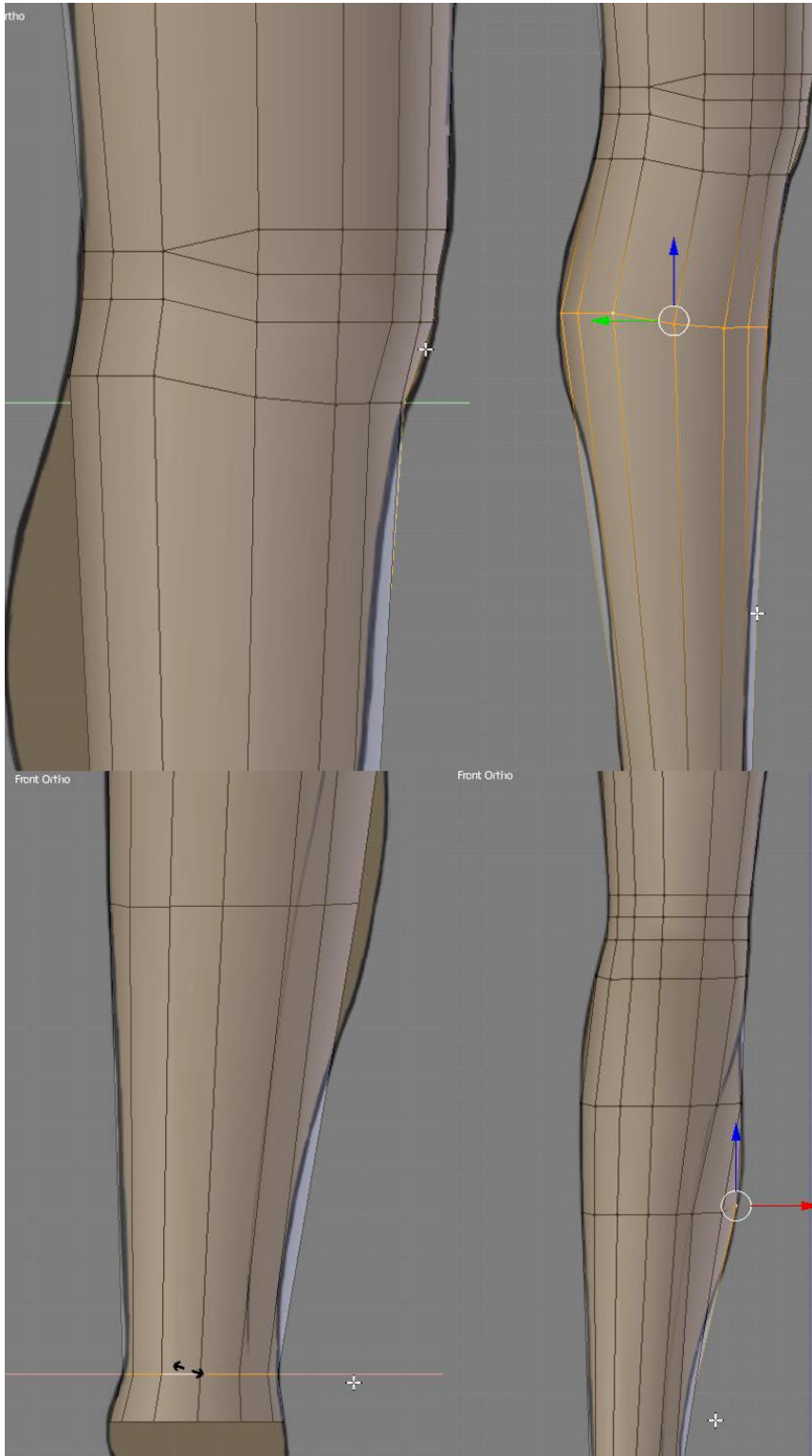


Image 40: Shaping the leg. Note how in the last shot the vertices have moved to make the leg less tube shaped and more like the references.

Creating the arms is mostly the exact same process as creating the legs; create a cylinder with as many sides as there are vertices on the arm hole of the torso,

resize and rotate it to match the references, delete the caps on both ends of the cylinder, attach the top vertices to the torso's vertices, shape the arm to match the reference. One major difference is that the new edge loops cannot be simply scaled to 0 on Z-axis to create a straight horizontal loop. This means the joint should be made before connecting the arm to the torso. The joint will also be facing the opposite way to the knee since human arms bend to the opposite direction compared to the knees. Image 41 shows some shots from the creation of Aroleir's arm.

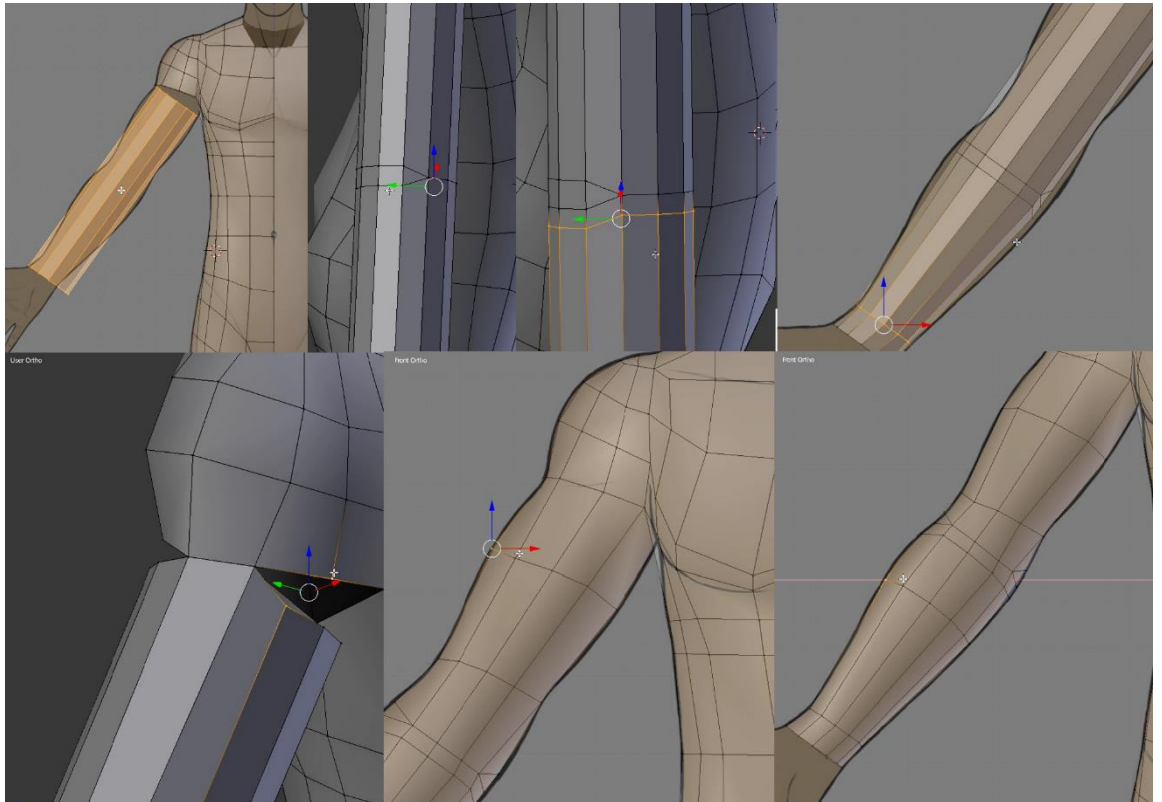


Image 41: The stages of creating an arm.

5.2.3 The feet and hands

Creating the feet begins by creating a cube in Edit mode and moving it to where the foot should be. The cube is then adjusted to be roughly the shape of the foot from both the side and the front. Once that is done, an extra edge loop is added about where the ankle would begin. The top vertices of this new edge loop are

selected and moved to be about the same height as the back vertices of the foot. Image 42 shows these steps.

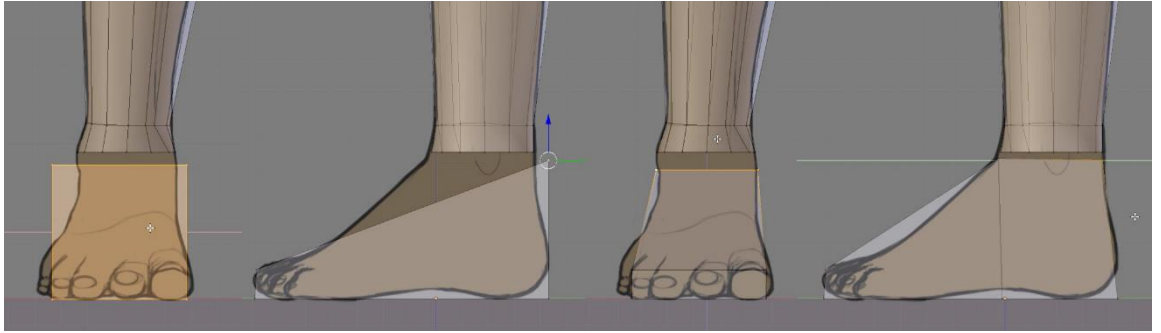


Image 42: Blocking out the shape of the foot.

Like with any other part that needs to be connected to the body, also the feet need to have a hole with the same number of vertices as the ankle holes on the legs. Aroleir's ankle holes had 12 vertices which is quite easily matched on the foot by adding four new edge loops that cross over the area that will be connected to the body. The polygons on top should be removed to create the hole for the ankle but it is better to shape the foot before connecting the parts. The edge loops added on Aroleir's foot can be seen in Image 43.

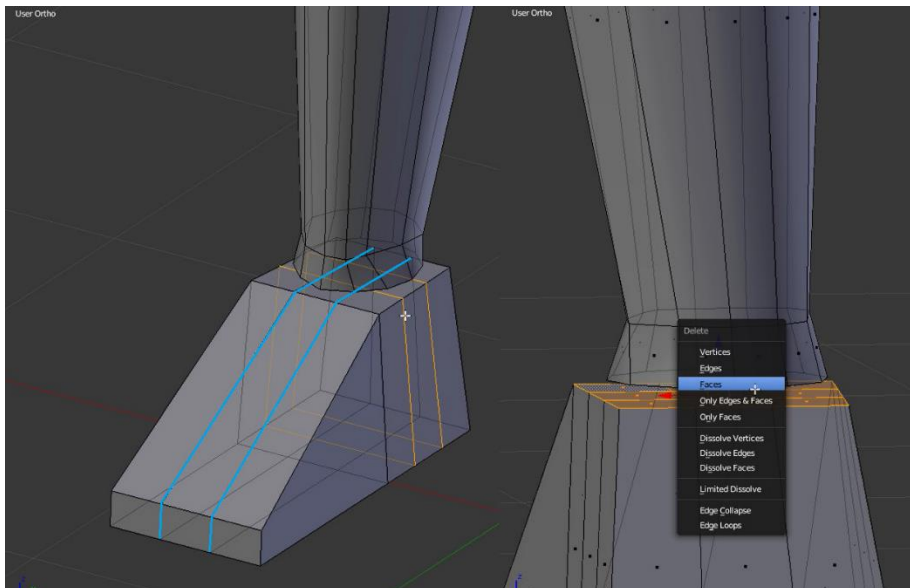


Image 43: The edge loops are added to create an even grid. The orange and blue lines highlight the new edge loops. The right picture shows which polygons are deleted to make the ankle hole.

The feet would be covered by shoes most of the time so most details would come from textures. This means the feet don't need refined details like separate toes. To make the foot into a shape that would look natural when textured, more edge loops need to be added to the sides that do not yet have many loops. The vertices of these edge loops are moved so they form more rounded shapes for the foot. Some points that are easy to miss are rounding the heel by pulling the bottom vertices inwards, raising the bottom of the foot on the inner side, and shaping the front of the foot to be the right shape when looked at from above. Image 44 shows what edge loops were added and the shape they were moved into. Note how the top of the foot's shape tilts to the side instead of being a perfect half-circle.

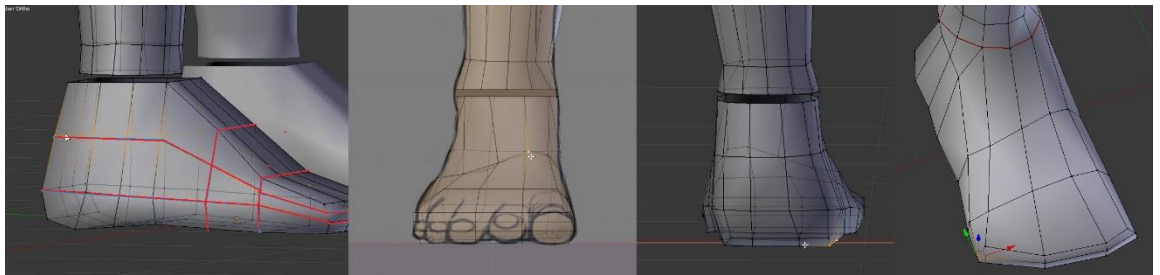


Image 44: Giving the foot its natural shape. The new edge loops are highlighted in red in the first picture.

Once the foot is ready, it is attached to the ankle the same way as the other parts before. Matching vertices are selected on the foot and the ankle and then they are merged together with the “Alt + M” command. Some vertices may need to be adjusted on the foot after being connected if the edges appear to tilt to one direction. After all, the foot was modeled from a cube and the leg from a cylinder, meaning the positions of the vertices may be different. Image 45 shows how the vertices of Aroleir's foot were twisted and how they looked after being straightened. The “G + G” command for Edge Slide is useful when straightening the edge loops.

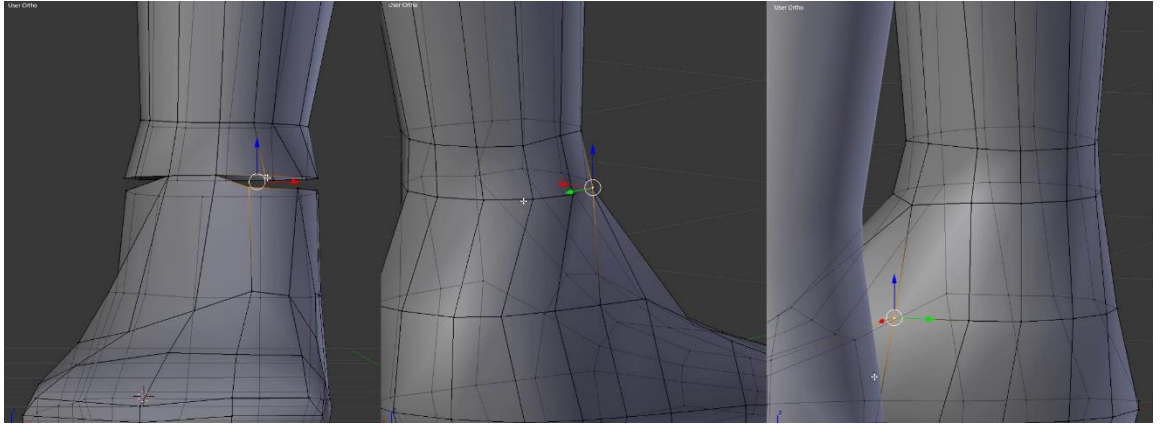


Image 45: Attaching the foot and straightening the twisted edge loops.

5.2.4 The hands

The level of detail needed for hands depends on the game. Since Aroleir is meant to be like the characters in Skyrim, he should have detailed hands as his hands would be seen up close when in first-person view. All fingers are separate and modeled in such a way that they can be fully animated. The hand models are even detailed in third-person view as the characters move their fingers when casting spells. Image 46 shows an example of a player character's hand in Skyrim.



Image 46: A character's hand in Skyrim when the character is sneaking without having a weapon drawn. Image rotated for a more natural view.

To create a hand, a cube is first created, resized and moved to the hand's place. The corners of the cube are moved to line up with the palm as seen on the reference images. The cube also needs to be scaled along the Y-axis to make the hand flat. Some edge loops need to be added to the shape and the modeler needs to keep in mind the number of vertices that the wrist hole has. Like before, these numbers need to match so the hand can be attached to the wrist later. Aroleir's wrist has 8 vertices, meaning the hand got two extra edge loops on both sides of the future wrist hole. The vertices of the hand are moved to align with the vertices of the wrist hole so the hand will be easy to attach once it is finished. The polygons at the end of the hand piece are also removed at this point to create a hole for the wrist. Image 47 shows how the vertices that would be connected to the wrist were before and after adjusting to the wrist hole's vertices.

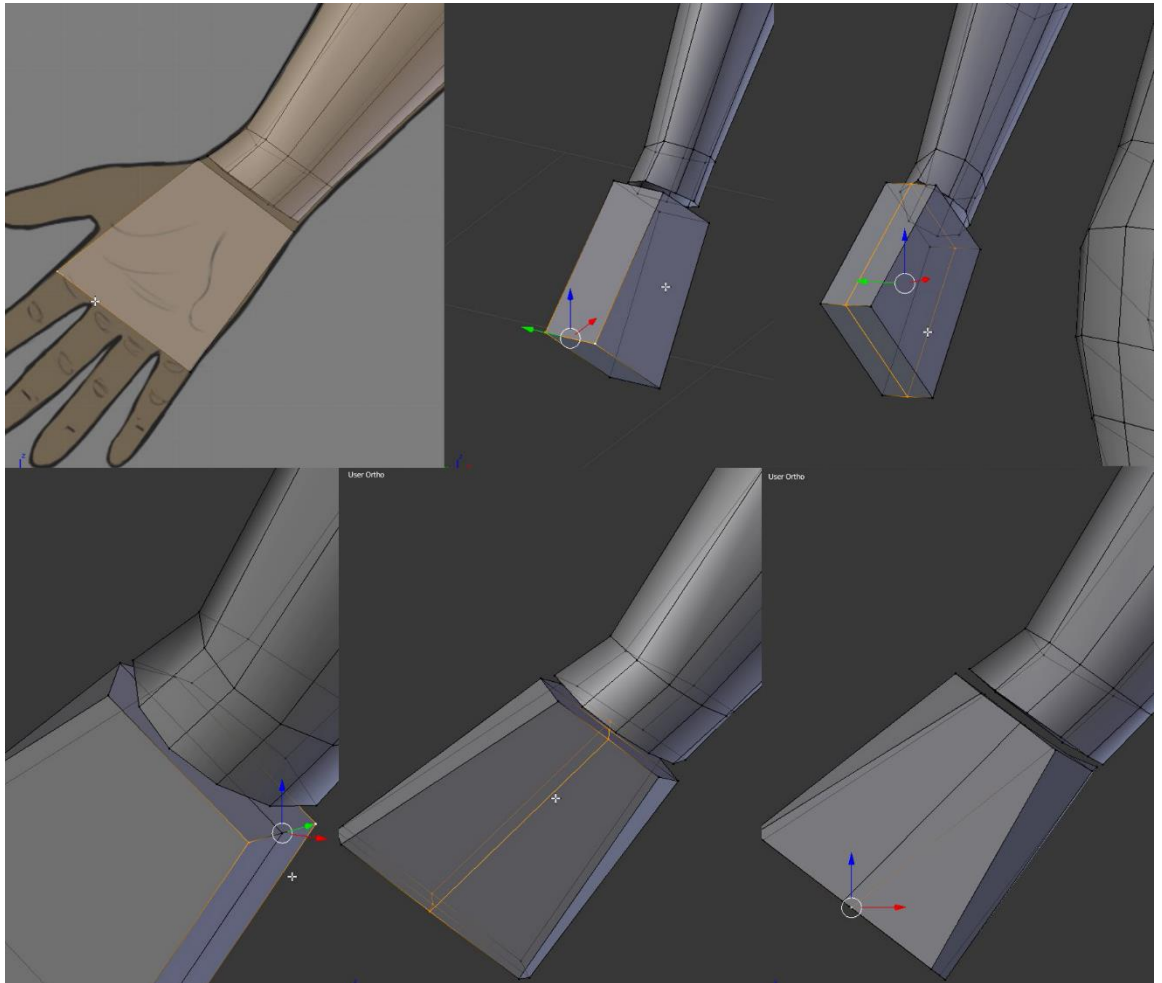


Image 47: The beginnings of creating a hand. In the last picture, the vertices of the wrist on both the arm and the hand have been moved to align for easy merging later.

The thumb is created next. The process begins by adding an edge loop to mark the part where the thumb will come. Some polygons on the side of the hand are selected and extruded (E) to create the root of the thumb. It is recommended to add an edge loop to run across the thumb sideways as it makes it easier to create the part of the thumb that is on the palm. The vertices of the extruded polygons are moved to match the reference pictures, only to have the polygons on top be extruded again. The finger will be extruded bit by bit until the desired shape is achieved both from the front and the side. The extrusions will give the finger a flat and angular form which needs to be rounded as the finger is being shaped. (Image 48)

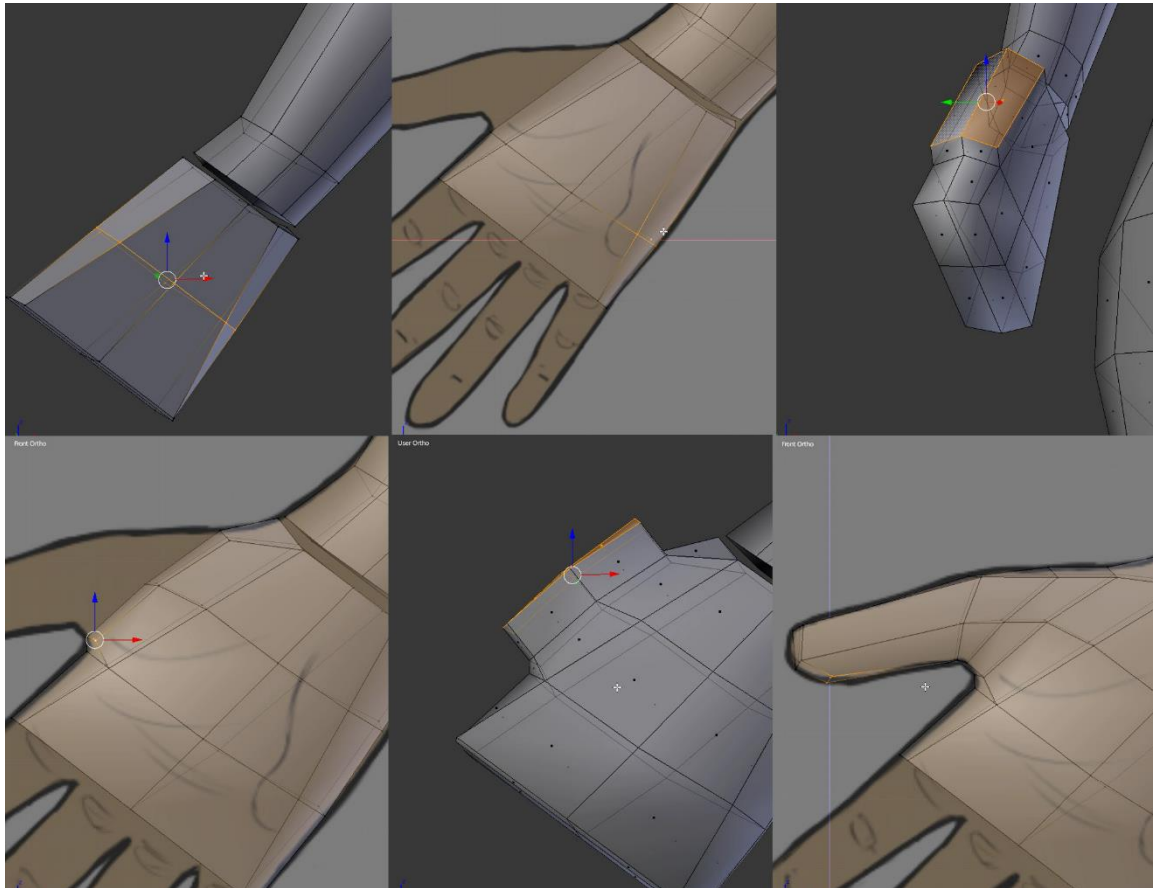


Image 48: The stages of creating the base shape of a thumb.

The thumb is not quite finished yet, though. The tip of the finger still needs to be rounded and a joint needs to be added. The shape of the fingertip depends on the level of detail and number of vertices at the modeler's disposal. Aroleir's thumb has eight vertices going around the finger which made it quite straight-forward to round up the fingertip. First, the finger's overall shape needs to be adjusted. A thumb is not perfectly round but rather of a more oval shape. It is important to note that on a real human hand the thumbnail faces towards the back of the hand when all fingers are straight. While it may be tempting to make the thumbnail face upwards along the Z-axis, having the thumb in such a position would make animating the hand more difficult later as the thumb would always rotate unnaturally.

Once the shape of the finger has been adjusted to be more oval, the faces that cap the fingertip are deleted. To remake the fingertip, the long sides of the oval are connected with faces by selecting two opposing edges and using the Face tool (F). This leaves two triangle shaped holes on the sides of the new faces. The

holes are filled by selecting the edges and again using the Face tool. All the created faces are then split with an edge. This will leave behind two triangles on both sides of the thumb but the extra edges are needed to make the thumb tip more rounded. The vertices created on this new edge are moved to shape the thumb tip. Image 49 illustrates these steps.

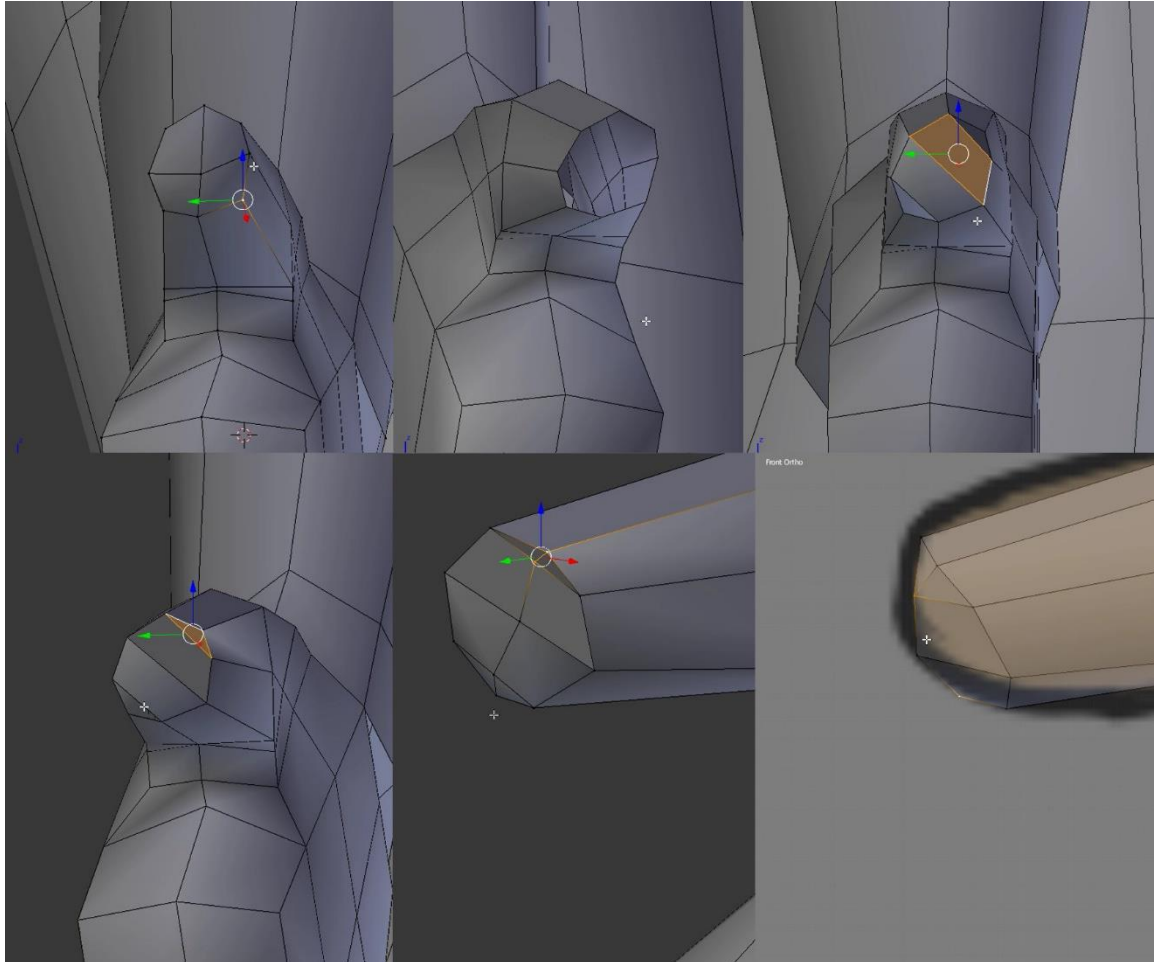


Image 49: Creating the tip of a thumb. Note the thumb's oval shape at the tip.

The joints of the fingers will be quite different from the joints of the knees and elbows. On the fingers the joints will only have two extra edge loops running around the finger. On an eight-edged shape like Aroleir's fingers, the joint will be symmetrical. The inner edge of the finger, or the edge towards which the finger will bend, will have its edges merged together like was done with knees and elbows. This means the two parallel edge loops will be merged together by selecting corresponding vertices on both edge loops and merging them with "Alt + M" and

selecting “At Center”. This will create two triangles on both sides of the merged edges.

The next step is to select three edges on the side opposite to the newly-merged edges. These three edges are subdivided with the “Subdivide” button on the left toolbar of the 3D Viewport. This creates a pentagon shape on both sides of the new edges. The pentagon needs to be divided into three triangles by connecting the ends of the new edge to the two bottom corners of the pentagon. All these steps are shown in Image 50.

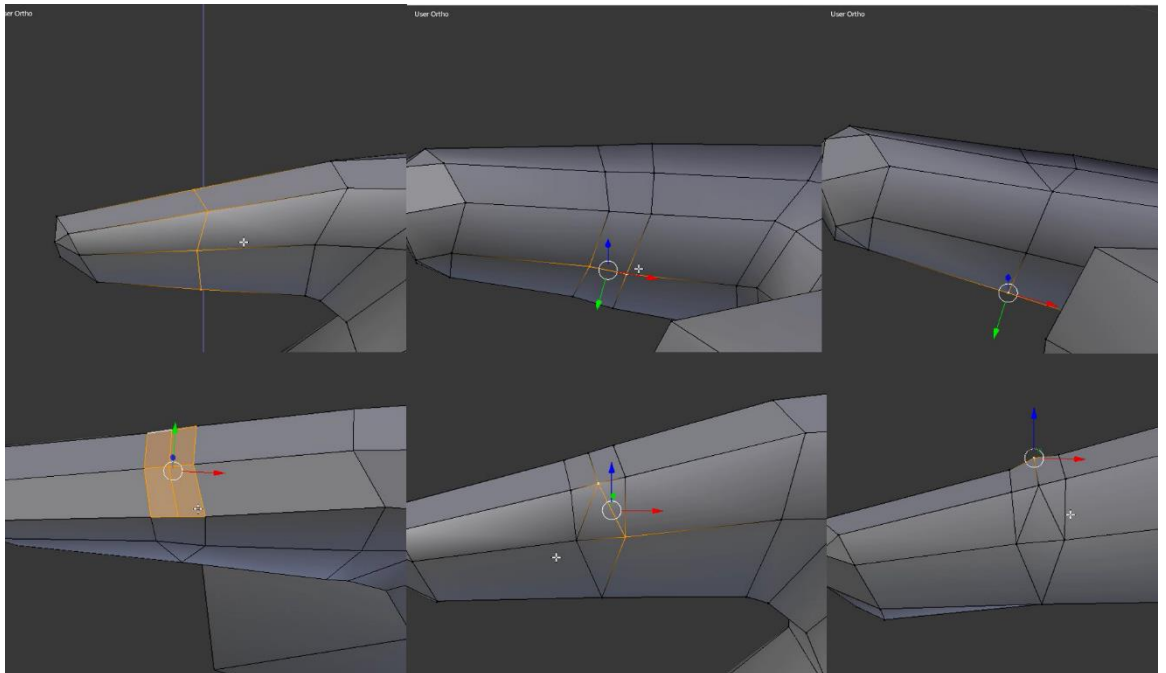


Image 50: The joints will be the same on all fingers.

For extruding the other fingers there are no proper work-in-progress shots as the recording was cut off, but the steps will still be illustrated as screenshot edits. To create the fingers, more edge loops will be needed. These loops will later have their ends merged together so they will not interfere with attaching the hand to the wrist. These edge loops will be added to where the fingers will separate. Once the edge loops are in place, the fingers are created by selecting the faces the fingers should begin from and extruding them (E) one finger at the time. To ensure all fingers will be straight, these extrusions should be made while viewing the character directly from the front or the back so the references are showing. Image 51

shows where the loops are placed and which polygons should be paired up for extrusion.

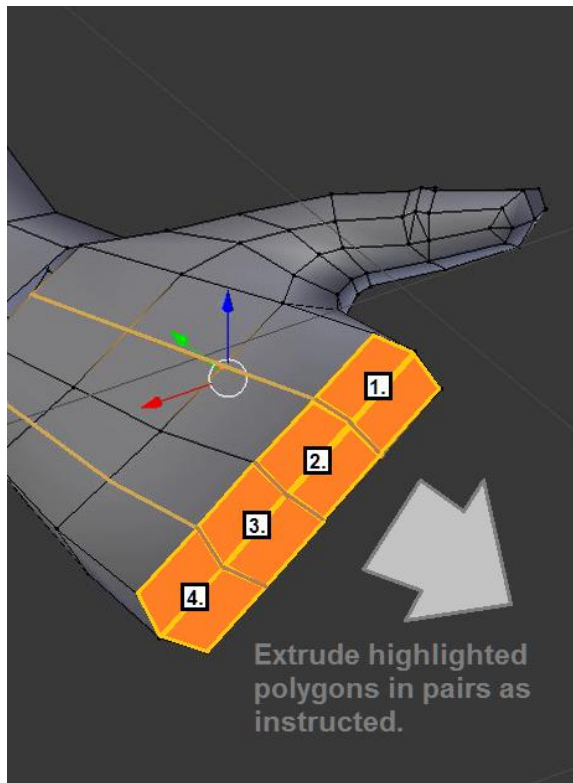


Image 51: Extruding the fingers. The fingers need to be extruded one by one so they will not be fused together.

After extruding the fingers, edge loops are added to each finger so they all have eight vertices around the fingertip. This way the fingers can be created like the thumb was. The only differences are that these fingers will have two joints each and these fingers will be round rather than oval shaped. Image 52 shows the stages of finishing up the fingers.

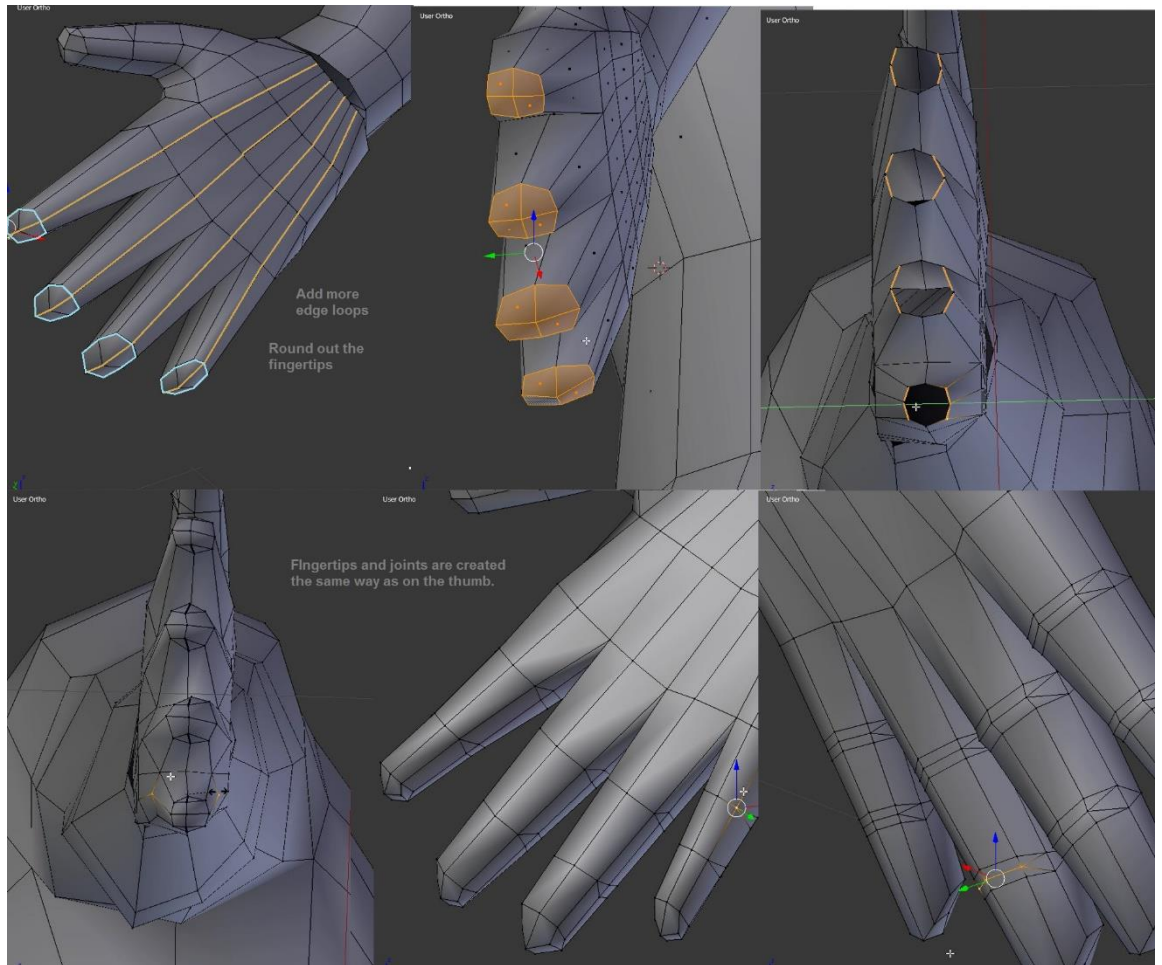


Image 52: Creating the fingers, fingertips and joints. The two last pictures show the finished finger joints from both sides.

Creating the palm and back of the hand for Aroleir was an experimental process. The hands needed to have certain minute details like knuckles and be manageable to animate while also not having too many polygons. An added challenge was finding a way to merge together all the extra edge loops from the fingers before the edge loops reached the wrist. This meant the character's hands would have triangles here and there and the true task was finding places where the triangles would not cause problems.

The back of a human hand does not deform much when the fingers move which makes it a good place to hide triangles. Meanwhile, the palm deforms heavily when the hand moves but it also is easy to hide with the fingers, a weapon or spell effects. With these points in mind, the focus with Aroleir's hand was on having the hand look good when in a neutral pose.

Both the back of the hand and the palm have small bumps at the root of the fingers. Triangles can be used to allow certain vertices to be raised higher than the ones next to them without the polygons bending in unfavourable ways. To create the knuckles, two edge loops were added to the hand. One was placed slightly under where the highest point of the knuckles would be and another was placed where the knuckles end. Image 53 shows the actions done after this as they are difficult to explain in words alone.

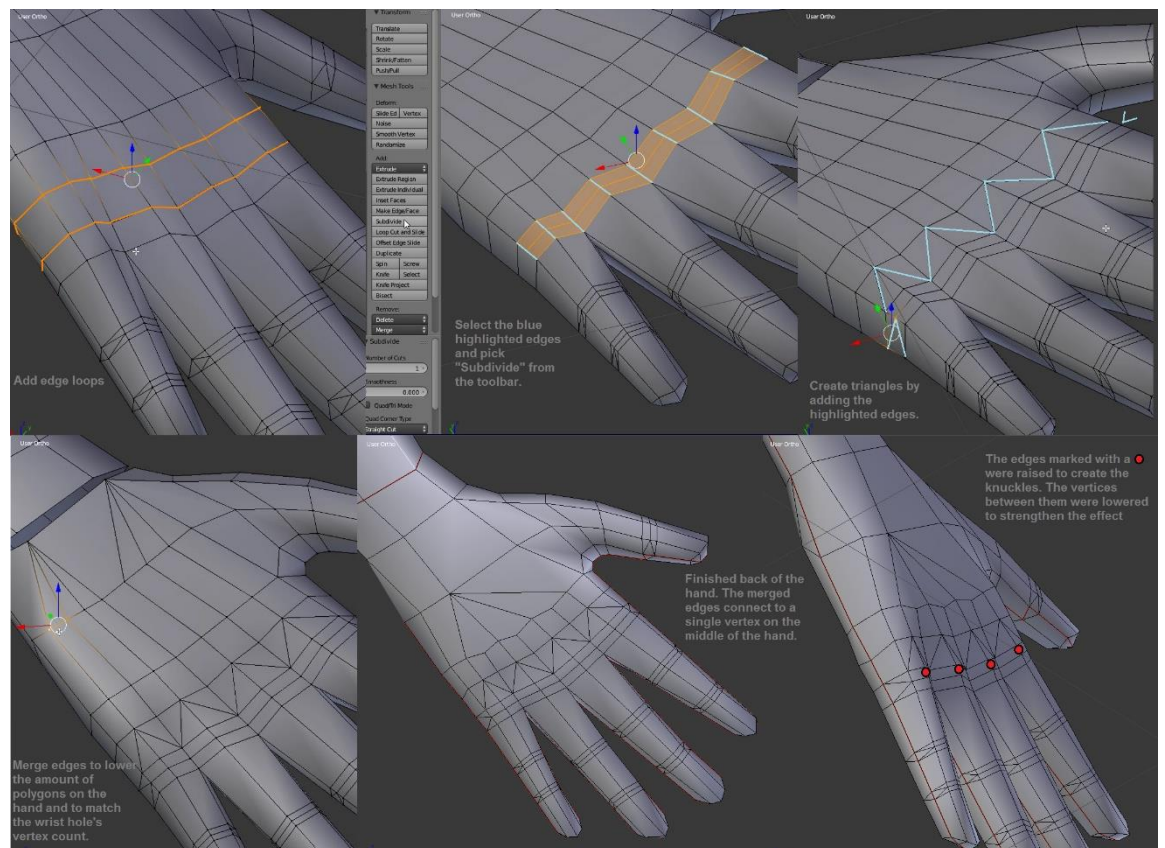


Image 53: The creation of knuckles and merging of the excess edge loops on the back of the hand. The triangles in the third picture were made by selecting two vertices and using the Join tool (J).

Since the edge loops made when creating the knuckles also reach to the palm, no new edge loops need to be added. It may not be too important to have all the small shapes of the palm down perfectly, but they were still modeled on Aroleir's hand as a personal challenge. Like with the knuckles, some triangles were created where the thicker areas at the root of the fingers were and the vertices were lifted and lowered to create the bumps. Certain vertices were fused together to rid of the

extra edge loops from the fingers. It is important to make sure the original edge loops that were aligned with the vertices of the wrist in the very beginning will remain in their place so the hand's polygons will not twist when the hand is attached to the wrist. Image 54 shows some work-in-progress shots from the creation of the palm along with the final product. Note how the edge loops added for the fingers were merged into the original edge loops of the hand.

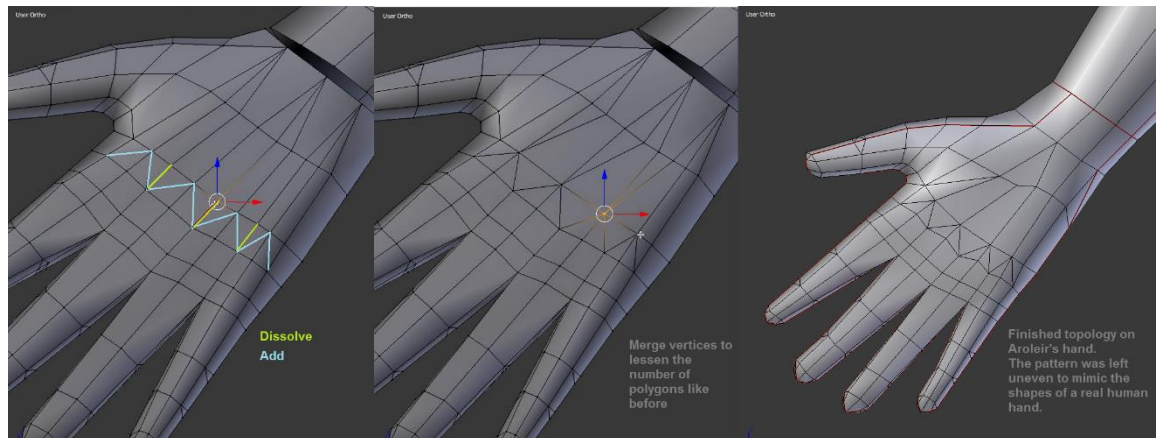


Image 54: Shaping the palm. One of the quads at the root of the thumb was split into two triangles so the shape would mimic the shape of a real thumb.

Attaching the finished hand is done the same way as the other parts before. The modeler should check that all the vertices of the hand are aligned with the vertices of the wrist before merging them together or the wrist might end up with twisted polygons.

5.2.5 The head and face

The way Aroleir's face was created was based on a tutorial on 3DTotal.com (Roger, 2009). Certain things were done differently but the methods and order of creating things were similar. In this phase, it also turned out that Aroleir's side and front references did not match up perfectly which made some areas harder to figure out. In such cases the front view took priority over the side view.

While the other body parts began with a simple shape like a cube or a cylinder, the face begins with a single plane. The plane is rotated 90 degrees around the X-

axis ($R + X + 90$) and scaled down so it is very small. While looking at the character from the front so the reference image is visible, the corners of the plane are moved so they form the first polygon of the corner of the character's eye. Once the polygon is in a fitting place, the view is rotated so the character is seen from the side. The single polygon is moved along the X-axis so it lines up with the edge of the eye. The vertices should not be moved on any other axis as that would alter how the polygon looks from the front. Image 55 shows how to position the first polygon and showcases how the eyes are not perfectly aligned in the side and front references.

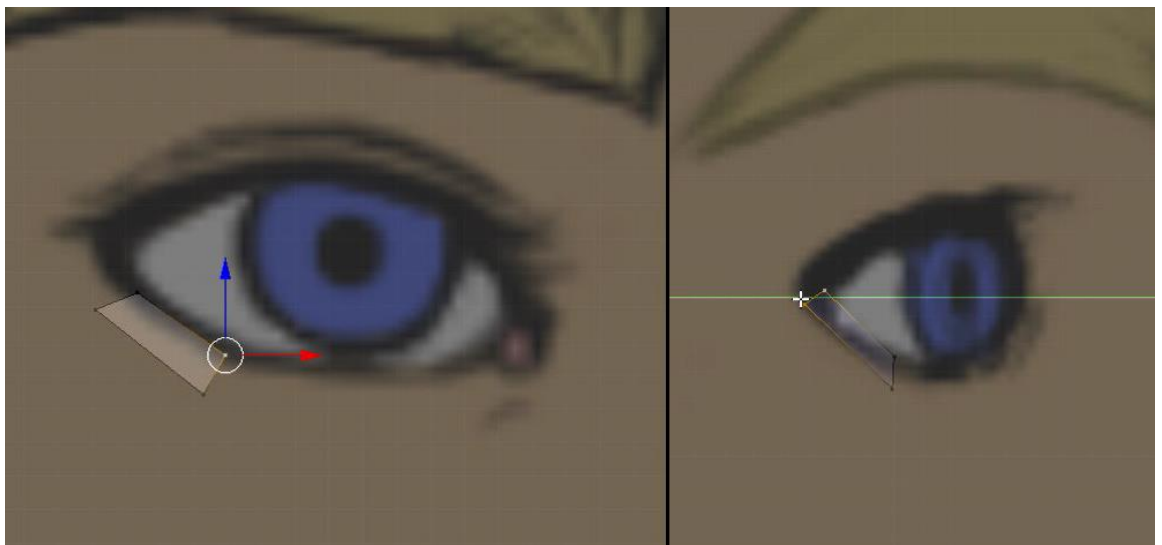


Image 55: Positioning the first polygon of the eye.

Once the first polygon is in place, the rest of the eye hole are created by selecting an edge and extruding it along the outline of the eye. This can either be done by one-by-one extruding and adjusting the polygons, or by first shaping the eye hole from the front, and then adjusting the shape from the side. In Aroleir's case the latter style was used as it felt easier with the fact the eye was not perfectly aligned in the side reference. It can be helpful to make the eye hole so that it has an even number of polygons. Aroleir's eyes ended up having 12 polygons forming the shape of the hole. The modeler need not worry too much about the shape of the eye hole in this phase as it can be, and probably needs to be, adjusted when the eyeballs are made. Image 56 shows how the eye holes were shaped for Aroleir.

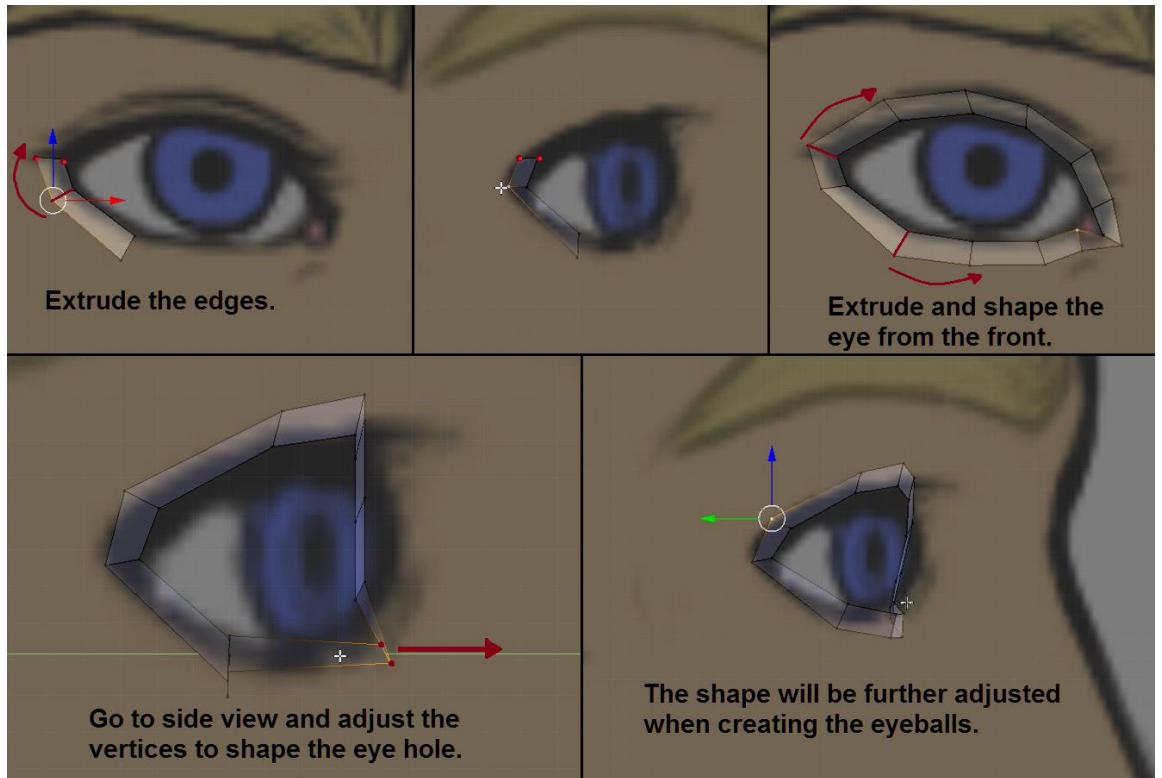


Image 56: Shaping the eye holes.

The mouth needs to be shaped next. Like with the eye, also mouth begins by creating a single plane and adjusting its shape to be the first polygon of the mouth hole. The shape should frame the outer edge of the lips. The mouth is first shaped from the front and next from the side. In this phase, the modeler need not worry about having an even number of Polygons as the Mirror modifier ensures the total amount is always even. See Image 57 for the way how the polygons were placed on Aroleir's face.

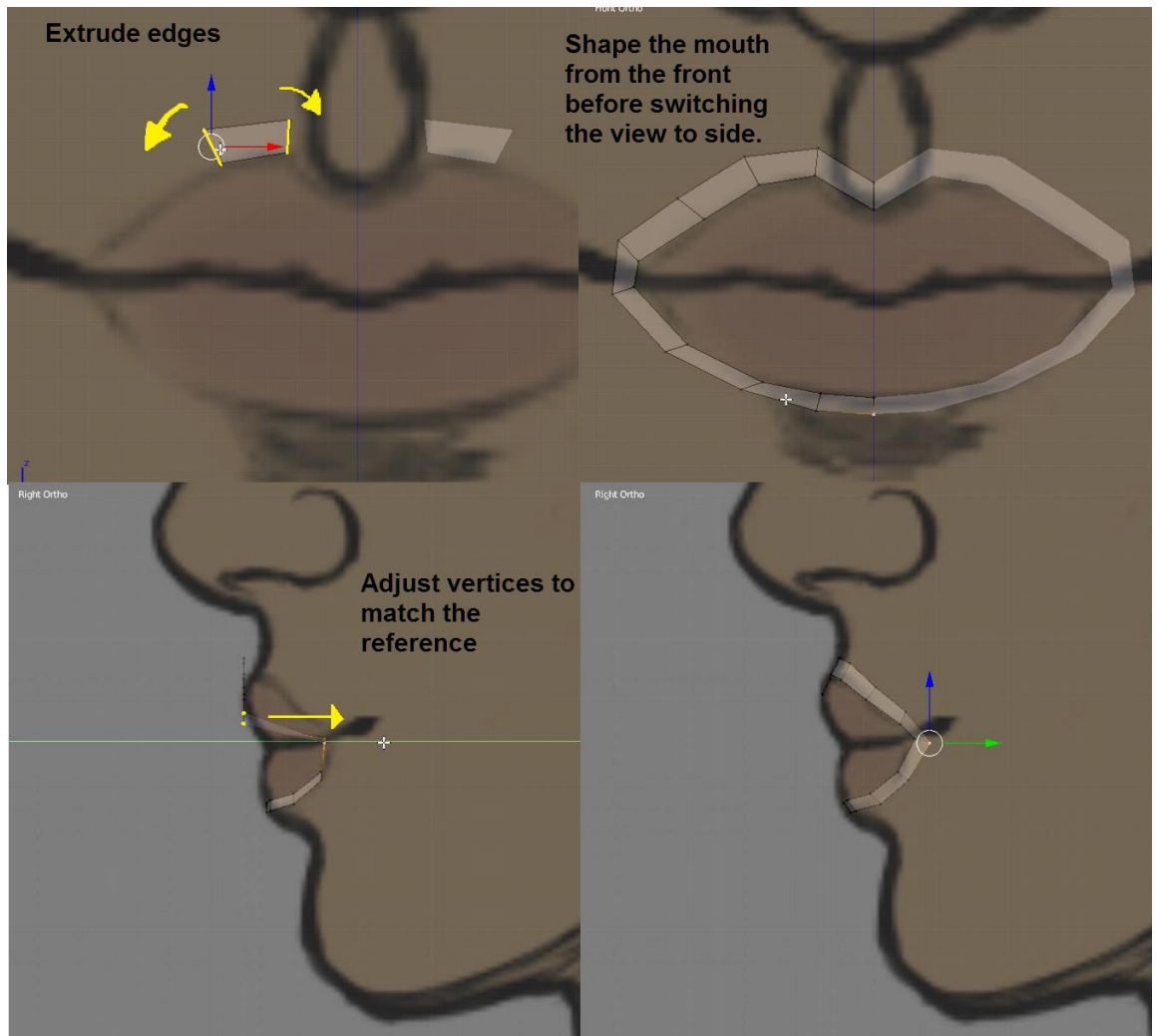


Image 57: Shaping the mouth is a similar process like with the eyes except the Mirror modifier does half of the work for the modeler.

Once the eye and mouth holes are done, they need to be joined together. The modeler needs to choose edges to connect on the eyes and the mouth. An edge is selected on both the eye hole and the mouth and a face is created between with the Face tool (F). On the mouth, the first edge to be connected should be the outer edge of the top lip. The side of the mouth should not be connected to the mouth this way. A rule of thumb is that the connected edges should not overlap with the sides of the nose when looking at the character from the front. Other than that, it is mostly up to the modeler to decide how many edges they will connect between the mouth and the eye.

The second step is to extrude (E) the remaining edges of the top lip to the bottom of the nose and to extrude the polygons that would form the area between the

eyes. The modeler should pull the polygons between the eyes all the way to the middle line so the Mirror modifier will weld them together. After this, some edge loops are added to the newly-created faces. It is up to the artist to decide how many polygons they wish the character's face to have but it is better to start simple. It is faster to shape the face when there are less vertices to move about. Regardless of how many edges are added, the modeler should strive to have the sizes of the polygons be relatively even on these extruded areas so the vertices are easy to merge together by selecting corresponding vertices and using the Merge tool (Alt + M).

Once the faces have been created, they need to be adjusted so they follow the intended shapes of the face. This means the edges should mimic the lines set for the nose in both the front and side references and the straight lines should be rounded for a more natural look. There is no need to worry about the shape of the cheeks from the side yet as they will be one of the last things to be refined on the face. See image 58 for how these steps were performed on Aroleir.

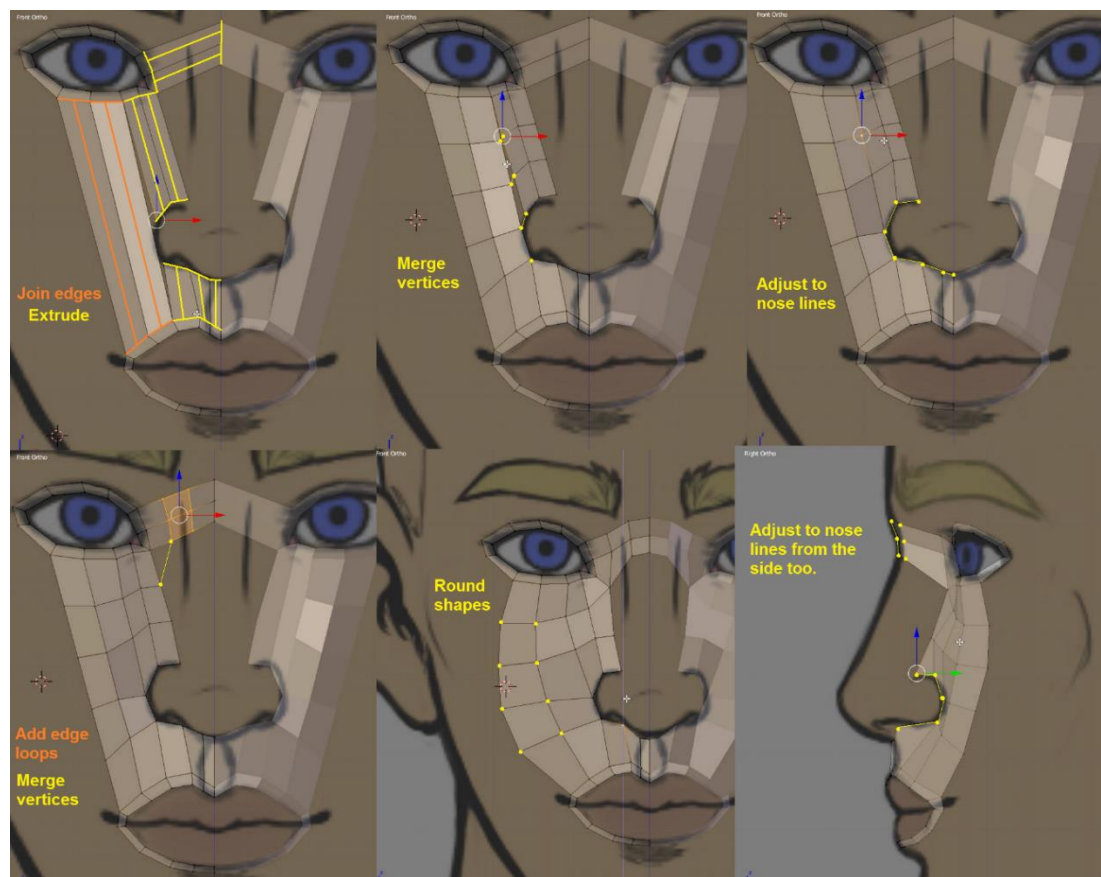


Image 58: Connecting the eyes to the mouth.

The areas around the eyes are done by extruding the edges around the eyes. Extruding a rounded edge is slight different from usual as the edges all move in the same direction regardless of orientation. The modeler needs to select the edges, extrude them without moving (E + Left Mouse click) and then scale the new edges outwards (S). Once the areas around the eyes have been extruded enough, the character's nose can be shaped from the side view. For Aroleir's face, the edges around his eyes were extruded three times now. Image 59 shows the extrusions and how the new vertices are moved to create the bridge of the nose.

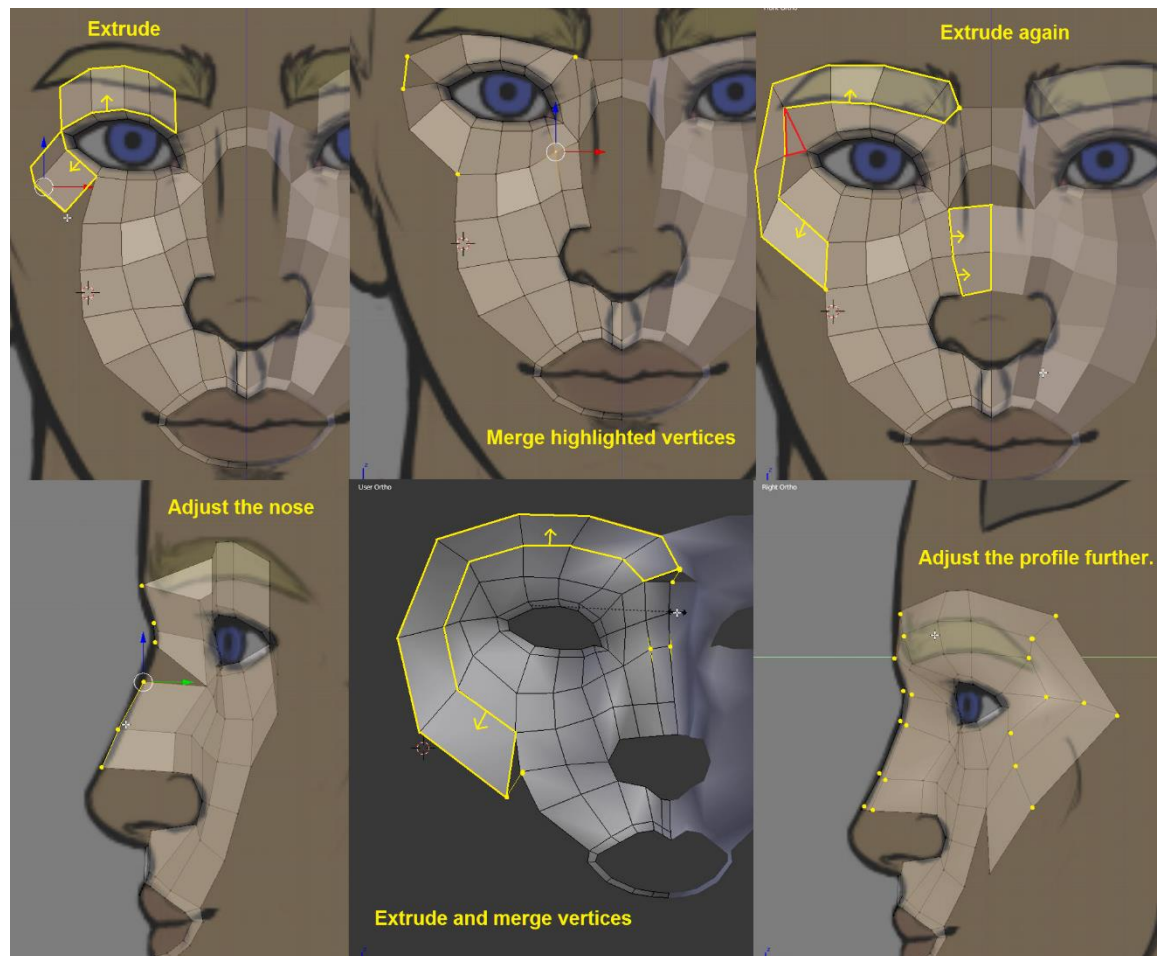


Image 59: The face is formed little by little. Note that the red triangle in the third picture will not be there if you merge the two vertices marked on the second picture. This triangle was removed from Aroleir's face later.

Since the polygons around the eyes have been created, this is an appropriate time to create the eyeballs and further refine the eye holes. The eyeballs should be created as a separate object for now as it is easier to adjust the eye holes when

the eyeball's vertices cannot be accidentally selected. This means the sphere needs to be created while in Object mode instead of Edit mode. Once the eyeball has been created, the Mirror modifier needs to be added to it separately. As before, for the Modifier to work, the object's center needs to be at the middle of the scene.

The size and complexity of the eyeball varies greatly depending on the character. Aroleir's eye will have transparent lenses over the irises and pupils. The lens will be created using the Glass material from Blender. The modeler should check if the game engine used supports the Glass material before committing time to creating eyes like these.

In eyes created this way, the number of polygons on the eyeballs should not be too low or the irises and pupils will appear to have obvious angles. For Aroleir's eyes, An UV Sphere with 16 segments and 16 rings was created. After creating the sphere, it is time to return to Edit mode. The sphere was then rotated 90 degrees along the X-axis (R + 90 + X) and moved so the middle point of the eye would be at the center of the pupil. When adjusting the Aroleir's eyeballs, the size was set so the second ring from the center of the pupil would outline the iris. If the iris is so large that the eyeball would be massive when scaled like this, the iris ring can be scaled independently without compromising the eyeball's round shape (G + G).

Once the eyeball is the right size, it is time to adjust the eye holes to match it. The vertices of the eye hole should be moved so they do not touch the eyeball but also do not leave large gaps anywhere. It is possible that the modeler must ignore the side reference of the eye in this phase and simply trust their own judgement to get the shape right. Aroleir's eyes ended up looking quite a bit smaller from the side than anticipated. Once the eye hole has been adjusted, its edges need to be selected and extruded inside the head. The extruded polygons should go through the eyeball so there are no holes between the eyeball and the eye holes. As a finishing touch, the edge of the eye hole is selected again and the Bevel tool (Ctrl + B) is used to soften the edge around the eyeball. Image 60 illustrates all these steps.

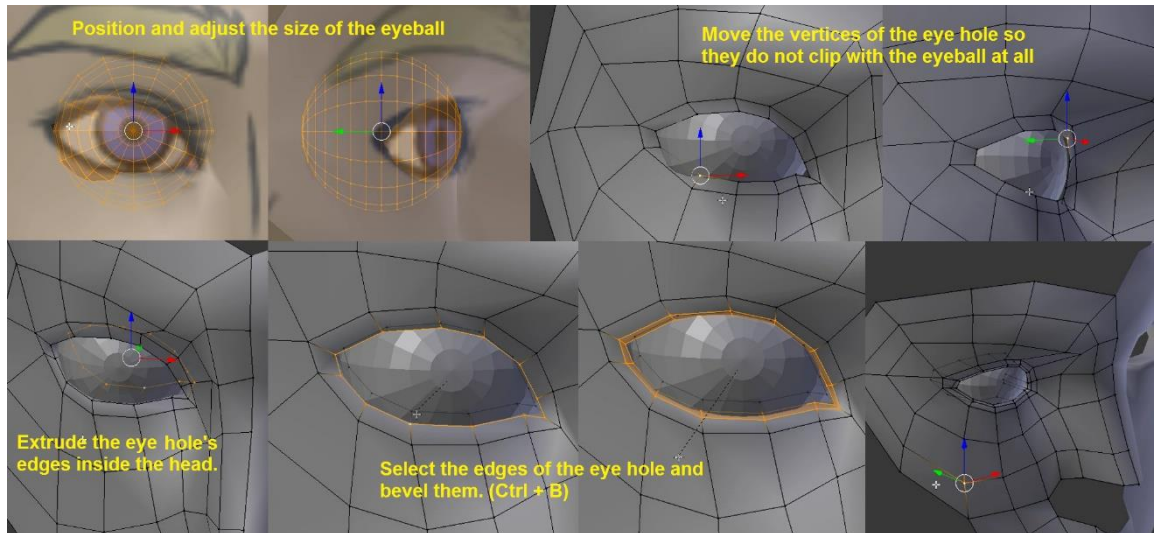


Image 60: Creation of the eyeballs and adjusting the eye holes to match.

Before creating the transparent lens of the eye, the area underneath needs to be prepared. The polygons that would form the iris and the pupil are selected and extruded just slightly inside the eyeball. Next, the pupil is selected and extruded much deeper inside the eye. The edge loop around the pupil hole can also be moved forward so it is further forward than the outer edges of the iris. This way the iris can be seen even from the side which adds to the realism.

As for the lens itself, its main part made by creating a duplicate of the polygons of the iris. This duplicate is then moved forward so it line up with the edges of the dip that was extruded from the iris. The duplicate is also scaled slightly larger so the edges between the eyeball and the iris do not peek through it. To make working on the eye easier, the lens can be separated to a different object (P + “Selection”). This way the lens can be edited without accidentally editing the eyeball underneath. The outer edges of the lens are selected and extruded back inside the eyeball so there will be no gaps between the eyeball and the lens. After this, the vertices around the hole on the lens are selected and merged together in the middle (Alt + M).

With Aroleir, every other edge on the middle of the lens were dissolved (X + “Dissolve Edges”) as test to see how having them as quads would work as opposed to leaving them as triangles. Doing so is completely optional. To create the rounded shape of the lens, it needs an extra edge loop around the middle. Though

because of the pole in the middle, the “Loop Cut and Slide” tool cannot create an edge loop there. All the edges starting from the middle vertex need to be selected and split with the “Subdivide” tool instead. The vertices of the new loop along with the middle vertex are moved forward to create a smooth curving lens above the iris and the pupil. These steps can be seen in image 61.

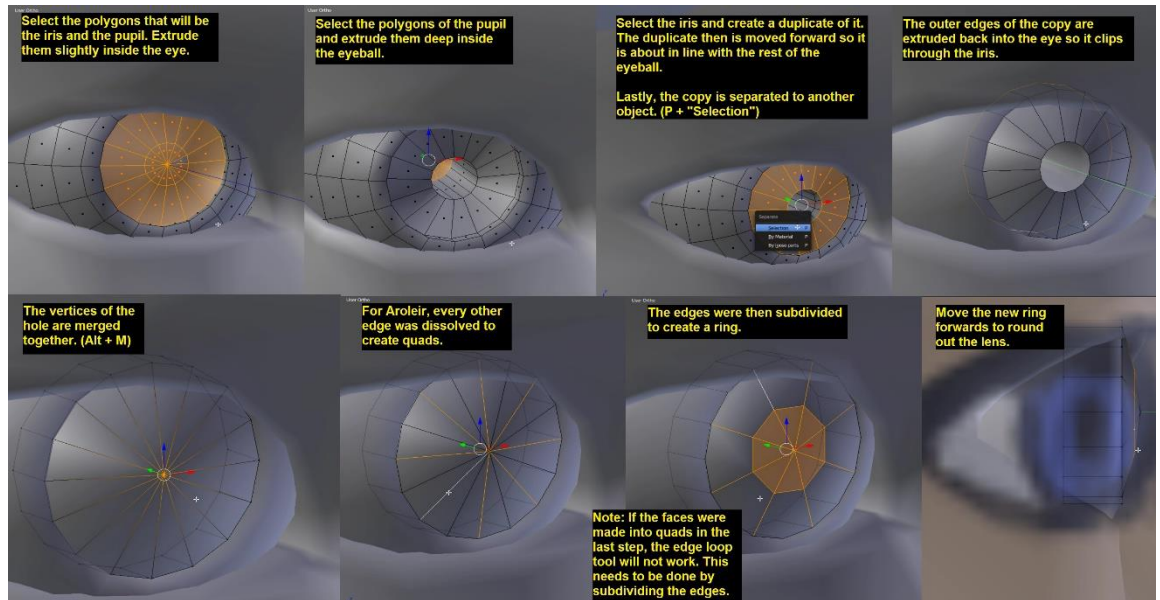


Image 61: The steps of creating a lens that only covers the iris.

To make the lens appear transparent in the render, a material needs to be added to it. Blender has a pre-made material called “Glass BSDF” which is suitable for the lens. The material is first created in the material tab with the “+ New” button. The “Glass BSDF” material can be selected in the “Surface” drop-down menu. The colour of the lens can be changed if needed but the default white works for human eyes. To apply the material to the lens, all the faces of the lens need to be selected (A). Next, the “Assign” button under the material list is clicked while the glass material is selected. Image 62 shows the finished material settings.

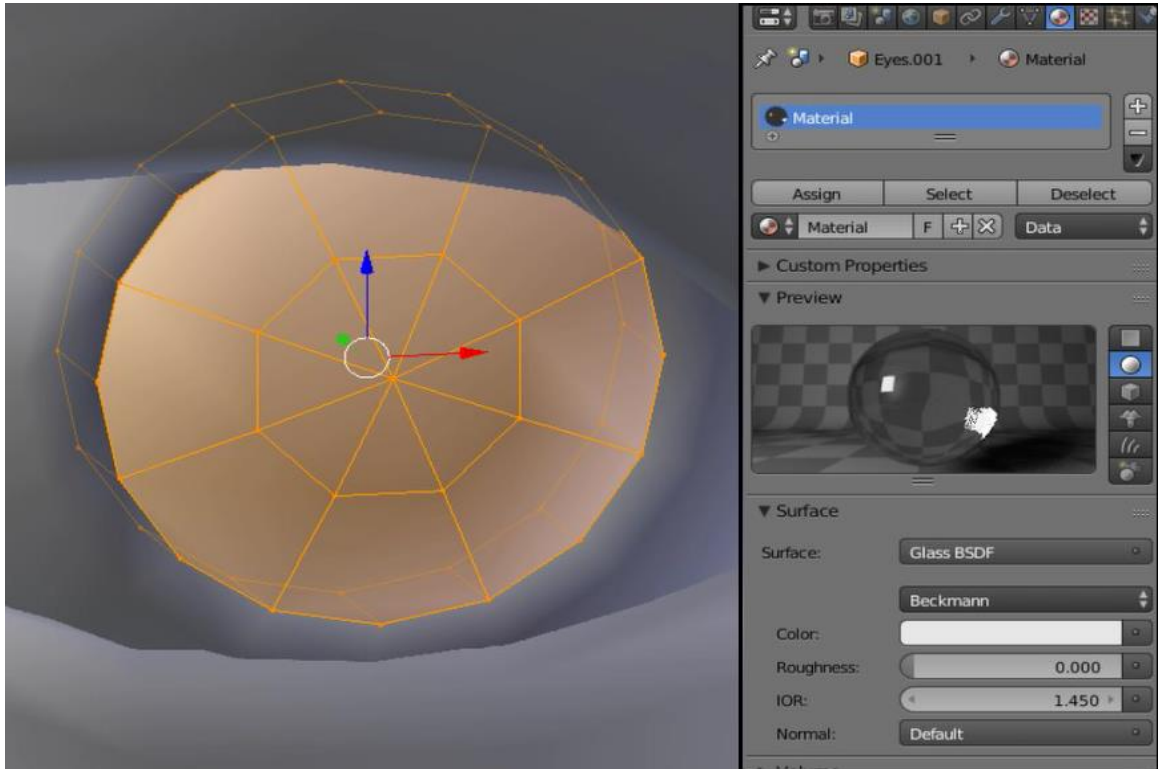


Image 62: Adding the Glass material.

In some cases, the modeler may wish to add eyelashes to the character. These can be done by selecting edges from around the eye and extruding them outwards. It is good to look at photographs as reference to make sure the eyelashes start from the right place and curve naturally. The eyelashes themselves will be created using a texture and an alpha map which will be looked at more closely in the future chapters. Image 63 shows a render image of the finished eyeball along with the eyelashes without a texture.

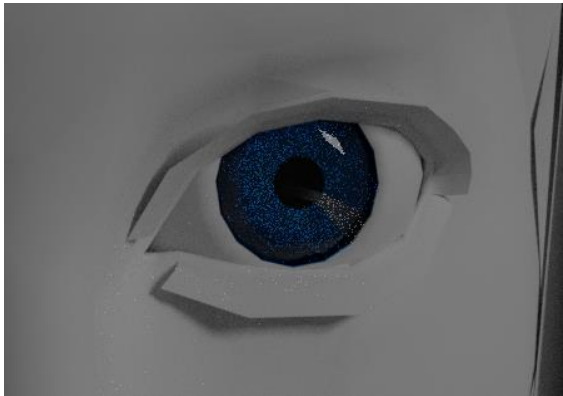


Image 63: The glass material looks peculiar in the render of Blender, but will look more natural when viewed in an engine that supports it.

Creating the top of the head begins by selecting and extruding the edges above the character's eyes. For Aroleir, two extrusions were deemed suitable. The sides of the forehead are then extruded to the side and then again towards the middle. This will create a strap-like ring of polygons behind the face. The Clipping option of the Mirror modifier needs to be turned on so the vertices on the back of the head will be merged automatically.

Once the extrusion has been connected, the modeler needs to add some subdivisions to it so the shape can be rounded out. Since Aroleir will have long hair, the back of the head does not need to have too many polygons. For Aroleir, one subdivision was added to the back polygons and two to the sides. These were rounded out to match the references. Once this is done, the forehead edges next to the extruded edges will also be extruded towards the middle. They need to be subdivided to have the same amount of edges as the strap underneath. The vertices of these two shapes then need to be merged (Alt + M) and rounded out. This process is repeated until the cap on the top of the head is two edges wide when not counting the Mirrored side. The steps after this are better explained with pictures. Image 64 goes through the steps above as well as capping the top of the head.

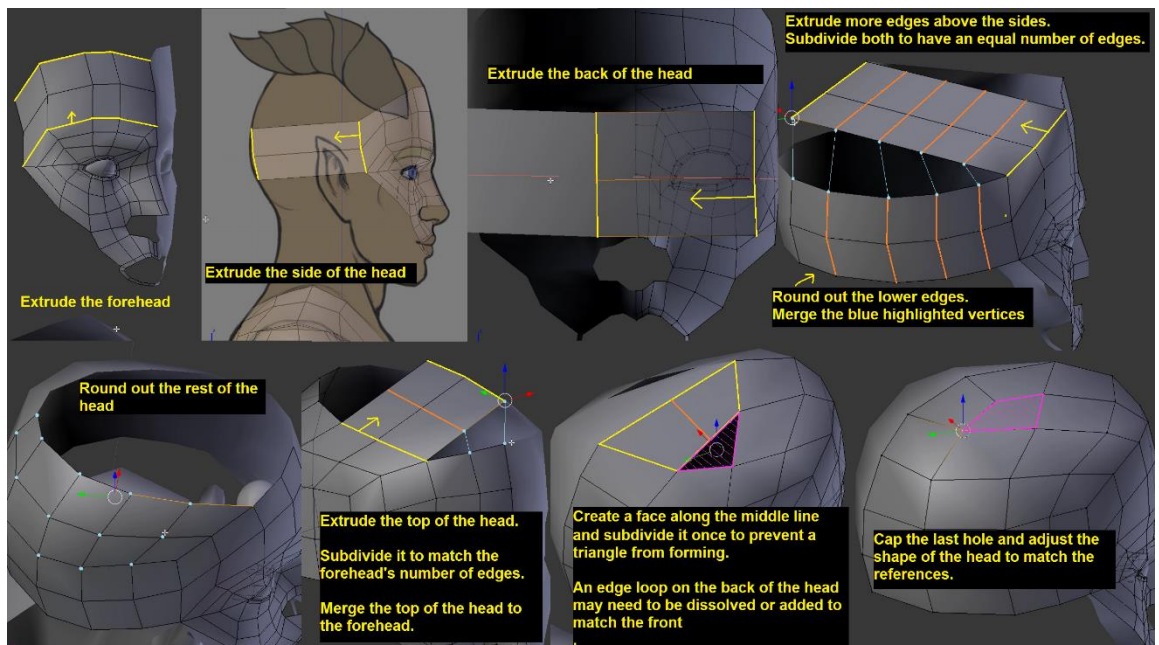


Image 64: Creating the back and top of the head.

To create the bottom of the face, the outline of the jaw is created by extruding the edge that would be right before the ear. It is good to adjust the edge already at this point so it runs just before where the ear begins. The extrusions should be done from the side first and then matched to the front reference by moving the vertices along the X-axis only. This way the two views will match even after the changes. With the Mirror modifier's Clipping option again, the last vertices of the jaw line can be merged together simply by dragging them to the middle.

The jaw line is connected to the lower lip by creating faces between them. The middle edge of the lower lip and the jaw are selected and a face is created with the F key. This is repeated with the edges next to the first one until there is only the edge of the side of the mouth left. Once the lip and the jaw are connected, some edge loops are added to them so the chin can be shaped from the side view. The modeler should make sure that the amount of edge loops on the chin matches the amount of edge loops on the top of the hole on the cheek. If too many edge loops are added, they can be later removed.

To fill in the hole on the cheek, the top edges of it are selected and extruded down. Edge loops are added to the extrusion sideways so the amount of edges matches the sides of the holes. Once this is done, the vertices are merged together so the hole is covered. Just covering the hole like this though leaves the character's cheeks very hollow in shape. Rounding the cheeks is a step that is easily overlooked while in orthographic view, but looking at the character while in perspective view shows just how unnatural the cheeks look. The vertices of the cheeks need to be moved outwards to make the character's cheeks look fuller. Additional character references and photos are very helpful in this phase.

In Aroleir's case, his upper lip was found too simple at this point and an extra edge loop needed to be added. Adding the loop after finishing the cheek though would cause the loop to run across the cheek and add unneeded edges all around. It is preferable to have the extra edge loop circle around the lips as the extra loops will make mouth animations smoother. Refer to Image 65 for progress shots of all these steps along with a visual guide to adding the edge loop around the mouth.

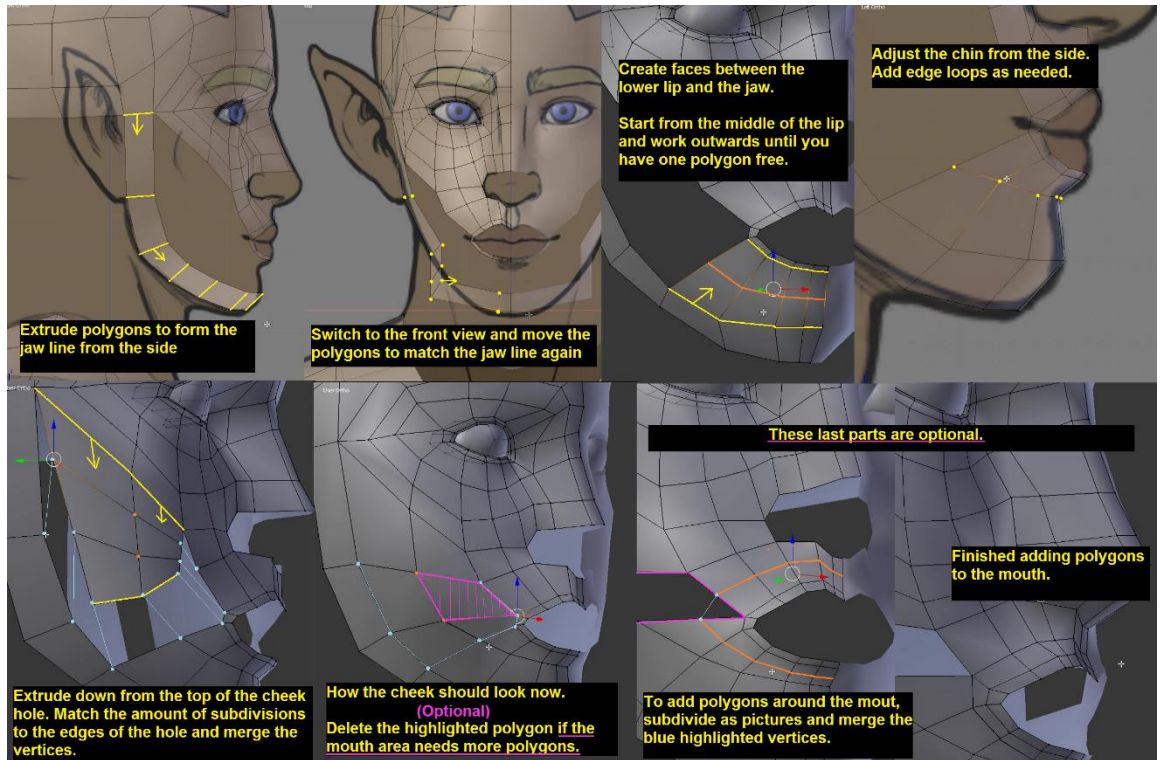


Image 65: Creating the lower face. The last three pictures the optional steps to adding extra edge loops around the mouth after finishing the cheek.

Creating the tip of the nose begins by connecting the edges on the upper and lower edge of the hole with new faces (F). Depending on the number of polygons above and below the nose hole, the new faces may or not leave a triangle-shaped hole to the side of the nose. In Aroleir's case this triangle was turned into a quad by subdividing both the new faces and by adding a new edge loop to the cheek. Once the hole has been covered, it is time to adjust the nose's shape to match the references. Some character models may not require holes for nostrils, but for Aroleir they were made as an extra detail. The nostrils were created by subdividing the faces of the lower half of the tip of the nose and extruding one of the polygons inside the nose. This left a small triangle at the corner of the nose. This triangle could be turned into a quad by adding an extra edge loop to the upper lip but the triangle was not seen as a problem worth adding extra edge loops to fix. Image 66 depicts the steps of creating the tip of the nose.

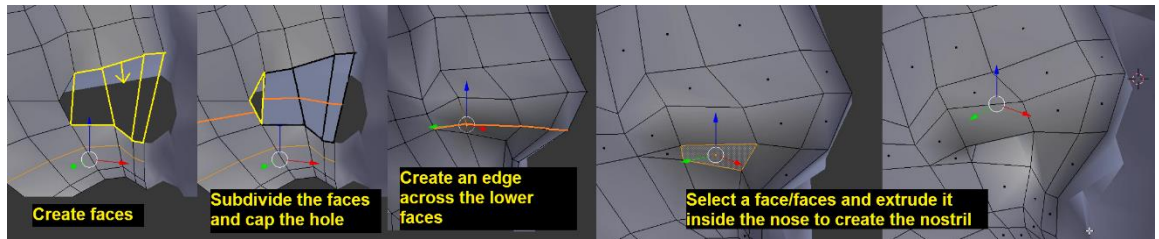


Image 66: Creating the tip of the nose and nostrils.

The lips are made by extruding the edges of the mouth hole without moving and then scaling the edges down. The vertices of the edge are then adjusted to line up with the parting line of the lips on the references. It is best to do this from the front first and to leave a small gap between the edges so the vertices will not get mixed up. Once the parting of the lips is done, another edge loop is added. The new edge loop's vertices are adjusted to make the lips rounder.

Not all characters need a mouth that can be opened and closed. In a game like Skyrim, non-playable characters have dialogue and even the player character does occasionally open their mouth to shout. If the character's mouth will be closed all the time, the mouth gap's vertices can be merged together with their matching counterpart. This though means the upper and lower lips should have the same number of vertices to begin with. As for creating a mouth that can be opened, the edges of the mouth hole are extruded inwards. The first extrusion will determine the thickness of the lips and thus should not extend too far into the mouth. The edges are then extruded further inside the mouth and scaled larger so the inside of the mouth is wider than the actual mouth itself. Once the inside of the mouth has been extruded far enough, the back of the mouth is capped. The bottom of the chamber can be raised up to create the illusion of a tongue when the mouth is opened, but this was not done for Aroleir as the player will likely never look at his mouth that closely and the tongue would hardly show from behind his teeth. Image 67 shows the steps of creating the lips and the inside of the mouth.

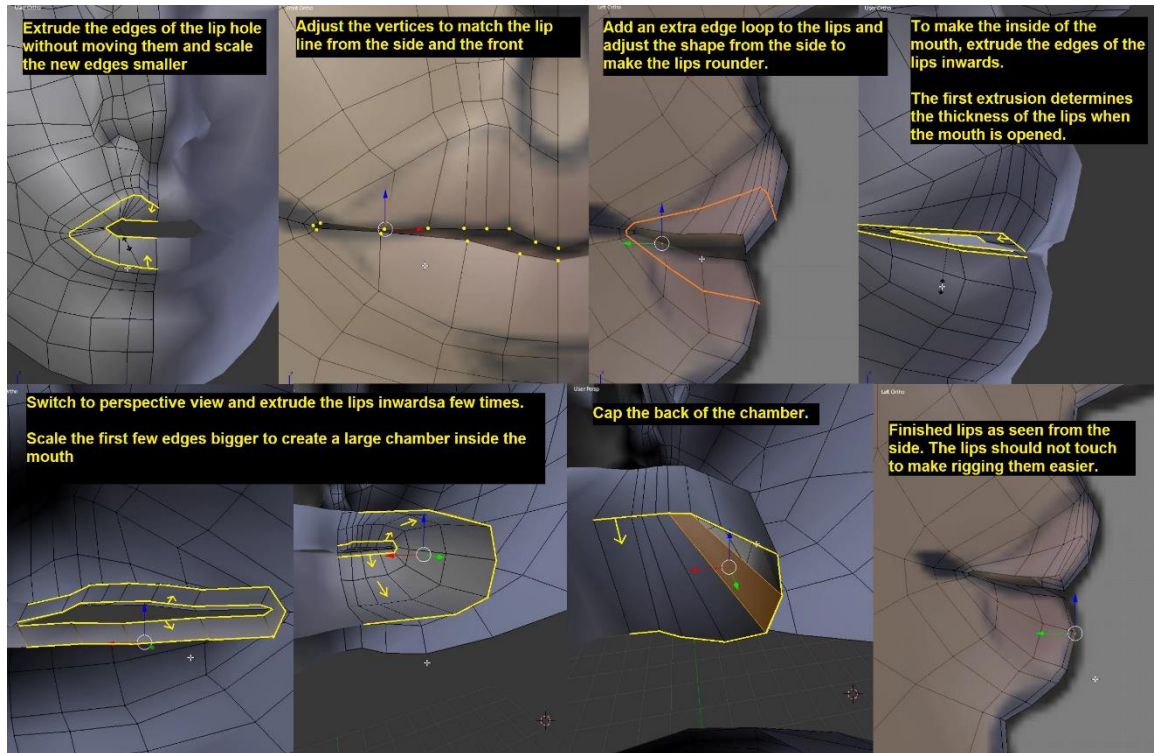


Image 67: Modeling the lips and the inside of the mouth. Not all characters need a mouth that can be opened.

When creating a character with an openable mouth, teeth are important. They are a small detail, but help convey emotion in facial animation even if there would be no dialogue. Making human teeth is quite simple as they do not require much detail in the modeling phase. The first step is to create a single plane in Edit mode and to scale it down so it is about the size of a front tooth. This will be the bottom of the tooth. It will also make the editing of the teeth easier if the polygons are placed on the side where the Mirror modifier creates the copy of the body. This way the teeth can easily be selected without accidentally selecting parts of the face.

The side will be extruded once the bottoms of all the teeth are done. The plane is placed in the middle so it will merge together with its mirrored counterpart. The other teeth are then extruded to the side one by one. It is good to note that human teeth are thinner on the front than in the back of the mouth. When all the teeth are done, their outer edges are selected and extruded upwards so they clip through the front of the mouth chamber. This creates the visible sides of the teeth. The inner sides do not need to be extruded as they likely would not be seen in normal speech anyways. All the faces of the teeth are then selected, duplicated using the

button in the right toolbar of the 3D viewport, and rotated 180 degrees along the Y-axis (R + 180 + Y). The new teeth are moved downwards so there is a small gap between them and the first row. As a last finishing touch, the lower teeth are scaled down as the lower row of teeth on humans is usually narrower than the top row. On most people the lower teeth are also just behind the top teeth. Image 68 shows the finished teeth on Aroleir.

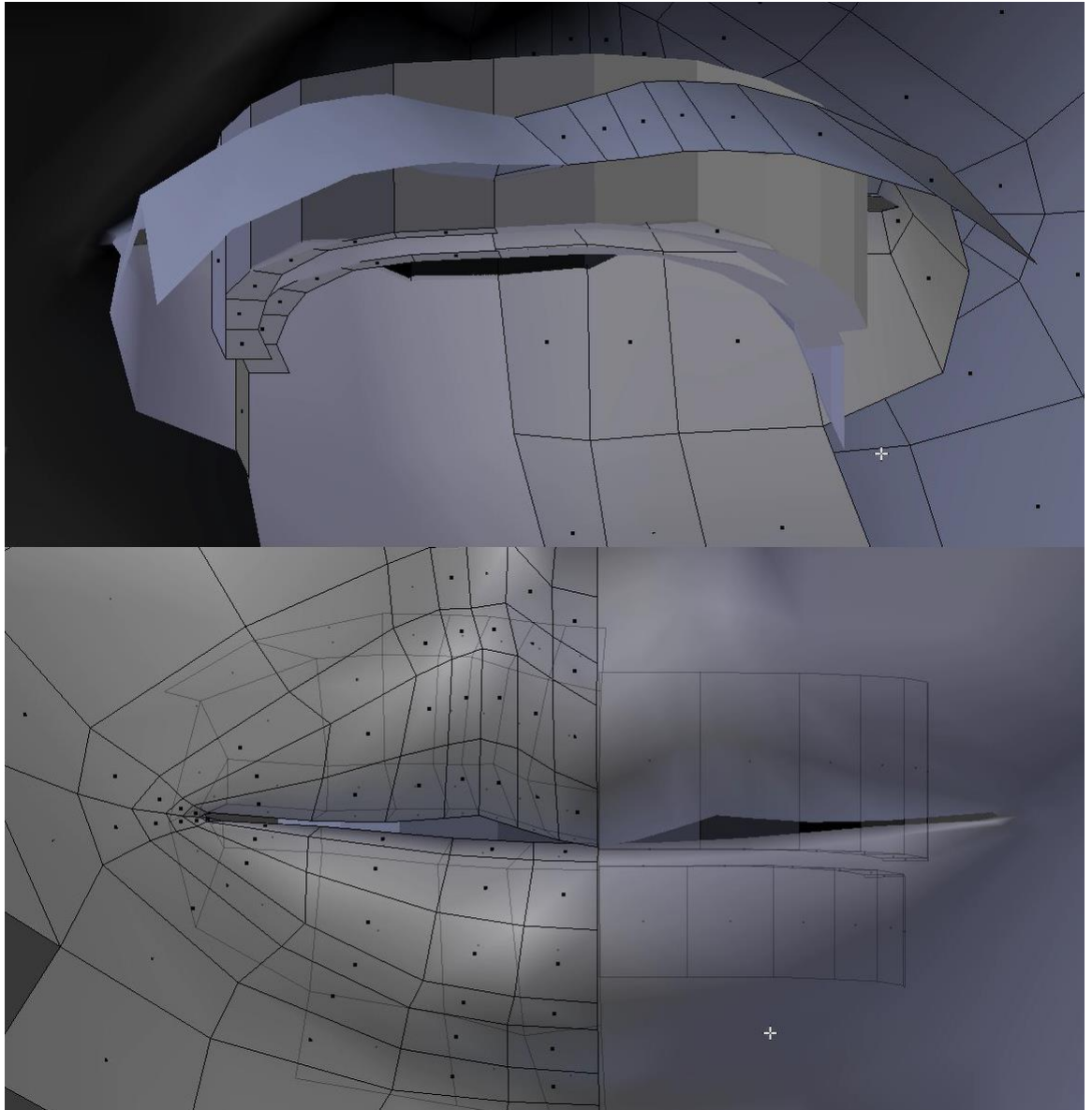


Image 68: One way to adding simple teeth into a character's mouth. Note how the teeth were modeled on the side opposite to the rest of the face.

To attach the head to the body, The back of the neck is first extruded down. The edge loops here should line up with the ones coming from the face until they reach

the corner of the jaw. After this, the base of the ear is determined by adding a thin edge loop just behind the jaw. The vertices of this new edge loop are then rounded back slightly to mark the area where the ear will be extruded from later.

More edges are extruded from the back of the neck and these are the ones that will be merged to the body. If the number of polygons on the back of the neck and the body don't match, edge loops can be added or removed to suitable places. On Aroleir, one edge loop was added to his arm and another to his chest. The one on his arm stops at the triangle of the elbow joint and the one added to the chest was merged to the middle line so it formed a narrow triangle that was hidden in the crotch. This way neither of the new edge loops ran all the way around but especially the front edges allowed to make Aroleir's chest and throat look more natural. Meanwhile, if the head has less edges than the body, extra edge loops can be added to the head and parts of them can be merged to the other edge loops on the top of the head. This way the new edge loops will not add too many extra polygons and the triangles can be hidden under the character's hair. The edge loop behind the ear will later be merged to the edge next to it.

Once the number of polygons on the back of the neck and the back of the body match, the vertices can be merged together (Alt + M). Next, the edge of the lowest neck polygon is extruded along the edge of the neck hole so the amount of faces matches the hole. The top edges of this new ring of polygons is then extruded upwards and merged to the side until there is one edge separating it from the jaw line. Image 69 shows how the throat gap should look at this point.

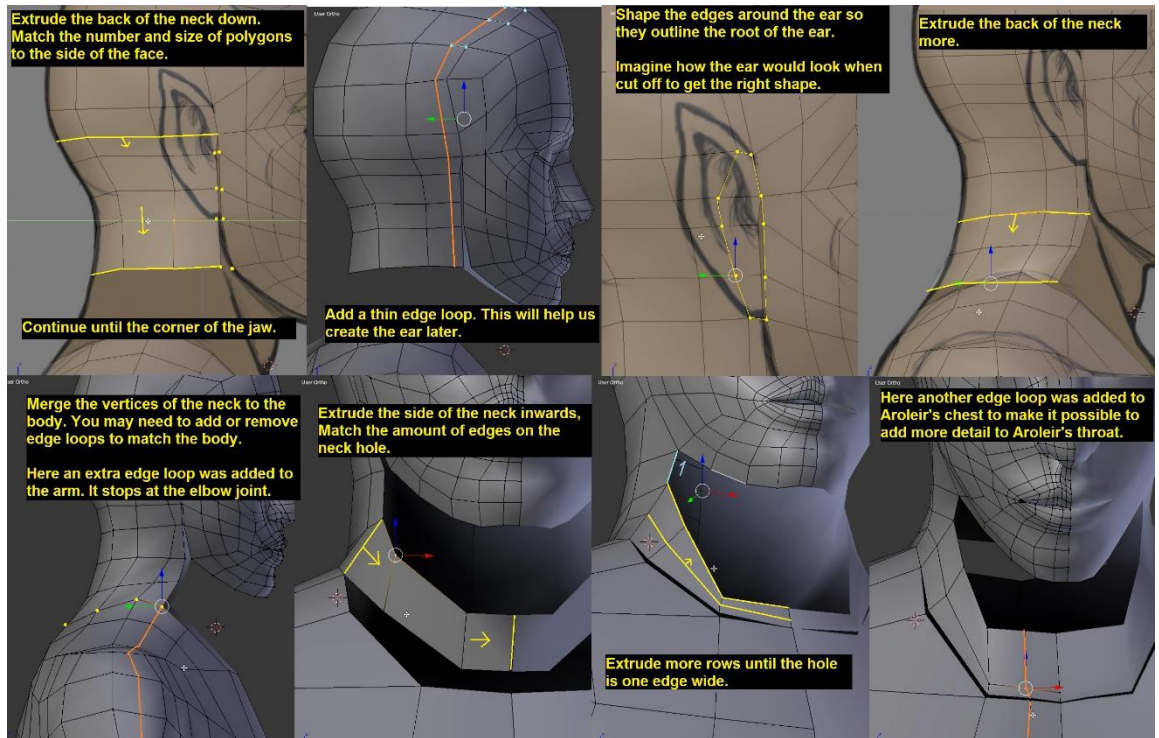


Image 69: Preparing to add the ear and shaping the back of the neck.

To cap the throat hole, the edges on both sides need to be connected with faces. Creating these faces should start from the middle of the throat and continue outwards. Having a triangle on the side of the throat at this point is fine as it will be used to add extra polygons to the throat. If there are significantly more edges on the bottom of the jaw than on the bottom of the neck hole, some triangles can be hidden under the chin. The steps taken past this point are quite complex and difficult to explain without pictures so image 70 goes through all the changes. The goal here is to have the throat match the side reference while maintaining a clean topology. In Aroleir's case, for example, the extra edge loop made for the ear was here forced to curve under the jaw to give more polygons for the throat. This process though can be very different depending on the number of polygons the character model has.

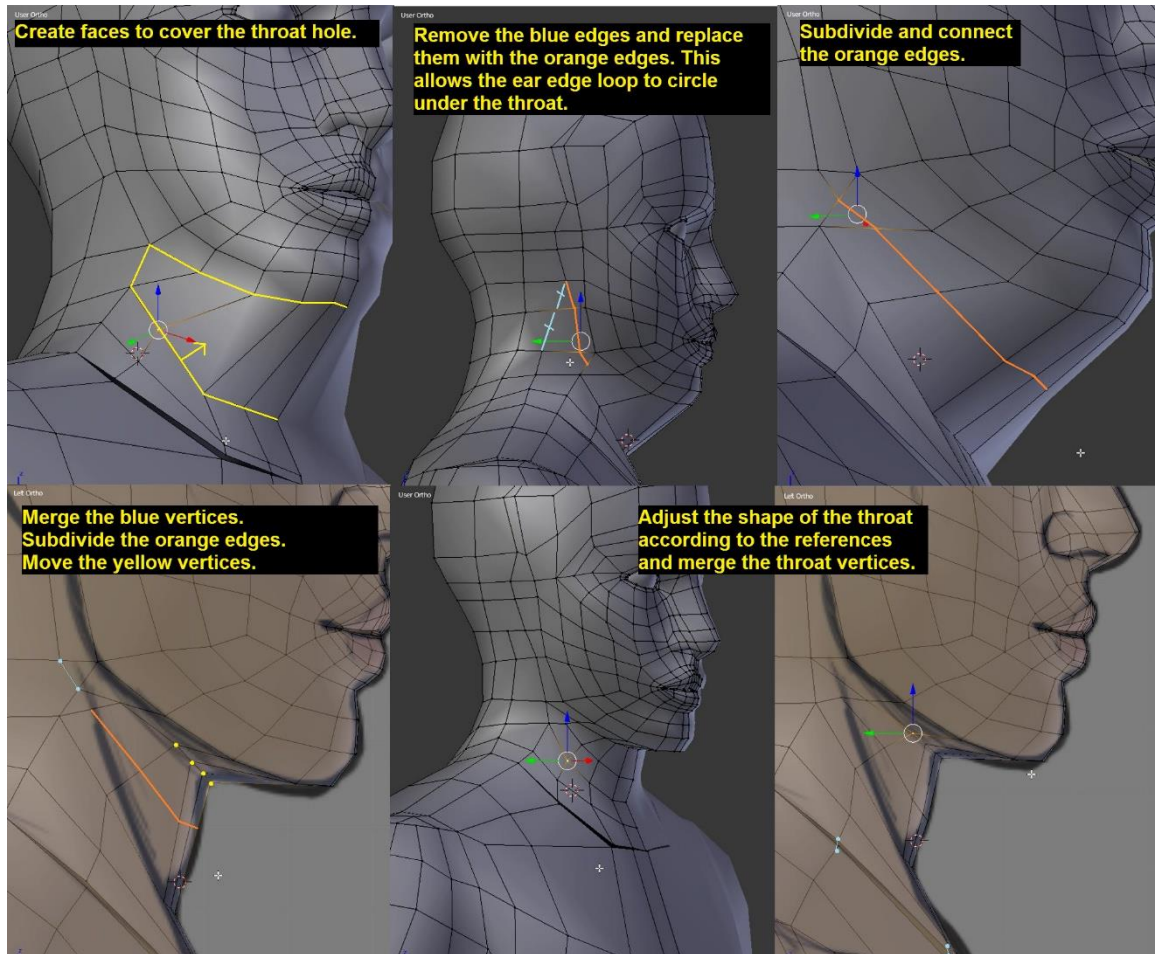


Image 70: Capping and shaping the throat.

The last thing to add to the nude character are their ears. The process of creating the ears varies greatly depending on the kind of ears the character has. Aroleir has elven ears that stick out to the sides like so-called “jug ears”. For low-poly elven ears, most of the detail will be painted on the textures. Thus the main task is getting the basic shape of the ear down without adding too much unnecessary detail.

For Aroleir, the edges that were shaped with the extra edge loop before were now extruded outwards three times. The vertices of the ear were then first adjusted from the front as for Aroleir the front view of the ears was more important than the side view. Once the silhouette was fine from the front, the view was rotated above the character and the edge loops of the ear were moved backwards so the ear would bend back slightly. The back of the ear was also rounded out more as it

looked too flat from the side. The inside of the ear was also pulled backwards to give the ear a realistic, curved shape.

It is possible that when extruding the ear, the tool will create extra faces inside the ear. If this happens, it will show when the ear is curved inwards as the polygons inside the ear will clip through the other polygons. These extra polygons inside the ear can be removed by toggling the “Limit Selection to Visible” button off and selecting the excess polygons. Image 71 shows how the ears were created on Aroleir. With the ears done, the nude character model is finished.

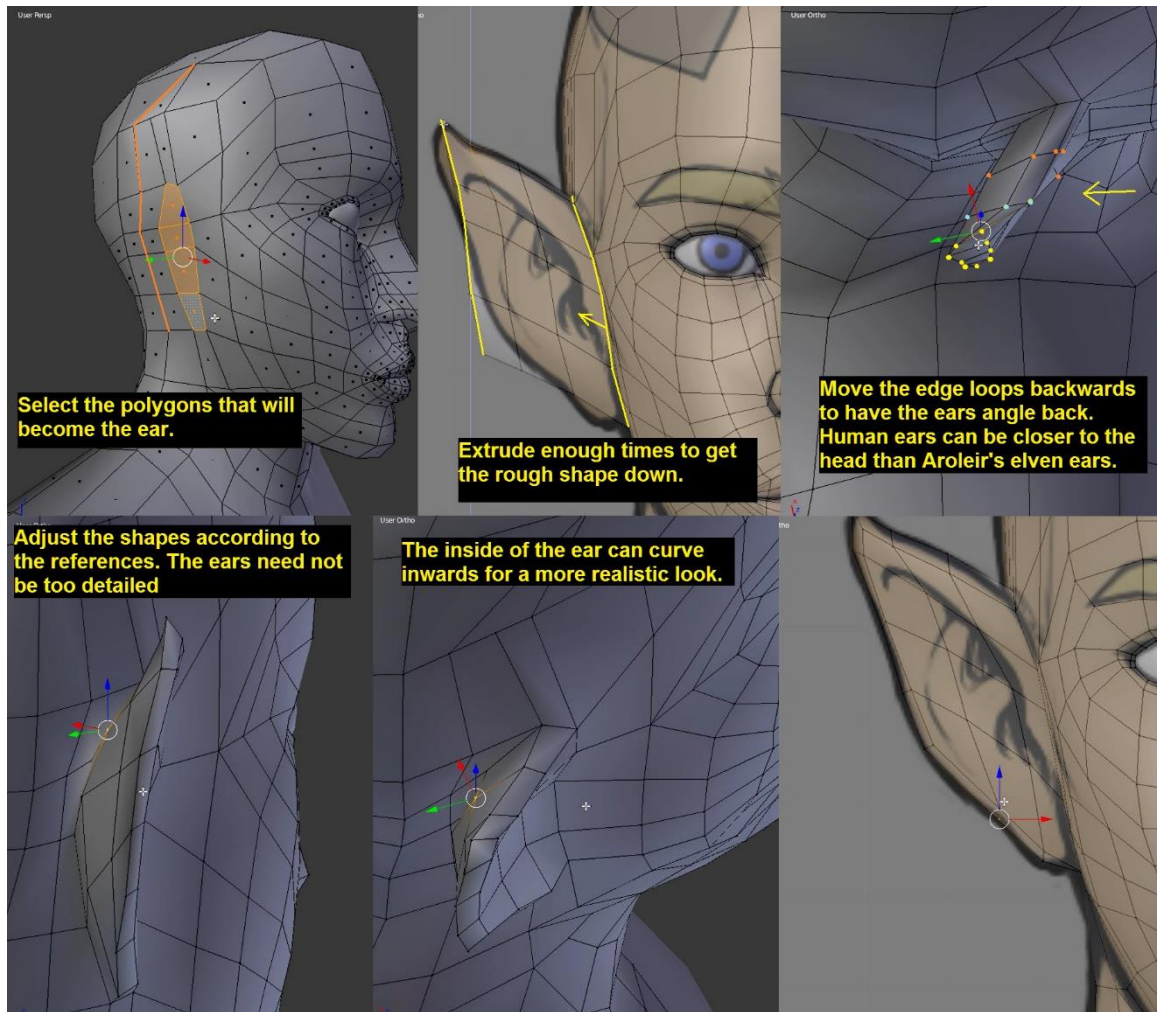


Image 71: Creating the ears. The first picture also features the last set of edges added to the area behind Aroleir's ears.

5.2.6 Clothes

There are several methods to creating clothes for a 3D character. One way to creating clothes for a character is making a duplicate of the whole body and editing the clothes from the body. The character's clothed parts are widened to show they are wearing clothes and certain details are extruded out directly from the body. Some details made like this could be the edges of a long coat or the sleeves of a shirt. As an example of a character with clothes made with his method is an earlier 3D character model created as an example for my seminar paper in late 2016. Image 72 shows the unclothed and clothed versions of the character's waist. The edges of the coat were extruded down from the middle of the waist so they were both a part of the same object.



Image 72: A coat edge made by extruding faces from a character's waist.

With Aroleir, the clothes were mostly modeled by duplicating parts of the nude body, scaling them up and adjusting the shape of the parts to match the clothes. This way the level of detail on the clothes remains similar to that on the character's body. Some pieces, like his pauldrons, though were modeled from scratch as they

were much simpler shapes than the character's body. Aroleir's fur bracers can be used as a simple example of the method described first.

The first step in creating Aroleir's bracers was to select the polygons that the vambrace would cover. These faces were then duplicated and separated (P + "Selection") to a different object. This way the unclothed body of the character remains unchanged and the pieces of equipment can easily be hidden when needed. When the piece is a separate object from the body, it is also easier to edit without accidentally editing the body.

Once the parts were separated, the object was scaled up until its thickness matched the reference. The object's both ends were then moved to the places where the vambraces edges would be as scaling the piece up also makes it longer. The rough shape is then refined by moving the vertices so it matches the references. Image 73 shows how the vambrace looked through all these steps.

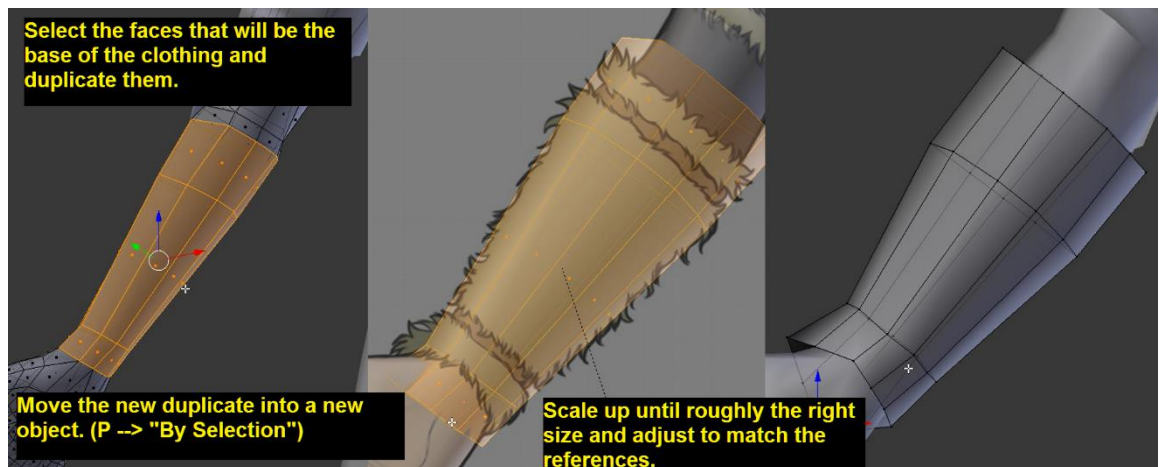


Image 73: The beginnings of creating a piece of clothing based on a part of the body.

Details were added to the bracer by selecting rings of polygons and extruding them inwards. Since the bracers are pieces of fur wrapped with two leather straps, the straps would cause dents in the fur. The top edges of the gaps were also moved outwards from the gap so the straps would not create a 90-degree angle to the fur. The edges of the fur there can also be adjusted so they are irregular, mimicking the random shapes of fur clothing. The back of the bracer also has a piece of steel plate armour which covers the leather straps. Thus, the extruded area does not

run completely around the bracer. Image 74 shows how the details on Aroleir's bracers were made.

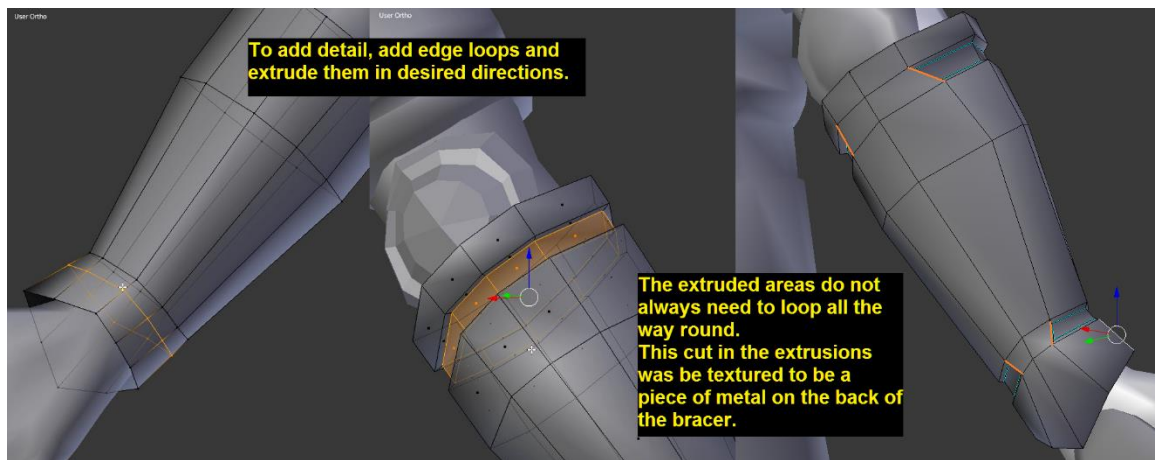


Image 74: Adding detail to the bracers.

Once the details are done, the equipment needs to be given proper 3D shape. This is done by selecting the edges of the ends of the piece, extruding them without moving the mouse and scaling them down so the edges clip through the body and are hidden inside the character's body. The hole should be hidden deep enough that it will not show even if the limbs change poses. As a final touch, the vertices of the equipment need to be moved so they do not clip with the body, not counting the faces that extend into the body as caps. Image 75 shows the final bracer and highlights the steps above.

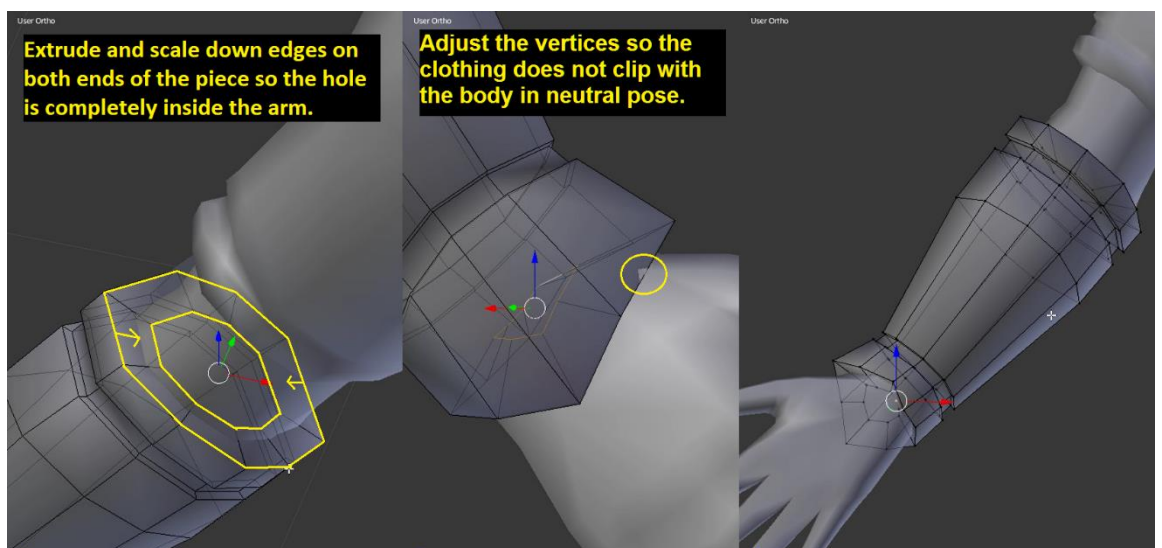


Image 75: Finishing off the bracers.

Several parts of a character's equipment can be created using the above method. In Aroleir's case, his bracers, shoes, pants and shirt were all created using this method. When it comes to parts with sharp turns in the shape, like the armpit of the shirt or the crotch area of pants, the modeler needs to pay extra attention not to have the clothing retain their shape when scaled up. Especially baggy pants are at risk of having polygons merge together unfavourably when the part is scaled up. Image 76 shows how the method is used in creating Aroleir's fur shirt. In the image, it can also be seen how the edges around the leather straps on the arms are irregular to enforce the random shapes of fur being pressed down.

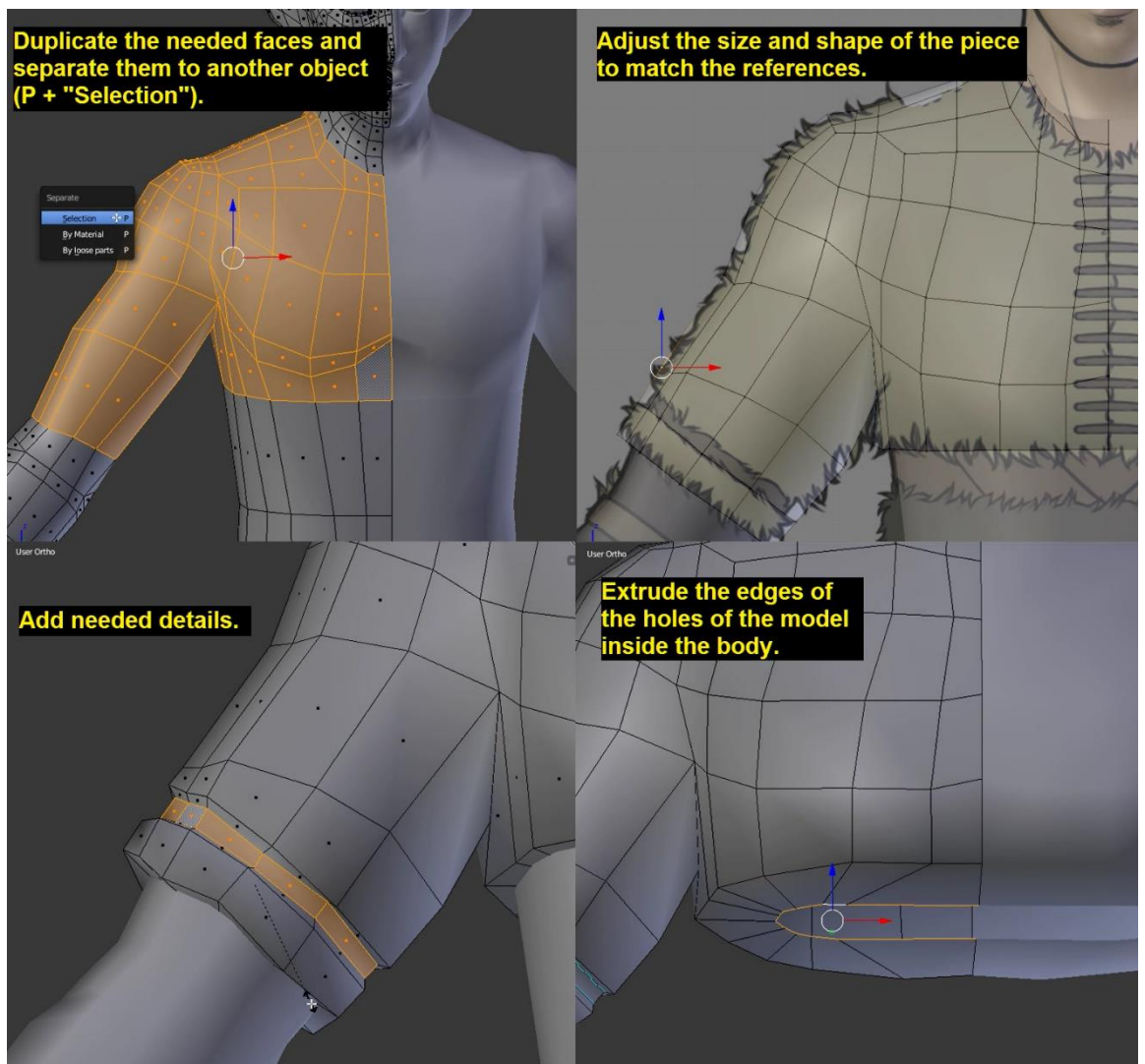


Image 76: This method can be used on many kinds of clothing. When making a shirt like this, the modeler should pay attention to the armpit so it retains its shape and the vertices do not merge together.

The base shape of the fur wrap around Aroleir's waist was also made with this method. First, one side of the wrap was modeled with the Mirror modifier still active. Edge loops were added and adjusted to line up with one side of the belts and the bottom hem was extruded down and shaped to have the rounded edge. While the other side of the wrap will be different, the mirrored version of the other side will provide a workable base. Image 77 shows the beginnings of creating the wrap.

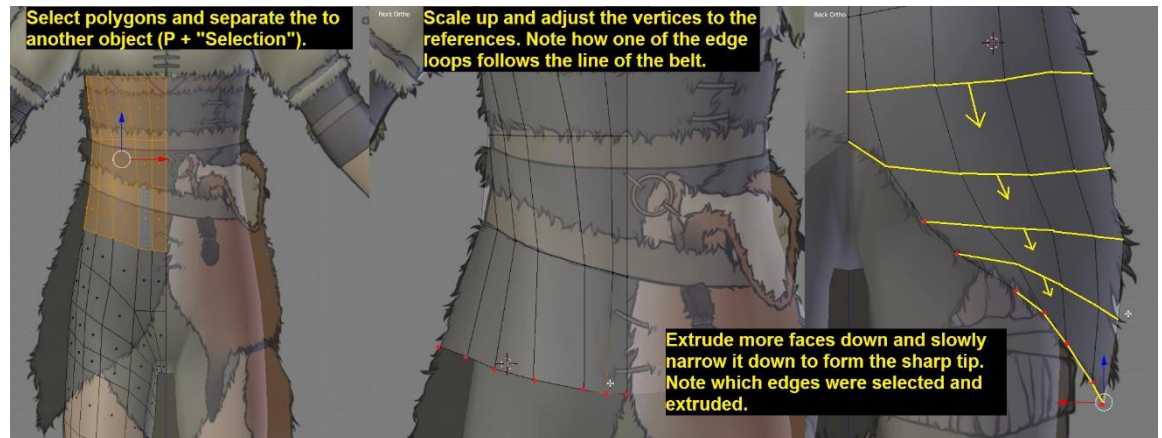


Image 77: Creating the base for the fur wrap. Even asymmetrical equipment will benefit from having a symmetrical base.

Once the base was done, the asymmetry needed to be added in. To do this, the Mirror modifier needed to be applied. The modifiers can be applied by pressing the "Apply" button in the modifier's settings though they can only be applied while in Object mode. With the Mirror modifier applied, the vertices on the mirrored side can be edited without it affecting the original side. Image 78 shows the finished fur wrap alongside an explanation of the steps taken to make all the different details of it.

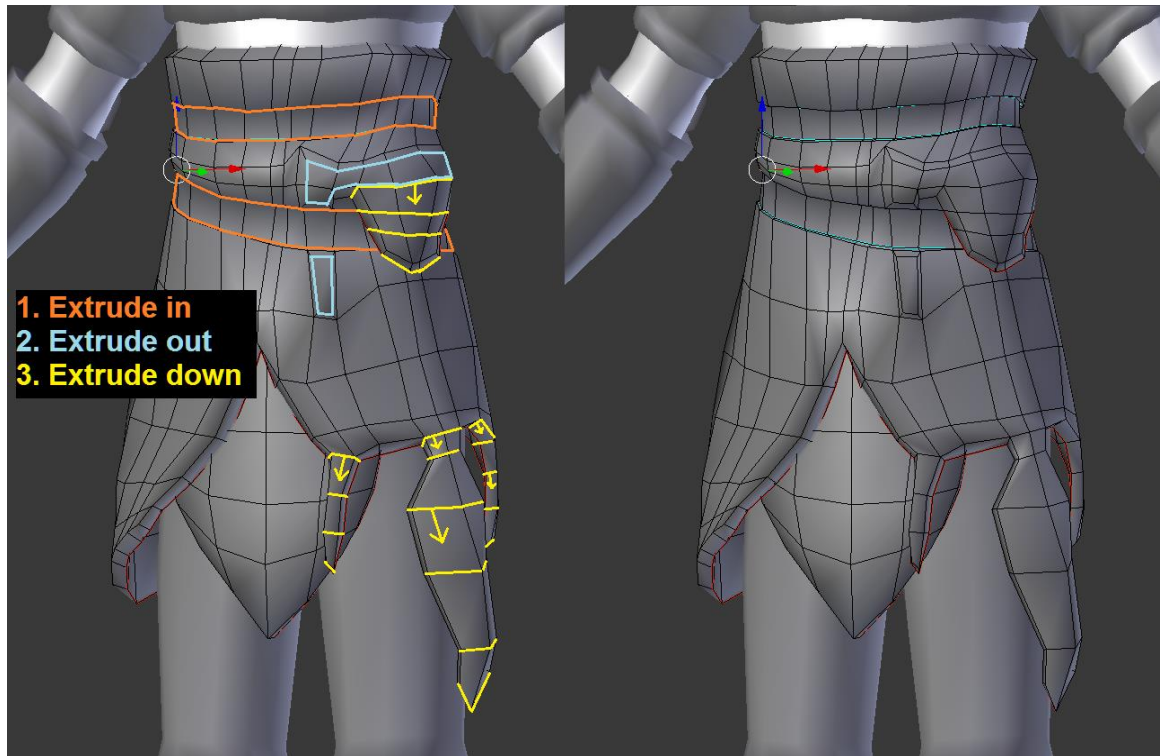


Image 78: The phases of adding detail to Aroleir's fur wrap. The flap on the front was added as a separate object instead of extruding it out.

Long pieces that can be seen from outside and inside, like the bottom edges of Aroleir's fur wrap or a long coat, should have some thickness added to them. A single layer of planes wrapping around the character's body rarely looks good. Meanwhile, using the same method as with pieces like the bracers and the shirt would be difficult to adjust to the shape of the piece if there is curvature.

An easy way to adding thickness to pieces like there is using the Solidify modifier. This modifier can be added to the piece at any point of the process but it is recommended to apply it after the piece is finished. After the modifier is applied, some of the polygons on the inside of the piece will need to be removed manually so only the polygons that would be seen are included. Image 79 shows the settings used with Aroleir's fur wrap.

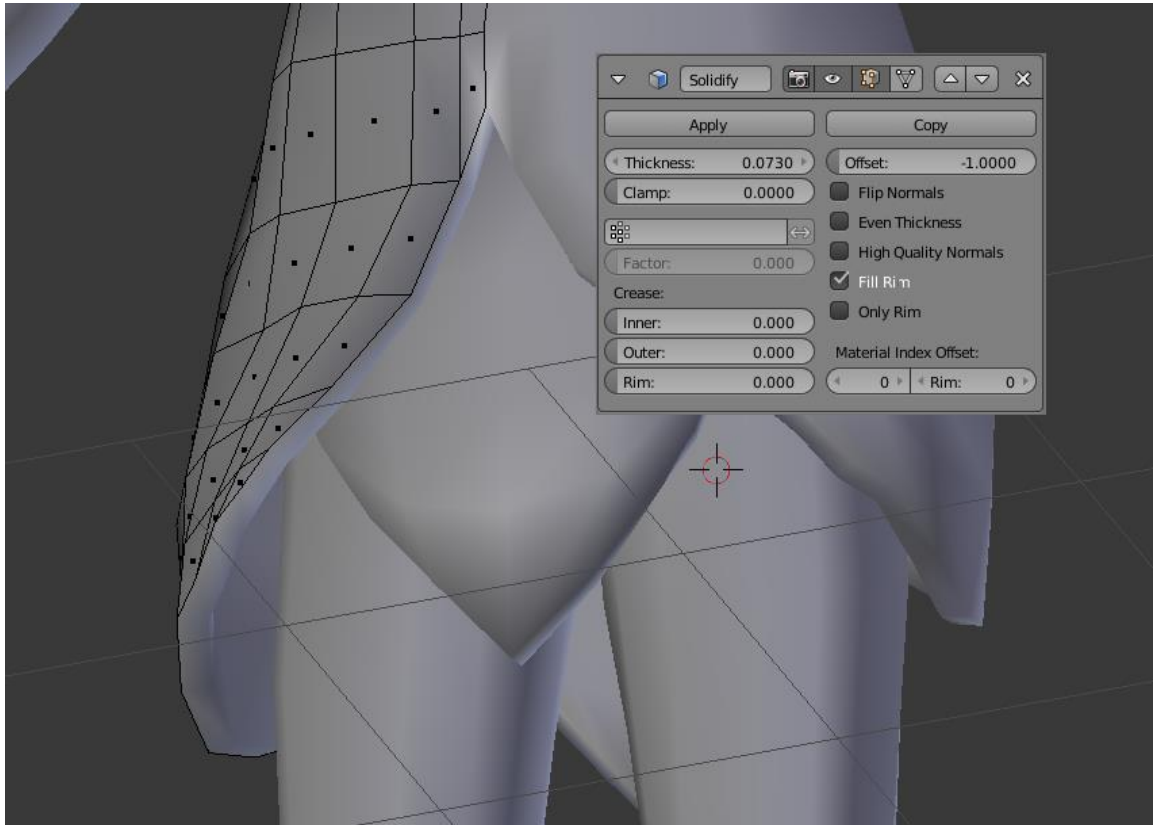


Image 79: The settings of the Solidify modifier. The main thing to pay attention to is the “Thickness” slider.

When it comes to simple armour like Aroleir’s pauldrons, the shape can easily be created from a plane. The plane was placed above Aroleir’s shoulder and it was subdivided three times along the Y-axis and five times along the X-axis. The vertices were then lowered so they formed a curved shape over Aroleir’s shoulder. As a final touch, the edges of the shape were extruded inwards so they clipped through Aroleir’s shoulder like was done with the bracers. The second piece of armour on the pauldron was also added with the same method. There are no progress shots from the creation of the pauldrons but image 80 shows how the finished pauldron model looked.

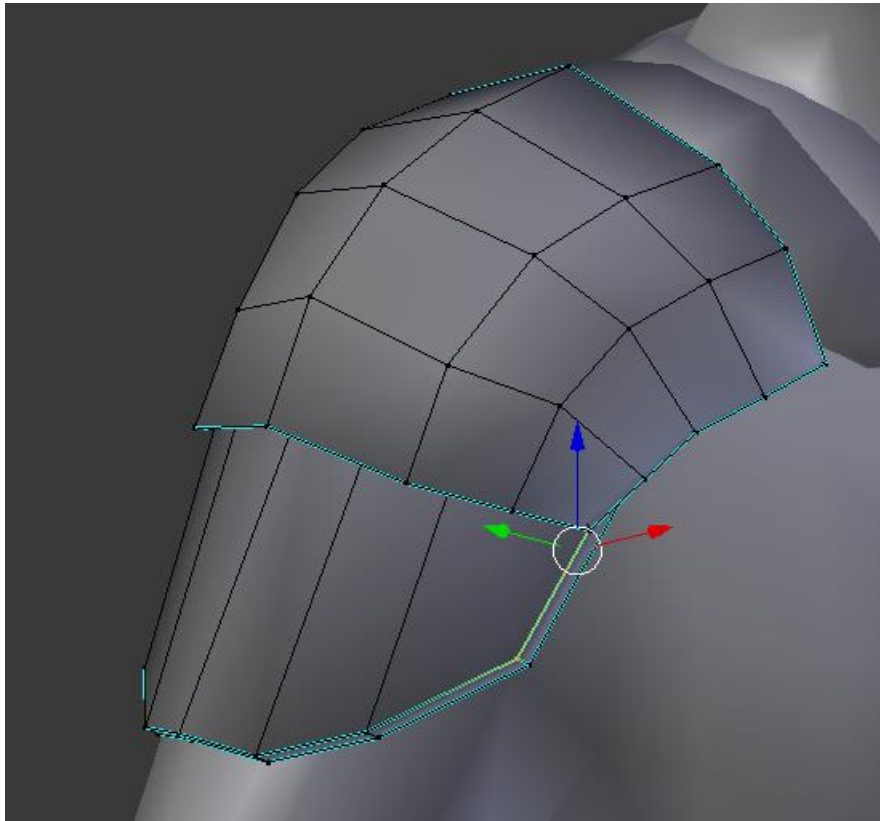


Image 80: Aroleir's finished pauldron model. It did not need many polygons as the detail will mostly come from the textures.

These are all the different techniques used in creating Aroleir's equipment from his shoes to his pauldrons. The small capelet around his shoulders was left unmodeled due to time restraints. His horns were also created at this point. The process of creating the horns was similar to creating the face in that first a single face was created and the shapes were formed by extruding the face as deemed fit. There are no process pictures of making the horns as most characters would not require them and creating horns this complex is a more advanced task than the rest of this guide. The finished set of equipment can be seen in image 81.



Image 81: The finished set of equipment and horns.

5.2.7 Hair

There are several ways to creating hair for a character. In some games the hair is a solid mesh like the clothes. Single strings of planes can still be used but they are still textured so there are no holes in the planes. A few examples of games with hair like this are Overwatch and any 3D game from The Legend of Zelda series. Another way to creating hair is to build a cluster of planes layered over each other and texturing the planes with transparency so the other planes can be seen through. This will usually create a more realistic look but is also takes more time to perfect. Image 82 shows how this latter technique was used in Final Fantasy XIII.



Image 82: Hair made for the character Lightning in Final Fantasy XIII. (Polycount 3)

For Aroleir, a mix of the two styles was used. Most of his hair is tucked under his clothes so the tips of the strands would not be seen anyway and using layered polygons there would be pointless. The strands that are pulled back to form the ponytail are also textured solid with only the tip of the ponytail having some transparency. This way the ponytail will be easier to animate with the cost of losing some of the realism. The free strands on the front through will be created by using two overlapping faces. These faces will have some transparency applied so they do not have just a solid colour.

Hair for Aroleir was created in three parts. The first part to create was all the hair that would be tucked under his clothes. Like his clothes, this part was created by duplicating and scaling up a part of the top of his head. The vertices were then adjusted to the references and the shape was rounded like when creating the nude body. Once the base shape was down, the outside edge was extruded and moved so it clipped with the head and neck. This way the inside of the solid hair model

could not be seen from any angle. The bottom edge was also adjusted the same way but that edge also needed to be rounded so it looked like the hair drooped above the neck hole of the shirt. Image 83 shows the progress shots of these steps.

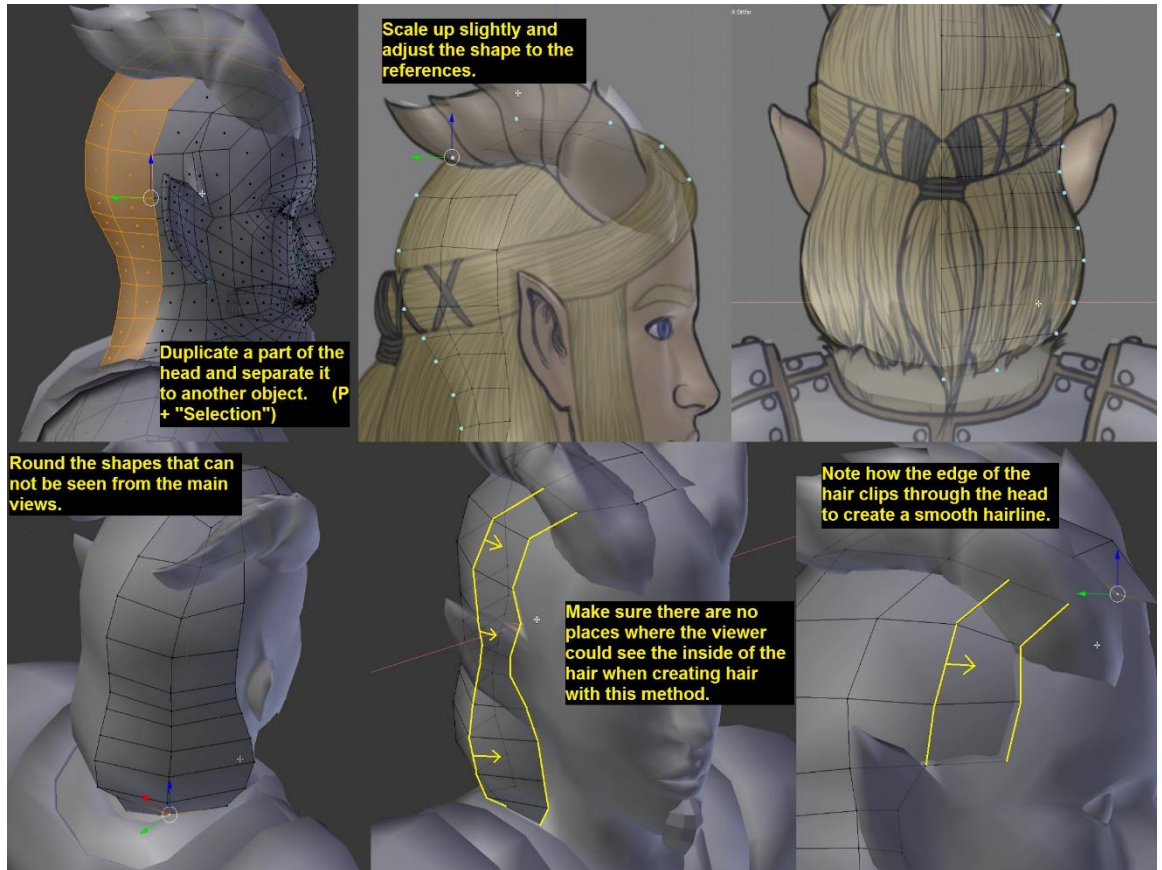


Image 83: Creating a solid mesh for the bottom layer of Aroleir's hair.

The second part to be created was the strands of hair that are wrapped around Aroleir's head and pulled back to form a ponytail. This part was first extruded from the base hair mesh to make sure it lines up perfectly. First, the edge at the top middle of the base mesh was extruded out several times. Each time an extrusion was made, the new edge was rotated so they together formed a loop that curved around the horn without clipping it. This line was continued until the last extrusion was merged to the middle line at the back of his head. The edge between the base mesh and the strand was then selected and split from each other (Y). The edge of the strand was then moved up and extruded down so it clipped through the base mesh.

Next, the strand was given a more rounded shape by adding an edge loop along its length and moving the new edge's vertices outwards. Once this was done, another edge loop was added to the spot where the ponytail would begin and the ponytail itself was extruded out. The ponytail's vertices were then adjusted to match the references. Image 84 illustrates these steps.

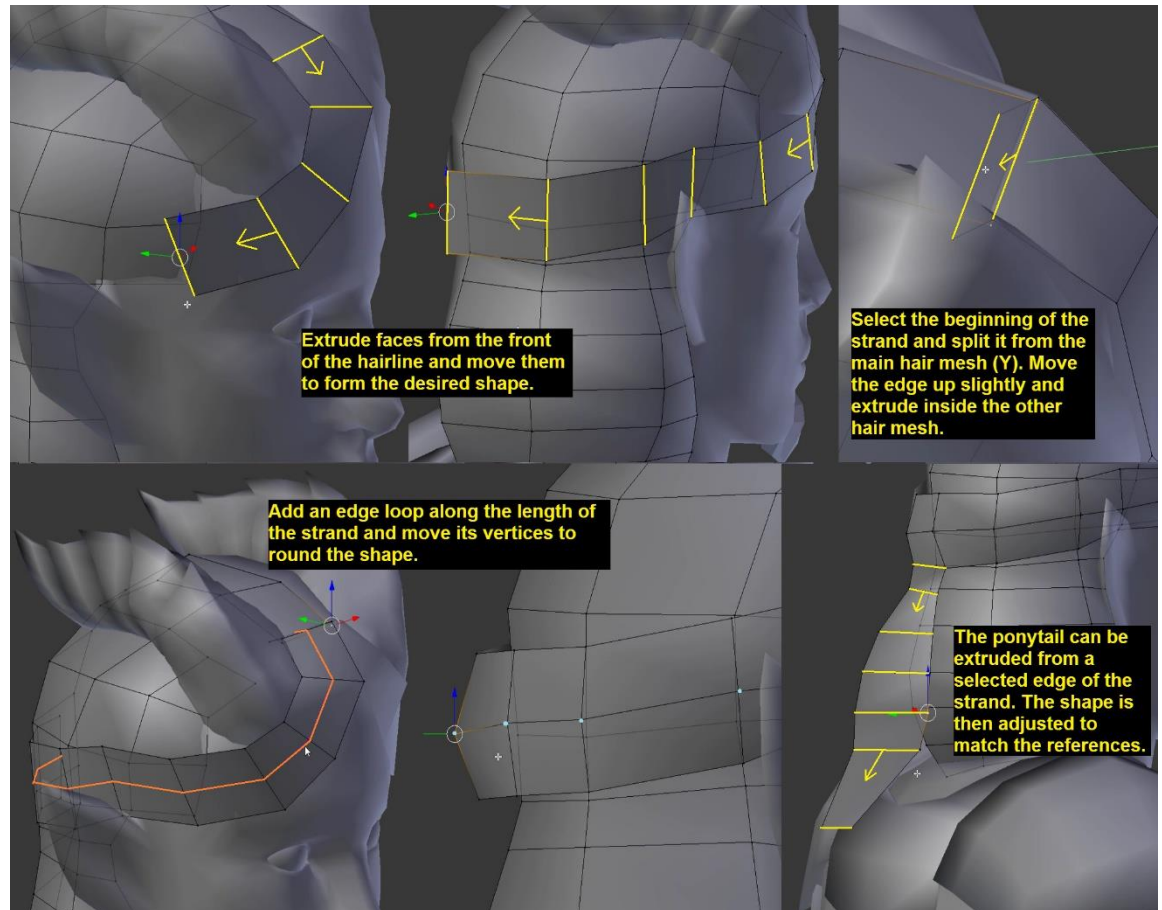


Image 84: Beginning to create the strands that are pulled back to form the ponytail.

While the base hair mesh works as it is, the ponytail and the strands forming it need a more solid shape. The easiest way to do this is to use the Solidify modifier once again. First, all the faces of the strand and the ponytail need to be selected and separated from the base mesh (P + "Selection"). If the faces would not be separated, it would be impossible to apply the Solidify modifier without affecting the base mesh. Once the part has been separated, the modifier can be added and its settings can be adjusted accordingly.

Before the Solidify modifier's changes are applied to the model, the shape of the hair needs to be refined. In Aroleir's case, this meant adding some extra edge loops to areas that needed smoother curves. Such areas were the ponytail and the front of the hair strand. The front of the strand curves around the horn so sharply that the corners would be obvious without the added edge loops. The ponytail also needed to be rounded out more as in the phase before it was shaped like a triangle rather than a rounded tube.

Once all the detail is in place, the Solidify modifier can be applied to the model in Object mode. Like before, this allows the vertices of the parts created by the modifier to be adjusted freely. Because of the curve along the length of the hair strand and the ponytail, the edges created by the modifier will curve inwards. The vertices of these edges need to be moved so the ponytail will be rounded properly. The inside of the long strand will also have excess vertices which have no effect on how the hair looks from the outside. These middle vertices should be merged to the vertices above them so the inside of the strand is a straight line of faces.

The tip of the ponytail was left open. The vertices could also be merged together but this way the length of the hairs is easier to figure out when painting the textures. When the ponytail would be textured, the textures would leave empty space between hair strands at the bottom. This would give the tip a fuzzy look instead of it having a solid tip. An example of a similar method being used in character hair would be Junkrat from Overwatch. The tips of Junkrat's hair are left transparent while the rest of the hair is modeled and textured solid. Image 85 describes all the steps above.

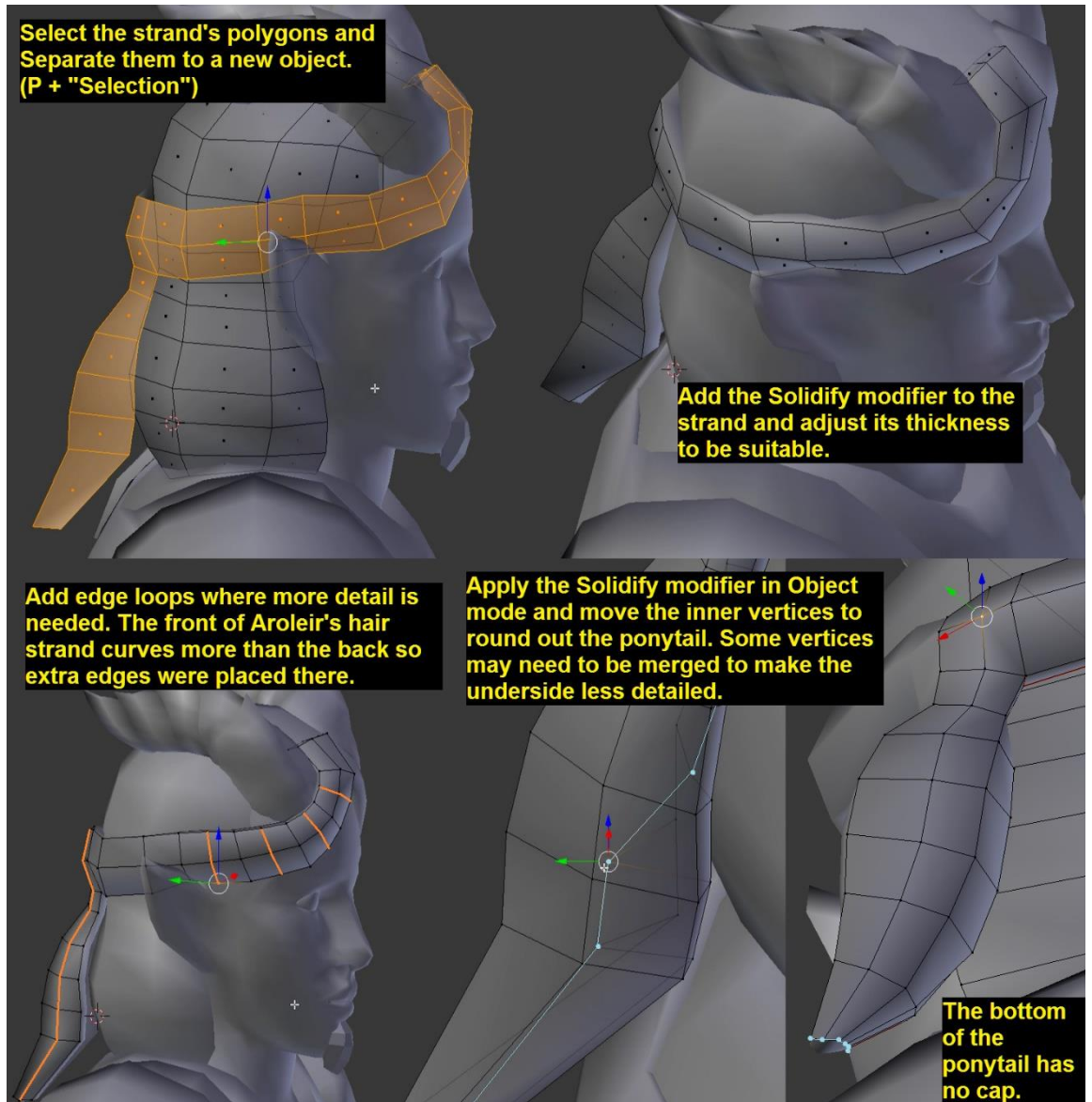


Image 85: Giving the ponytail thickness and rounding out the shapes.

The last part of the hair to model are the hair strands before Aroleir's ears. These strands will be made of two or more overlapping lines of polygons that are textured with some gaps between hairs. At this point only two are made. If more are needed later, they will be duplicated from these two so they will use the same texture. This will save both time and space on the UV layout.

The first step of creating these strands is to extrude an edge from the base hair mesh. This face is then separated to a new object (P + "Selection") and extruded in both directions to create a string of faces that are shaped according to the side reference. To give the strand more shape, an extra edge loop is added along its

length. The vertices of the strand are then adjusted from the front so the middle edges go along the outer edge of the hair strand in the reference and the front edge is moved to align with the inner edge of the hair strand. The back edge is placed somewhere between them without letting any of the edges clip through the face.

After the shape is done, the top edge of the hair strand is moved inwards so it clips through the base hair mesh. Because Aroleir's ponytail strands cover this area, in his case the connection of the front strands do not need to be too clean. Finally, the hair strand is duplicated. The copy is moved, rotated and reshaped so it does not clip through the top strand while still being visible from most front angles. Image 86 shows these steps more clearly than words can express.

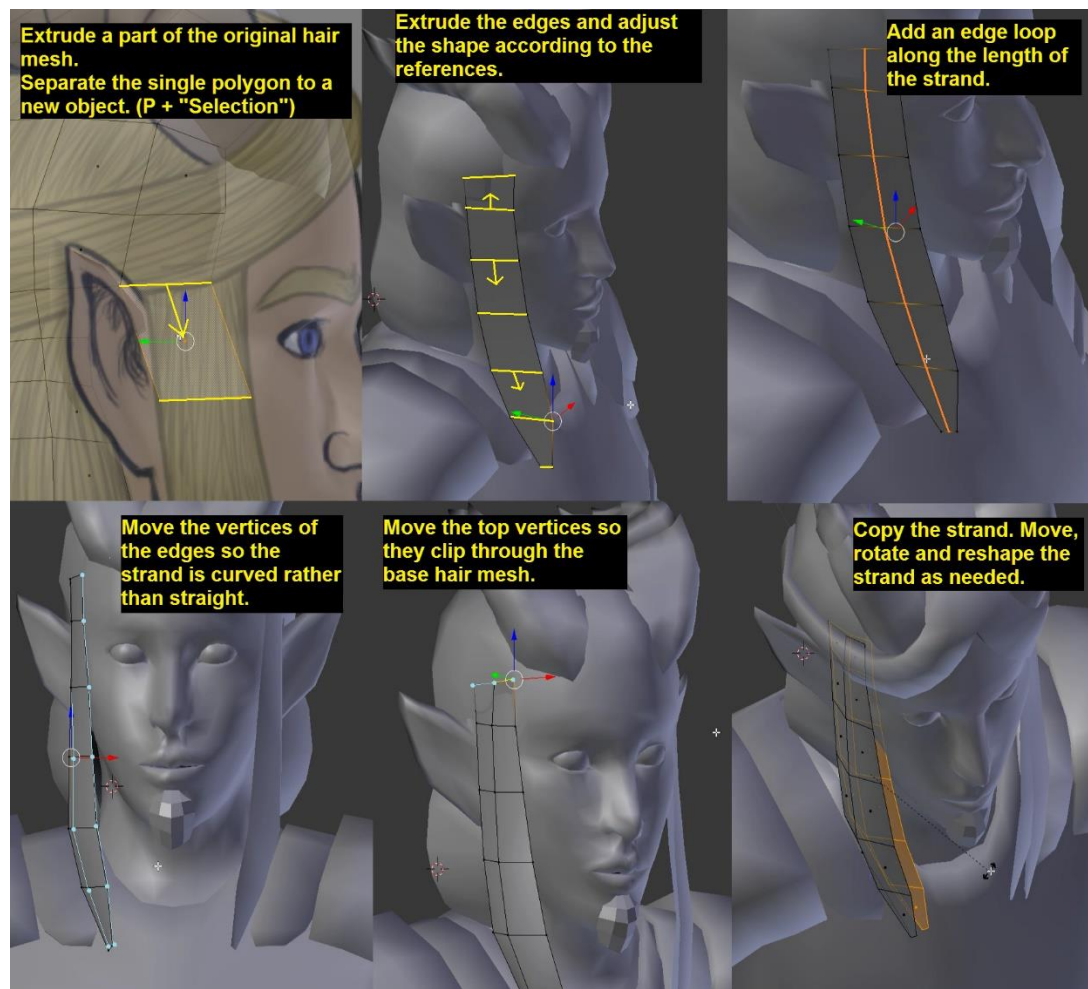


Image 86: Creating the hair strands on the front. This method can be used to create a full head of hair or even just an airy ponytail but for Aroleir that was not needed.

The last step is merging the three objects together. This is done by selecting the three objects in Object mode and joining them (Ctrl + J + “Join selected meshes”). With this, the hair mesh is done and the entire character model is ready for the next step; unwrapping. Once the unwrapping and texturing phases are done, more of the frontal hair strands may be duplicated from the existing ones. If this was done now, they all would need to be textured separately. Image 87 shows the finished hair mesh.

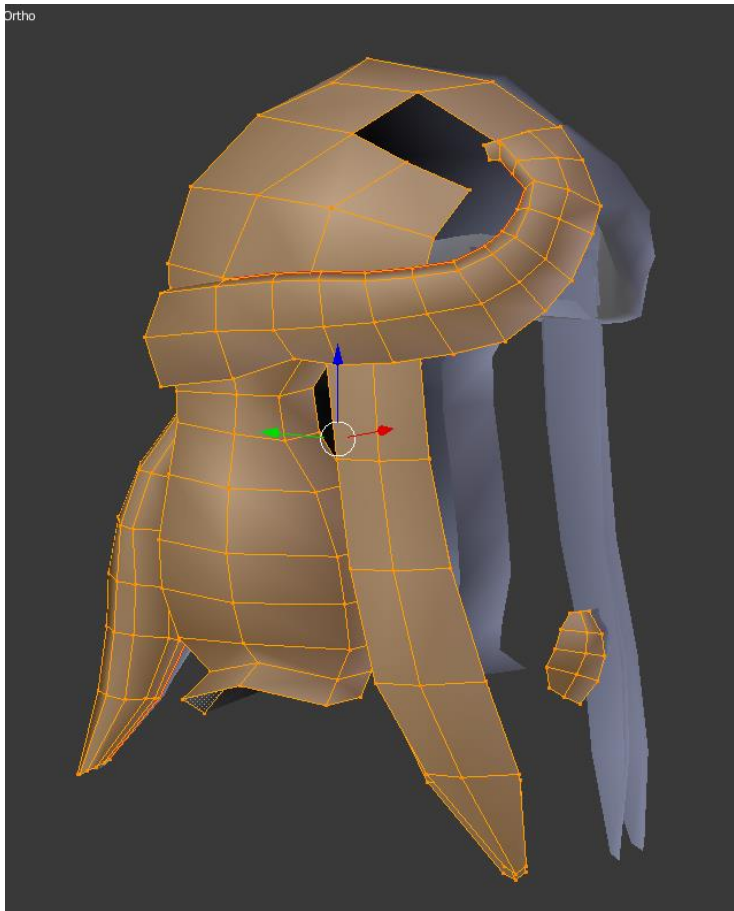


Image 87: The finished hair mesh. More frontal hair strands may be added after the texturing phase if deemed necessary.

Like with the hair, also the equipment can be merged into a single object if needed. On Aroleir, each piece of equipment was kept as a separate object so they could be equipped and unequipped freely. If the character though only has one set of equipment which never changes, joining objects allows the model to use fewer texture maps. It is good to save a separate version of the model before joining all

the meshes in case pieces of it are needed later. In the joined model, hidden polygons should be removed to make the model lighter. If it is not seen, it should not exist. With Aroleir, the removal of excess polygons would be done after texturing so different equipment combinations can be taken into consideration.

5.3 Unwrapping

Unwrapping a 3D model is a task of adding seams that allow the mesh to be laid down onto a flat surface without deformation while also hiding those seams into places where they cannot be seen. A good place to learn how to hide seams is real life clothing. While some clothing use seams as decoration, basic clothes with patterned fabric place them into places where they attract no attention and leave as much as possible of the fabric's pattern intact. As a bonus, when texturing clothes, the model's seams can be helpful to the artist who can paint a dark line on the edges of the seams. This way the modeler need not worry about hiding the seam while painting the textures. Another place where seams are useful are areas where the colour of an object changes suddenly. Seams give the model places where the colour can change sharply without the pixels being visible even with low-resolution textures.

Before unwrapping the model, the modeler should hide all the pieces of the model and keep only one piece visible at the time. Every separate object, every piece of equipment, needs to be unwrapped separately. With Aroleir, the first piece to be unwrapped was the eyeball. This shall be used as a step-by-step guide for unwrapping due to the simplicity of the object itself.

The first step to unwrapping a model is planning. The modeler should look at the model and try to imagine where the seams would be useful to the texture artist and where they are a necessary evil. On the eyes, examples of the seams being useful are the edges that run around the iris and the pupil. If a seam would not be placed around the iris, the colour of the iris would easily bleed into the sclera of the eye as a pixelated mess. Meanwhile, an example of necessary evil in unwrapping would be the seam that needs to run around the sclera of the eye. While most of

the eye is hidden, a seam around the widest part of the eye can still be visible if the character turns their eyes to the side. The seam could also be placed further behind the eyeball, but the further back it goes, the more the unwrap will distort the textures.

Once the modeler has some idea of where the seams should go, they should select the said seams and press the “Mark Seam” button in the “Shading / UVs” tab of the 3D Viewport. The seams need not be set all at once and they can be removed at any point. The object can be unwrapped even if all the seams are not in. All the modeler needs to do, is select all the faces of the object and pick the “Unwrap” option from the drop-down menu above the “Mark Seam” button. This will create an automatic UV layout of the unwrapped object. Image 88 shows the location of the seam and unwrap buttons.

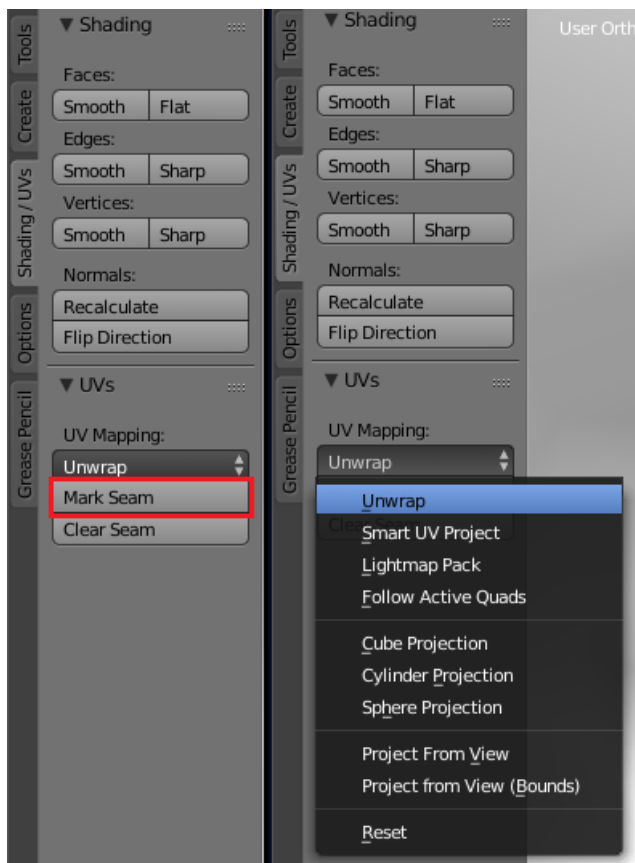



Image 88: The seam creation and unwrapping tools.

Once seams have been set and the object has been unwrapped, a new editor view needs to be created alongside the 3D viewport to make the work easier. In the left

corner of the editor view, top or bottom depending on settings, the editor type can be changed from 3D viewport to UV/Image Editor. If the object has not been unwrapped, the viewport will show a plain grid on a gray background. If the object has been unwrapped, its pieces have been automatically laid down within the boundaries of the grid. The parts of the unwrap are always in scale to each other if they have been unwrapped at the same time. This is all that needs to be done to unwrap an object in Blender but there are some tools that can be useful when working with more complex objects.

The pieces of an unwrap can be selected, scaled and moved as needed. The shortcuts are mostly the same as in the 3D viewport but, for example, vertices cannot be merged with the “Alt + M” combination. Instead, vertices can be welded together in the UV Editor with “W → Weld”. This will not alter the actual 3D model.

One important thing to note is the “Keep UV and edit mode mesh selection in sync” button at the bottom of the UV/Image Editor. This button is marked with the  symbol. When this button is toggled on, moving a vertex on a seam moves the corresponding vertex on all the unwrapped parts (Image 89). This can be useful if the modeler needs to figure out where a specific vertex of the UV map is located on the 3D model. Meanwhile, when the button is off, the UV/Image Editor will only display the vertices or faces that are selected on the 3D model. Only the selected areas of the 3D model can again be seen and edited on the UV/Image Editor. This will also allow for the vertices that mark the seams be moved without it affecting the other unwrapped parts. This is also the only way the full UV islands of the unwrap can be selected with one click.

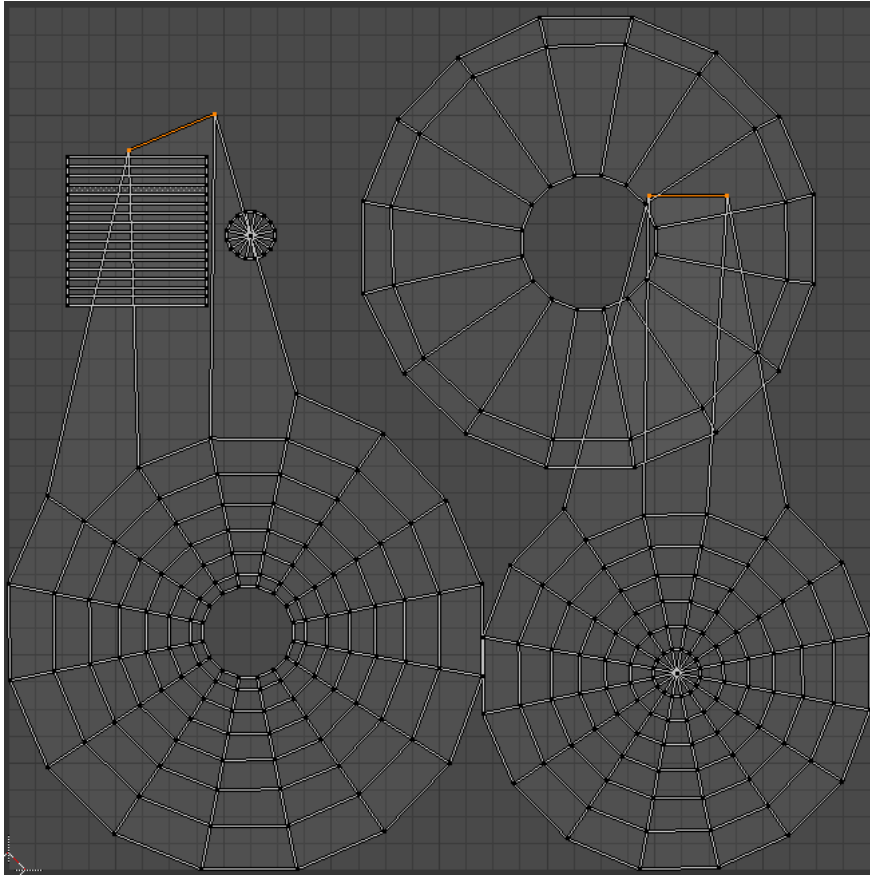



Image 89: Only the left edge was selected and moved but also the corresponding edge on the right moved because the  button was toggled on.

If there are some seams that were added after the first unwrapping, the unwrap can be redone. Though if there were some areas that needed to be adjusted after unwrapping, like a character's hands, the modeler may not want to redo their work. Luckily, there are two ways to changing the unwrap in this situation. For one, all faces of the object need not be unwrapped at the same time. Only pieces of the object can be selected and unwrapped independently. Though this will unwrap the part so it is scaled to fit the grid without regard for the sizes of the other pieces. Thus, the piece needs to be scaled down separately which can throw the scale off when texturing. It often is better to use the Pin tool (P) to mark the vertices that are already where they need to be and then unwrap the whole object. This way the pinned vertices will not change shape or location while all the unpinned vertices will adjust according to the changes made to the seams.

Unwrapping objects in Blender is a quite painless process when compared to programs like 3D Studio Max. This is mostly thanks to the automatic UV layout creation. Still, certain areas are always harder to unwrap than others. With Aroleir, the hands ended up requiring more work than a press of a button and quick resizing of key areas. Some polygons on the sides of the fingers overlapped in the UV map and needed to be adjusted. (Image 90)

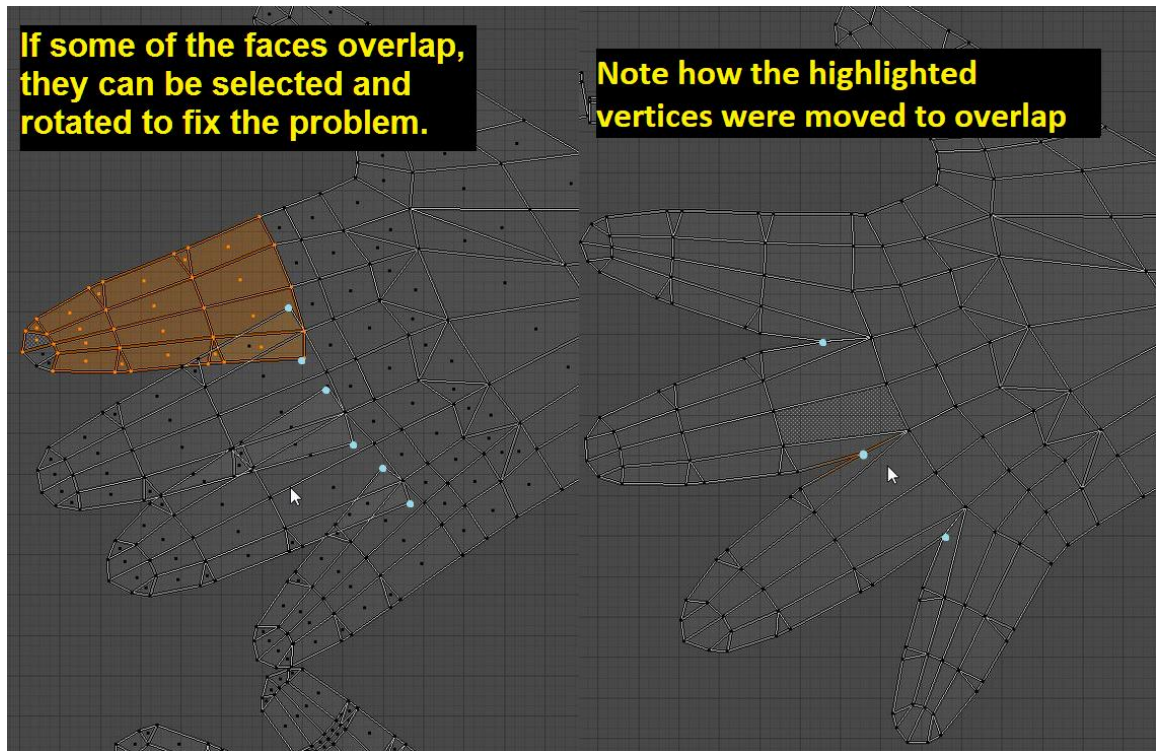


Image 90: Fixing the overlapping faces on the hand UVs.

Aroleir's fur wrap could be used as a more complex example of unwrapping. Here, the seams were used to create a clear line between the belts and the pieces of fur. Meanwhile, the main piece of fur and the fox skin were not separated with a seam as there would be stitches painted over the seam. These stitches would be difficult to align and the transition from one fur type to another would be unnaturally straight if there was be a seam running through them. Image 91 shows the UV layout with simple colours blocked in to show what part is where on the 3D model.

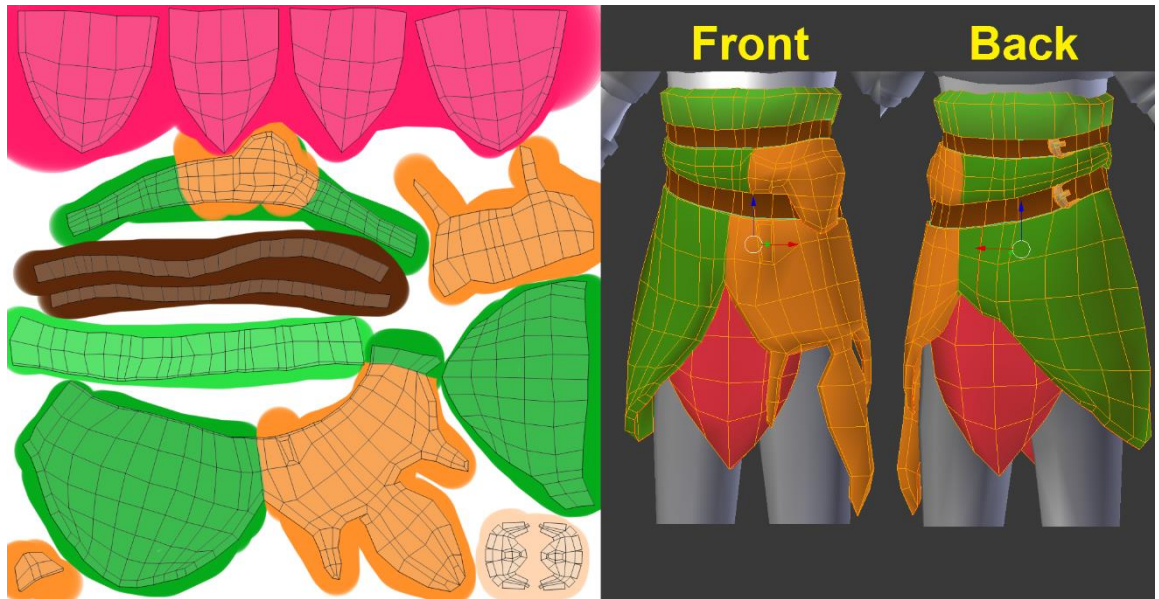


Image 91: An example of using the seams to separate areas with different colours and materials.

With Aroleir, everything symmetrical was unwrapped with the Mirror modifier still active. Thus, only one half of each piece needed to be unwrapped and the modifier would copy the unwrap information as well. Attachment 3 shows how the seams were placed on Aroleir's unclothed body and how the UV layout looked after all the adjustments.

The last step before letting a texture artist begin their work is exporting the UV layouts from Blender. The texture artist can then use these layouts as guidelines when creating the textures. Blender exports the UV layouts as a semi-transparent png file which is easy to paint by. The layout can be exported via the UV/Image Editor by selecting all the faces of the UV layout and choosing "Export UV Layout" from the "UVs" menu on the bottom toolbar of the editor. The texture maps need to be created for each object separately.

5.4 Texturing

The textures of a 3D character can be made in any 2D art program from Adobe Photoshop to Microsoft Paint. Blender itself also has tools for creating painted textures and materials. The tools used depend heavily on what visual style the game

should have. Some games need realistic textures made from actual photographs while others need hand-painted stylized textures. All Aroleir's textures were made with PaintTool Sai while the seams were cleaned up using Blender's own texture painting tools.

Depending on the game, a character model may use several different texture maps per object. Each map has a different purpose ranging from giving the object colour to determining how shiny certain areas are. At this point, Aroleir will only receive two types of maps as time is quite limited. These maps are the Diffuse map that gives the model colour, and the Alpha map that determines which areas are transparent. All parts of the model will have a Diffuse map while only few will have an Alpha map.

When creating textures, it is good to frequently apply the textures to the model to make sure everything is where it should. With that, the first step to texturing is adding flat colours to all the Diffuse maps the character will have. This is done to make sure the textures do not distort, all the details are included and the colours look good on the actual 3D model. The texture artist will open the exported UV layouts in their art program of choice and use them to apply colour where they belong. Image 92 shows one of the many flat colour diffuse maps made for Aroleir and how the flat colour maps together look on him.



Image 92: Aroleir model with flat colour textures. Here we can see some details, like the straps of the armour, were forgotten and needed to be added before beginning to add the shading.

Applying a Diffuse map to a model in Blender is a quick task. Materials are created in the Material tab of the Properties editor, only one tab away from the Modifier tab. After a new material is created, it will be solid white by default. The material can either be set to have a solid colour by clicking on the colour box, or it can be given an image texture by clicking the small square next to the colour box.

If there are no materials on the model, the first material will automatically be applied to the entire object. When using texture maps, an object usually only needs one Diffuse map material but some games can also skip the creation of Diffuse maps and instead use solid colour materials. One object can have several materials which apply to specific polygons. To have a material only affect a part of the object, the artist needs to select the needed faces on the 3D model and press the Assign button while having the right material selected. Image 93 shows how a Diffuse map can be applied to a material.

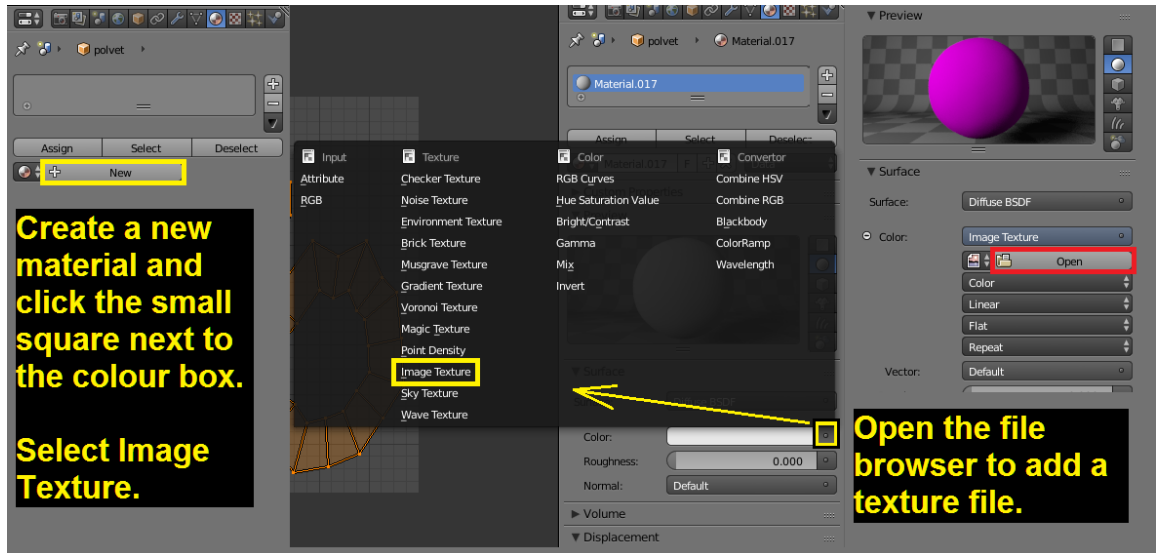


Image 93: Adding a Diffuse Map to the model.

Alpha map is created for objects that have transparent areas. Such objects on Aroleir are his hair, shoes and body as they have areas where hair or fur are drawn on planes that require transparency to achieve the desired look. For example, on the body such areas are the eyelashes. An Alpha map can be created by taking the Diffuse map and colouring all the non-transparent areas black and colouring the transparent areas white. It is good to store the Alpha map in the same psd file as the Diffuse map so the Alpha map can be quickly altered if the transparent areas are changed. Image 94 shows the Diffuse and Alpha maps for Aroleir's body.

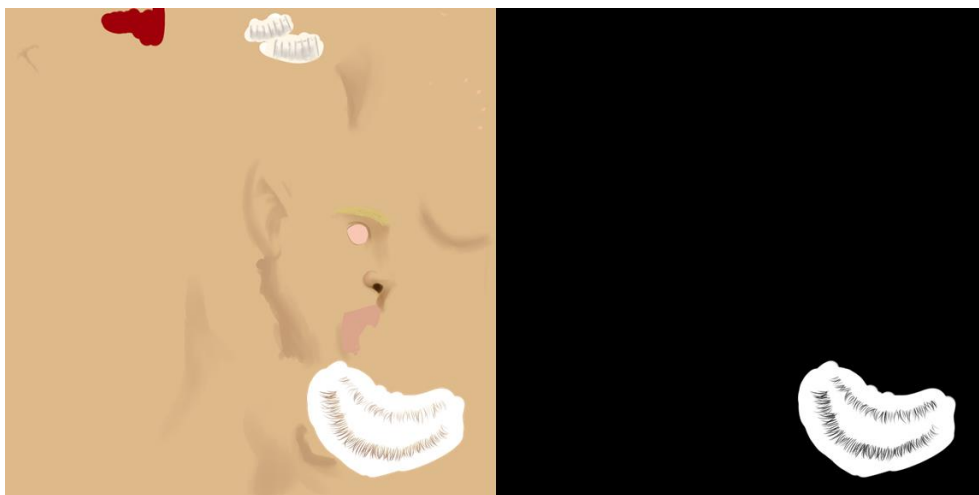


Image 94: Aroleir's body's Diffuse map on the left and Alpha map on the right. On the png of the Diffuse map, the white areas around the eyelashes are transparent while on the Alpha map the white area is coloured solid white.

Adding special maps like the Alpha map to a model in Blender is a slightly different and more complex process than adding a Diffuse map. The artist should open Node Editor to one of their viewports from the Editor selection drop-down menu. An object with just a Diffuse map applied has three nodes; Image Texture, Diffuse BSDF and Material Output. New nodes can be added from the Add menu in the viewport's toolbar.

To add the Alpha map, three new nodes need to be added. The first one is the Image Texture node (Add → Texture → Image Texture). The Alpha map file is then added to this node from the Open button. The second node to add is the Transparent BSDF node (Add → Shader → Transparent BSDF). The last node is the Add Shader node (Add → Shader → Add Shader) which allows the model to display several maps on one material.

Just adding the nodes to the editor is not enough though. The coloured circles on the nodes are called node links and are used to connect nodes to each other. A node link can only be attached to one of its own colour. The yellow node links on the Image Texture and the Transparent BSDF nodes are connected. The Add Shader node is added between the original Diffuse BSDF and Material Output nodes as it is the node that gives room for the new Alpha map. Image 95 shows how the finished node structure should look.

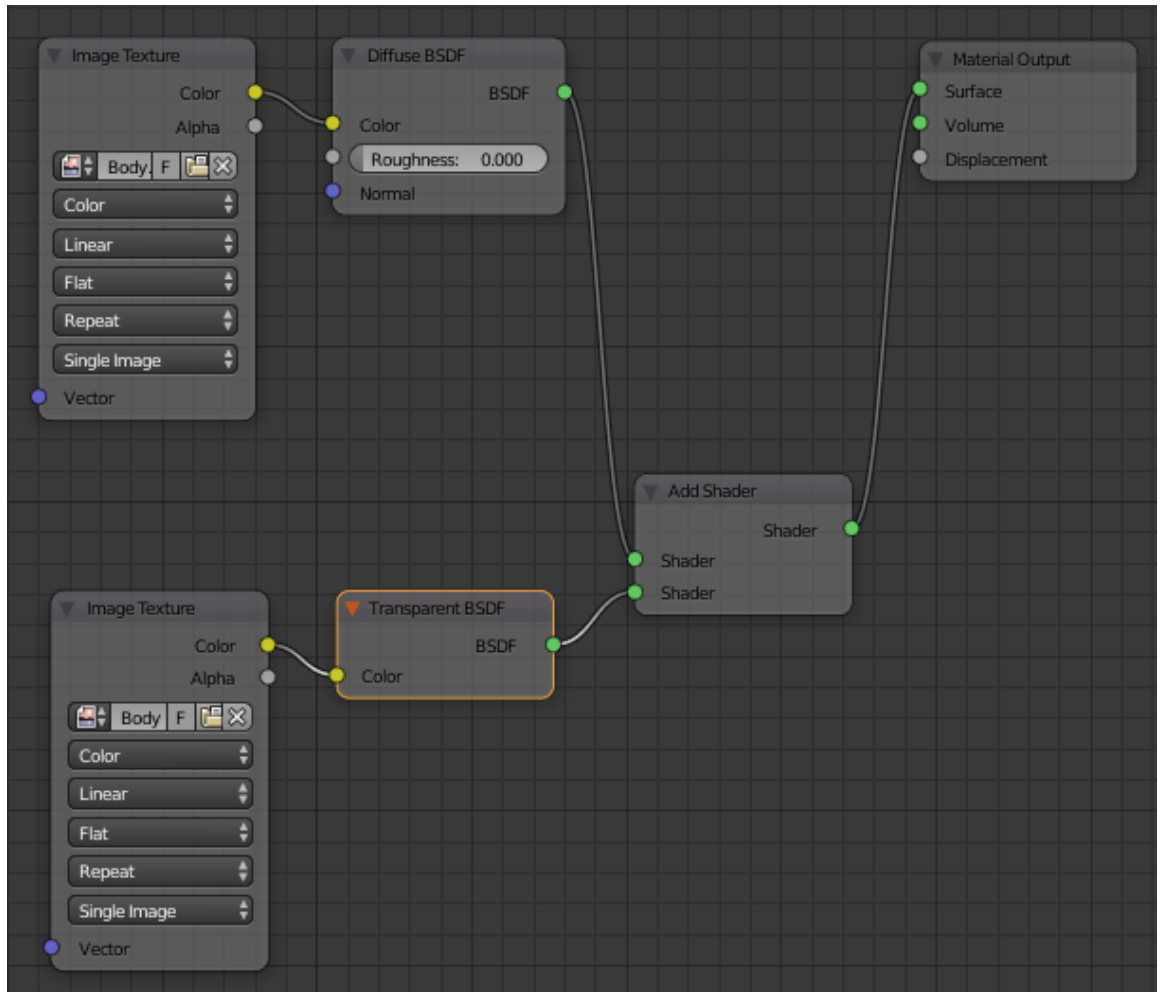



Image 95: Adding an Alpha map to a material using the Node Editor.

Once the materials have been set up, the image files can quickly be updated by pressing the + at the top right of the node editor to open the side menu, selecting the node whose texture you wish to update and pressing the reload button () next to the source file box. This update can be done whenever the image file is edited and saved in the art program.

All that is left to do now, is actually finishing up the textures. Aroleir's textures were left very simple for now due to time limitations. The textures would be finished up after the thesis is finished. The model will also be given more texture maps later. For example, a roughness map will be made to determine which areas of the model are more matte than others. A bump map would also be created to have certain areas, like the breastplate, reflect light as if it was a 3-dimensional shape rather than just painted on.

5.5 Rigging

The method of creating a rig for Aroleir is based on the YouTube tutorial “Blender Tutorial: Basics of Character Rigging” by Sebastian Lague. The rig created in the tutorial is very simple and does not include minute details like the fingers, hair or the face. Still, the video gives a solid base to work on and the rig can work for simple character models just fine. For Aroleir, certain areas were done differently but the process is still mostly the same as in the tutorial video.

The first bone of an armature is added in Object mode by pressing “Shift + A” and selecting “Armature → Single Bone”. The modeler should ensure that the 3D Cursor is at the center of the scene before doing this. If the 3D Cursor is out of place, it can be returned to the center with “Shift + S” and by selecting “Cursor to Center”.

All the following changes to the armature are done in Edit mode. The first bone is moved up to be at about the height of the character’s hips. More bones can be added to the link by using the Extrude tool (E). Bones are added from the sharp end of the first bone upwards until the string reaches the top of the head. The number of bones depends of the complexity of the character but it is good to make at least three bones at this point. The lowest bone controls the lower body, the middle bone controls the upper body and the last bone controls the neck and the head. For Aroleir, a total of six bones were added. Four of these control the torso, one controls the neck and one controls the head. The bones are also all adjusted so they are about in the center of the model.

The next area to create bones for is the arms. While in Edit mode of the armature, the “Shift + A” command will add a separate bone automatically wherever the 3D Cursor is located. A new bone is then moved to the height of the shoulder and the thin end is stretched so the ball is inside the elbow. The placement of the ball of this joint determines where the character’s arm will bend so placing it correctly is crucial. With the Extrude tool (E), new bones are created for the arm and the hand. For a simple hand, a single bone for the hand is enough, but for more detailed

hands like Aroleir's, bones are extruded from the wrist for each finger. Image 96 shows the rig for the torso and the arms. In Aroleir's case, the arms were later connected to the spine by extruding bones from the spine and connecting them to the top of the arms. Doing this is not necessary but it does give added control when animating the character's arms. The legs can be rigged the same way as the arms.

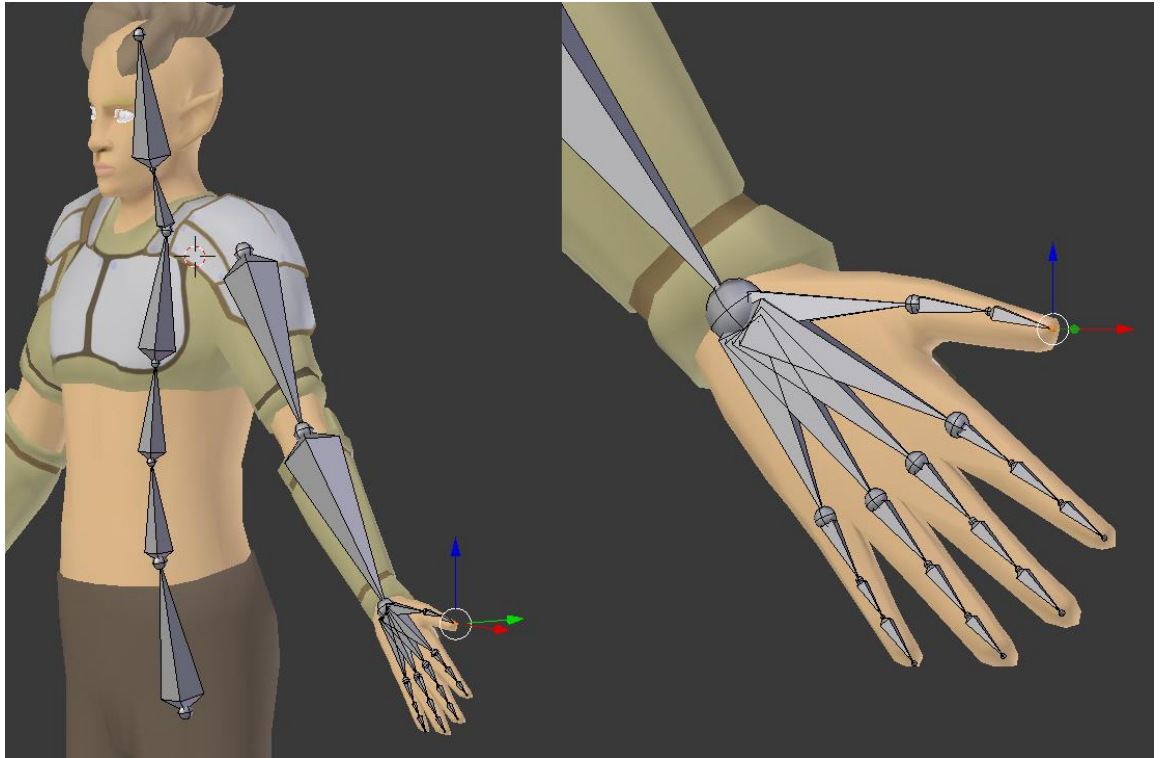



Image 96: Aroleir's rig for the torso, head and arms. The finger bones all begin from the wrist joint.

Once all the limbs have bones, it is time to create IK handle bones for the arms and the legs. These bones are used for controlling the limbs once the Inverse Kinematics (IK) constraints are applied. The IK bones are created by selecting the tips of the bones in the wrist and the ankle and extruding them backwards. The IK bones are then separated from the armature by selecting the said bones, pressing "Alt + P" and selecting "Clear Parent".

The next point is creating the pole targets for the elbows and the knees. These are made by creating two new bones (Shift + A) of which one hovers before the char-

acter’s knee and the other before their elbow. These will be used to set the direction where the character’s elbows and knees will point when animated. the IK bones nor the pole targets deform the character’s body and thus the “Deform” box from the bones’ options () should be ticked off. Image 97 shows the positions where the IK bones and pole targets were added.

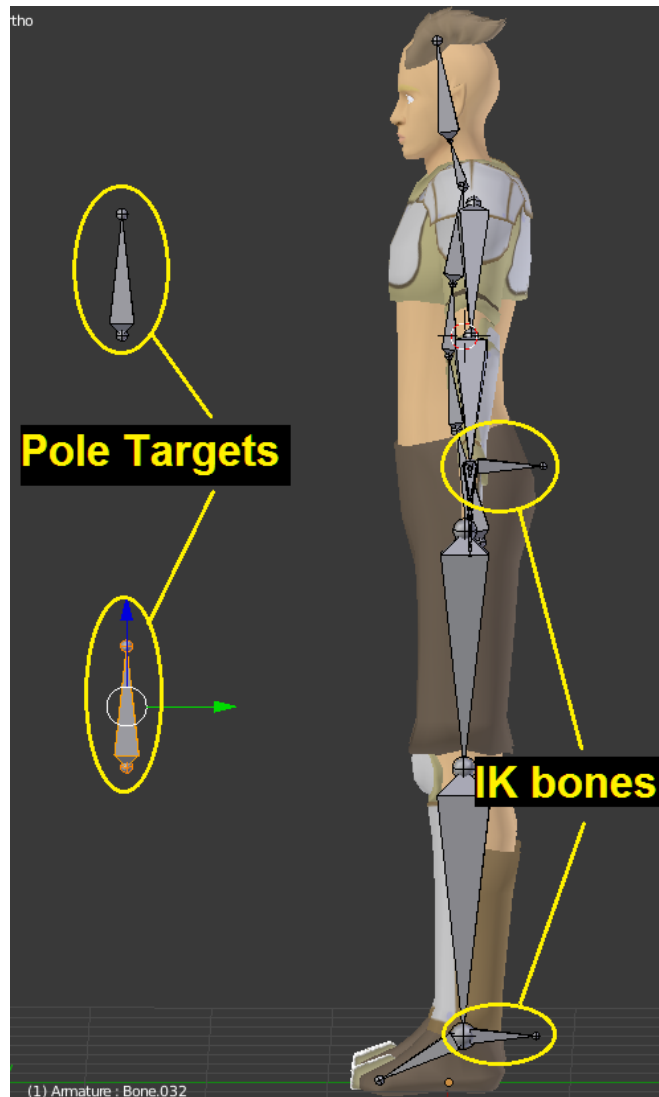



Image 97: The position of the IK bones and the pole targets. An extra bone was later added to Aroleir’s feet to allow him to stand on his toes.

The bones of the armature can be mirrored to the other side of the model but before this can be done, the bones need to be named. Blender can automatically rename mirrored bones that have names that indicate whether they are left or right bones. Therefore, all bones on one side of the model should have “_L” or “_R” at

the end of their name so Blender can recognize them. For example, Aroleir's arm bones were named "UpperArm_L" and "LowerArm_L".

Once the bones have been named, it is time to work further on the IK handle bones. The IK bones are used to move the entire limbs by moving a single bone. Along with this, the IK bones are also used to rotate the character's foot and hand. To do this, the IK bones need to be linked to the foot and the hand. The bones are linked by selecting the foot bone and the leg IK bone, pressing "Ctrl + P" and select "Keep Offset". This is done to the arm IK and the first finger bone of each finger counting from the wrist as well. To check if the bone works as it should, the modeler should go from Edit mode to Pose mode and then rotate the IK bone. Rotating the IK bone should move the entire hand or foot's bones with it.

The next step is applying constraints to the IK bones. To do this, the arm IK bone is selected first and the last bone of the chain of arm bones – usually the lower arm or leg bone – is selected second. With the "Ctrl + Shift + C" combination in Pose mode, the "Add Constraint (With Targets)" menu can be opened. From here, the Inverse Kinetics option is chosen.

After this, the chain length needs to be set from the Bone Constraints () menu in the Properties panel. The chain length is the amount of bones connected to the IK bone in the constraint. For human limbs, this number is usually 2 for both the arms and the legs. This makes it so moving the IK bone will also move the connected bones. The Pole Targets are also set in the Bone Constraints menu. The Armature is selected from the Pole Target drop-down menu. This will create another drop-down menu titled Bone. In the Bone menu, the knee or the elbow Pole Target done is selected to assign it. This way the knee or the elbow will always face the Pole Target. If the elbow or the knee turns the wrong way when the IK bone is moved, the Pole Angle slider can be adjusted until the knee bends in the right direction.

The next step is to finally mirror the bones to the other side. First, the 3D Cursor needs to be placed in the center of the scene (Shift + S). To rotate items around the 3D Cursor, the "." key needs to be pressed. Then, while in Edit mode, the bones that need to be mirrored are selected, duplicated and rotated to the other

side (S + X + -1). As a finishing touch, the names of the bones can be changed by going to the Armature drop-down menu in the bottom of the viewport and selecting “Flip Names”.


Applying the armature to the character model is done in Object mode. The Armature and all the objects of the character model are first selected. The “Set Parent To” menu is opened with “Ctrl + P” and “With Automatic Weights” is selected. This way Blender will guess which bones will control which parts of the mesh. The automatic weights are rarely perfect but offer a competent base to work upon. The modeler should test all the bones to see how they work and if everything moves even remotely as it should. At this point it is good to pull the model to its absolute limits and make note of any problems that arise.

On Aroleir, there were two especially notable problems with the automatic weight paints. For one, the inside of the fur wrap around his body somehow was connected to the bones of his hand and the polygons would follow the arm’s movements. Similarly, whenever the head was tilted, Aroleir’s teeth would move at a different speed and clip through the side of his lips. This again was caused by the fact the teeth were weight painted partially to follow his chest bone which weakened the neck and head bones’ influence on the teeth. Image 98 shows the teeth clipping through Aroleir’s lips.



Image 98: Mistakes in weight painting can lead to unpleasant results.

The automatic weights can be edited by first selecting a bone in Pose mode, then selecting a piece of the mesh and finally switching the mode to Weight Paint. In this mode, the mesh is coloured to show how much that specific bone affects that mesh. Blue areas are not affected by the bone at all while red areas follow the bone completely. The tools can be changed from the top-left corner of the viewport. The Add tool can be used to strengthen a bone's influence on the mesh while the Subtract tool can be used to weaken it. For Aroleir, these tools were used to make the face area only follow the head bone instead of having the lower half also be partially attached to the neck bone. Having the lower half respond to a second bone caused his mouth to open and jaw to deform when his head was tilted back.

Meanwhile, for more precise work, a bone's influence on specific vertices can quickly be checked and altered from the Data tab () of the Properties panel. While in Edit mode, the vertices of the model can be selected and in the Data tab the bones can be chosen from the Vertex Groups list. The bones' influence on the

object can then be removed by pressing the Remove button. Image 99 shows the Data tab.

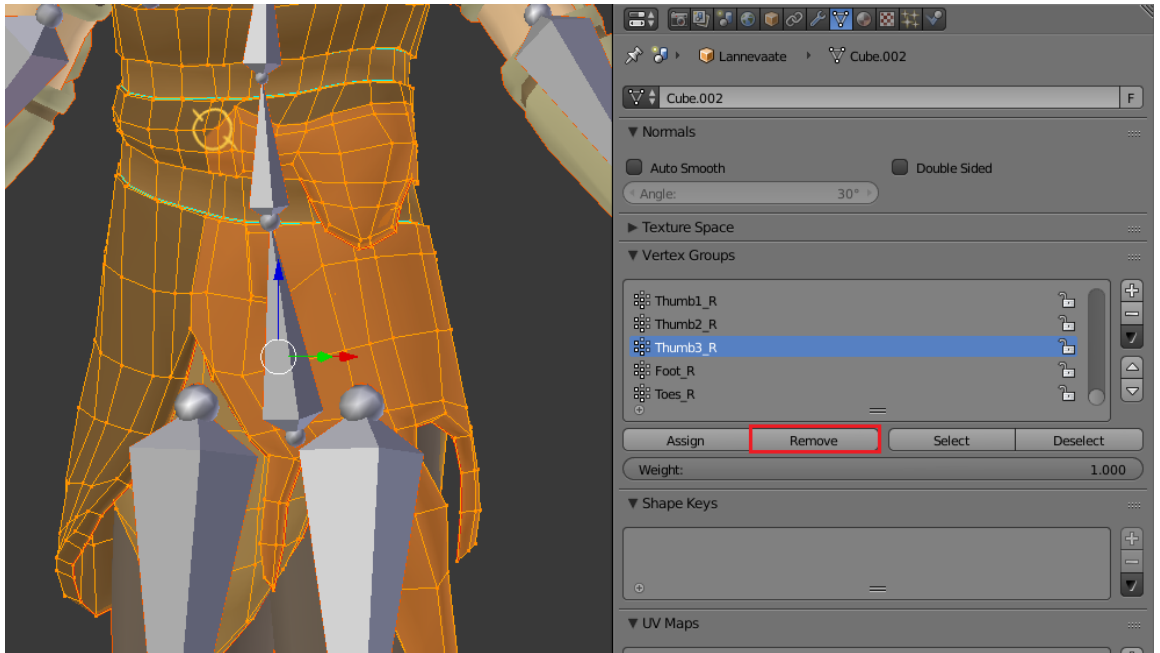


Image 99: Removing a bone's influence on a mesh. On Aroleir, the hand bones somehow were connected to the fur wrap around his waist.

At this point, when the lowest torso bone of the character is moved, it will not affect the character's arms or legs, causing those body parts to distort heavily. To have the legs and arms respond to the torso's movements naturally, the first bone of each chain needs to be parented to the lowest torso bone. In Edit mode, the bones of the upper arms and thighs are selected first, the hip bone is selected last and the bones are parented with the Keep Offset action (Ctrl + P + "Keep Offset"). On Aroleir, the arm bones were not parented like this as his rig already had separate bones for the shoulders which were connected to the spine. Image 100 highlights the bones that are to be connected.

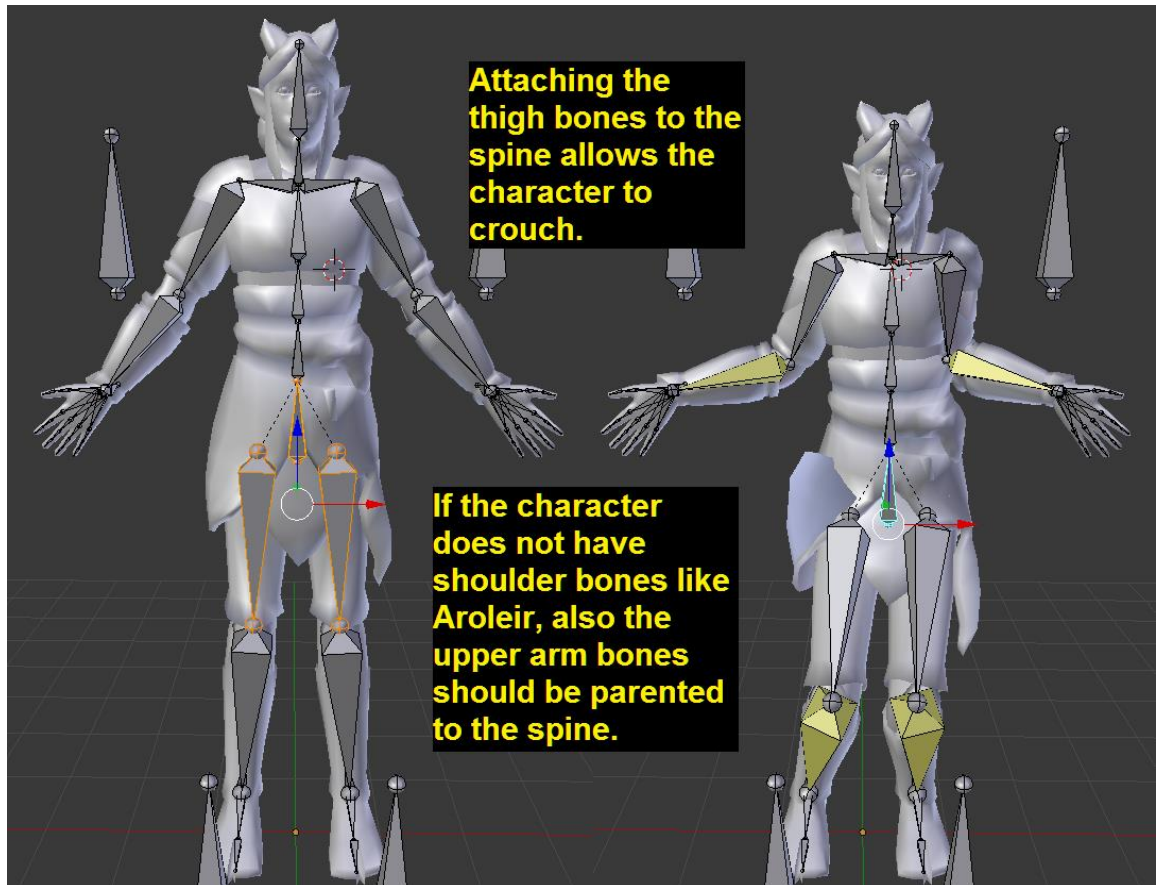


Image 100: Giving the character the ability to crouch easily. At this point the weight painting of the rig was still unfinished so the inside of the fur wrap clipped through the rest. This was fixed when the entire armature was finished.

Long hair does not always need to be animated, but in Aroleir's case, the hair was rigged as a personal challenge. To be specific, the strands on the front, the ponytail and the loop of hair that was tied to the ponytail all were given a simple rig. The hair loop and the ponytail were rigged as a separate armature by creating a chain of bones that begun from the root of the loop on the front and ended at the tip of the ponytail. IK bones were then extruded backwards from the beginning and the tip of the ponytail. The bones where the IK handle bones were extruded from were then given the Inverse Kinematics bone constraint the same way as the arms and legs before. Image 101 shows how the rig looked and worked at this point.

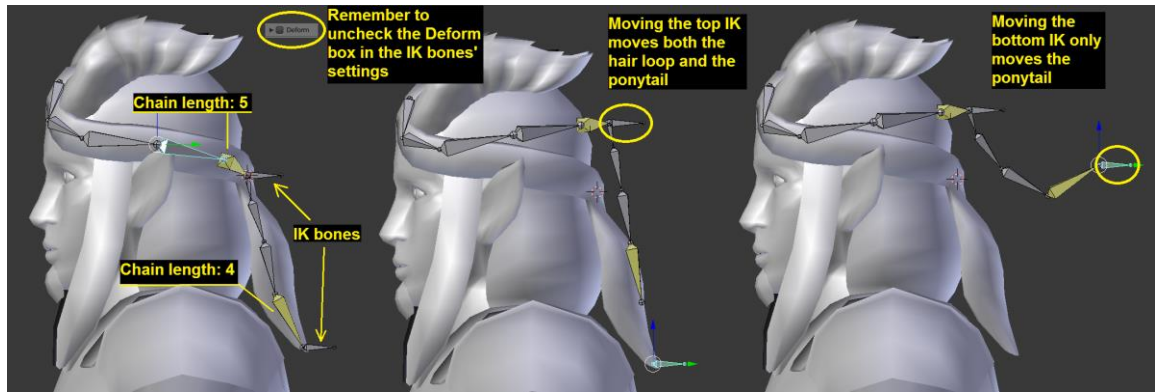


Image 101: The armature for the hair loop and the ponytail.

The free strands on the front were rigged the same way the ponytail. When the chain of bones was created, the amount of bones was matched to the amount of horizontal edge loops in the hair, not counting the top edge of the strands. That was the strands are easier to weight paint as each bone controls one edge loop on the hair strands. As before, the IK bone was extruded from the last bone of the chain. This last bone was also given the Inverse Kinematics constraint.

Once one side of the hair rig was done, the other side could be mirrored just like with the body rig. The bones that needed to be mirrored were selected and duplicated. The duplicated bones were then scaled by -1 along the X-axis. This will also retain the bones' settings so the mirrored side of the hair loop will respond to the same IK handle bone as the original side without any edits. Image 102 shows the finished hair rig.

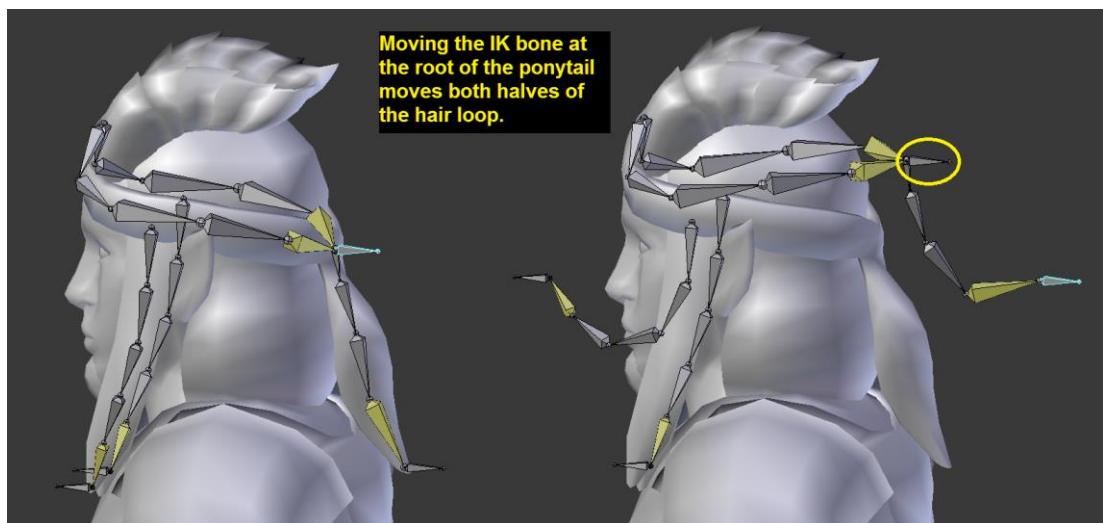


Image 102: The finished hair armature.

The next step was to apply the rig to the hair mesh. While in object mode, the hair object was first selected and the rig after it. With the “Ctrl+ P” command and “With Automatic Weights” option, the hair rig was connected to the hair mesh. This way the hair rig will only affect the hair. Unlike the body rig, though, the hair mesh needed more work than this. Blender’s Automatic Weights tool does not know that the rig should only control specific parts of the mesh. By default, the ponytail bones deformed the base of the hair and the bones of the front strands deformed the beard. To fix this, all the polygons that the rig should not affect were selected in Edit mode and then unlinked from the hair rig’s bones under the Data tab (🔗). Every bone’s vertex group needed to be unlinked one by one but this caused an immediate improvement in the way the rig functioned.

The weight paints on the ponytail and hair loop worked as they should by default, but the front strands had a peculiar problem. The outer strand of hair was completely connected to the last bone of the last hair strand rig. This caused the outer hair strand to spin wildly while the inner strand worked relatively well. In the end, this was fixed by manually weight painting the hair strands to the correct bones. As each horizontal edge loop of the hair strands – not counting the top edge which would remain static in animation – had their own bone, selecting and assigning each of them to the correct bone’s vertex group did not take long. Image 103 illustrates these last steps.

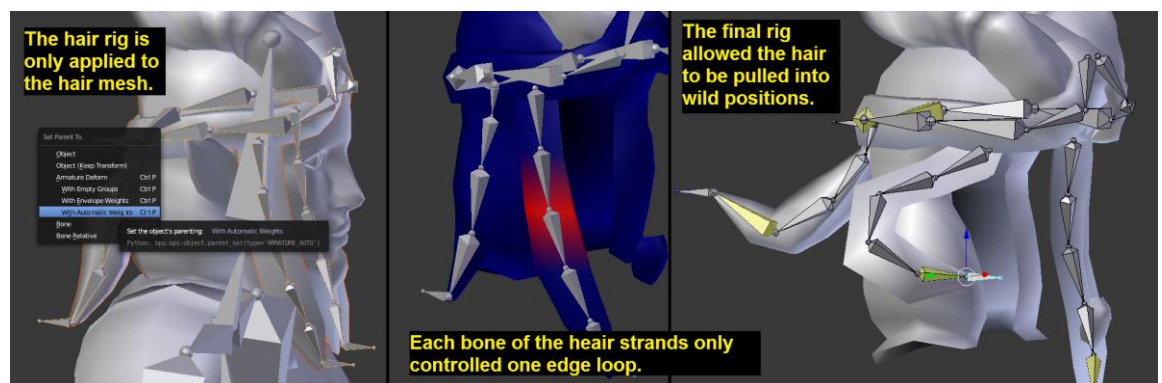


Image 103: Attaching the hair rig to the hair mesh.

While the hair was rigged with a separate armature, it needed to be combined with the armature of the body. The hair armature and body armature were selected in Object mode and joined together with the “Ctrl + J” command. It is good to make

sure the rigs both work after this joining as the weight paints may disappear if there were excess objects selected at the time of the joining. Once the armatures are a single object, the first bones of each bone chain in the hair rig need to be parented with the head bone so the hair rig will move with the head. In Aroleir's case, the weight paints of the hair also needed to be adjusted slightly to ensure the hair would not be left behind when the head was moved. Image 104 shows the specific bones that were parented to the head in Aroleir's case.

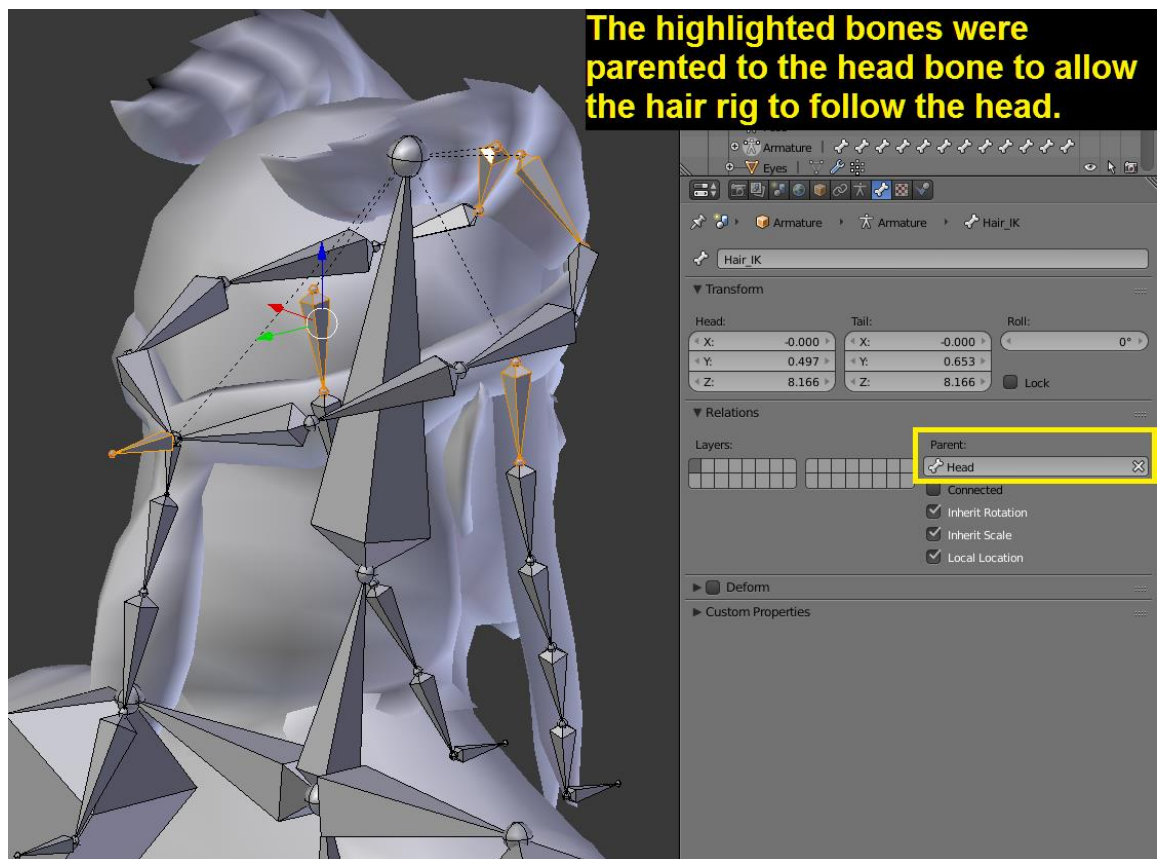


Image 104: Parenting certain bones of the hair to the head bone allowed the hair rig to follow the head's movements. The tips of the hair strands and the ponytail were not parented to the head as they should hang free and move independently.

When Aroleir's arms were bent, the automatic weight painting did not do its job perfectly but it was deemed good enough for now. There was though a rather big problem with Aroleir's elbow plates. The plates did not stay where the elbow naturally should be and instead moved to the side of the arm, thus clipping through the bracers and the shirt sleeve. To stop this, new bones were extruded from Aroleir's elbows and the elbow plates were weight painted to respond to no other

bone than these new elbow bones. The new bones allow the elbow plates to be rotated around the elbow and thus placed wherever the back of the elbow would be in any pose. It is possible that this problem would not have occurred in the pole target bone had been placed behind the elbow instead of in front of it, but there was no time to test a new arm system at this point. Image 105 illustrates the elbow plate's placement before and after the extra bone's influence.

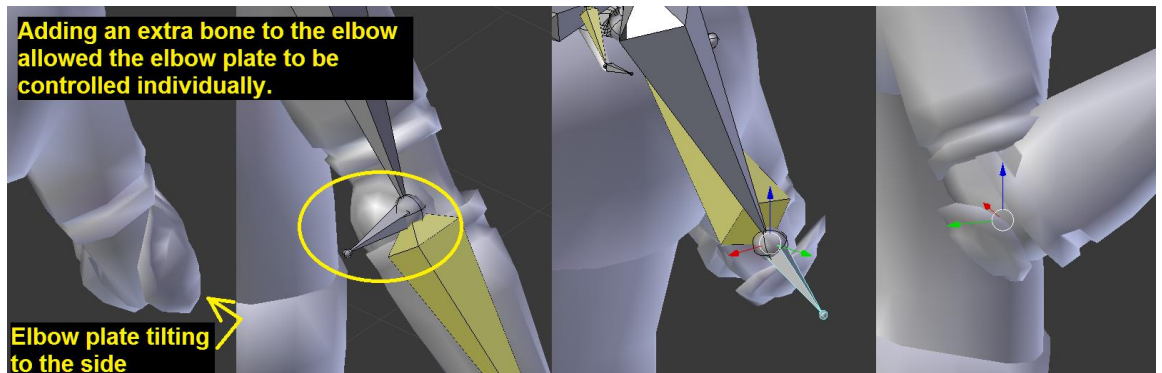


Image 105: Adding extra bones to control specific parts of armour can be helpful.

It is useful for a rig to have a root bone that can be used to move the entire character model without affecting its pose. This bone can be arranged by creating a bone at the character's feet, rotating it 90 degrees so it points forward and then parenting to it all the bones that do not yet have a parent. An Aroleir, such bones were the hip bone, most of the the IK bones as well as the pole targets. If done right, moving the new root bone in Edit mode will cause the entire character to move without any bones being left behind.

As a finishing touch, the shape of certain bones can be customized. This can be done by selecting a different layer in Object mode and creating new basic shapes. A sphere and ring. The ring can be made by creating a circle object, extruding its edges in Edit mode and scaling the new edges inwards. Once the shapes have been created, the layer or layers with the character model are selected. The new shapes can be added to a bone in Pose mode under the Display tab of a bone's properties (🦴). The custom shapes can then be selected from a drop-down menu. For Aroleir, the pole targets were set to be spheres while the root bone created in the last paragraph was set to be a ring. If the shapes are too large, they can be adjusted with the Scale slider under the custom shape box. The sizes should not

be changed by scaling the bones from the armature itself as this could affect the rig's functions. Image 106 shows these steps along with the finished rig.

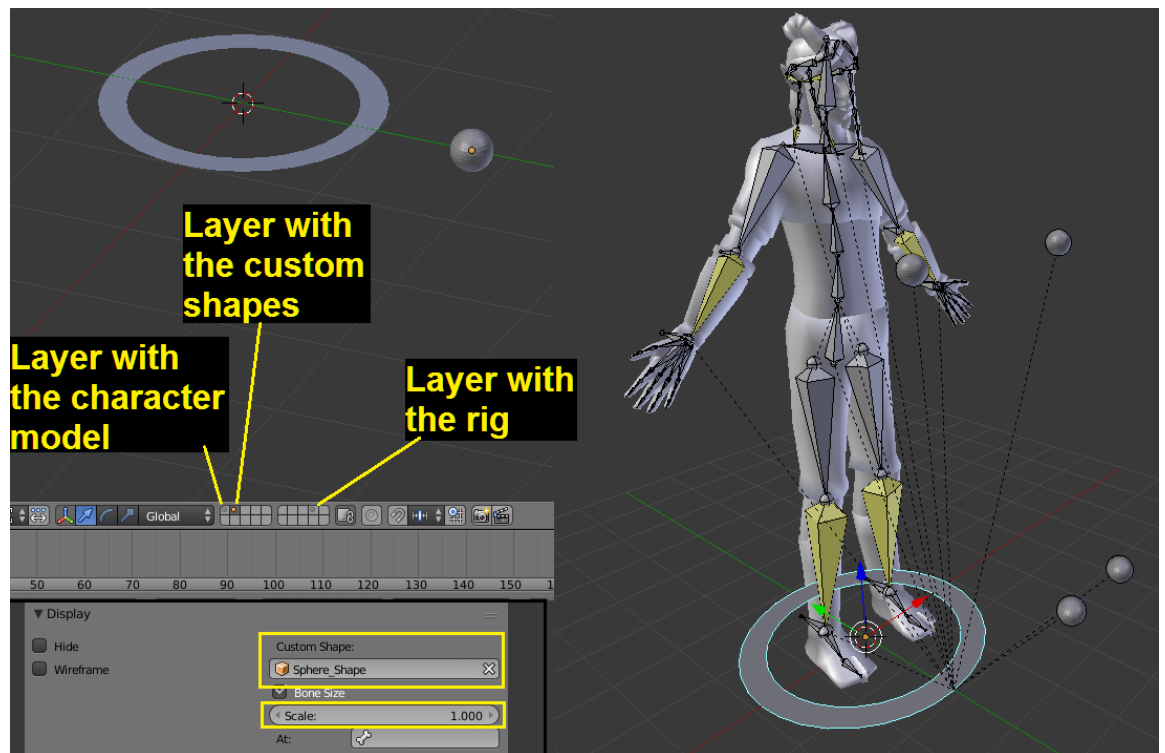


Image 106: Customizing bone shapes.

After all this is done, the main rig should be finished and the model ready for animation. Of course, it is good to still test the model by pulling it into extreme poses and see if it performs well even under stress. When it comes to a model like Aroleir, there is still also the task of hiding the polygons that are hidden by other pieces of equipment. For example, most of his torso should be hidden as it would be covered by his shirt, pauldrons and fur wrap. Some characters also could benefit from having a rig for their face but in Aroleir's case that is left to be a future addition as creating a face rig would need more time than was available.

6 CONCLUSIONS

From 2D character design and texture creation to mastering a 3D program's features and understanding the needs of a game engine, creating a 3D character for a video game is a process that requires several different skills. While all this can be done by one person, it is easier on everyone if the workload is shared between several individuals. That way each part of the pipeline can be handled by someone who has the time and skills for that specific task instead of someone needing to act as a jack of all trades.

Creating a 3D model sheet is a task that requires the artist's full attention if they wish to do it properly. Creating a model sheet where every little detail aligns perfectly on all the views is tricky. Still, the artist should not worry too much over having everything be perfect as the 3D modeler can fix any discrepancies while creating the model. The artist should though state which view is the most important to follow in case of some areas not matching up perfectly. For example, on Aroleir the front view took priority in most cases, but for the lips, nose and the horns the side view was more important.

The task of creating the character model for Aroleir was not too difficult a task in itself. The true difficulty stemmed from the fact every step needed to be recorded and explained in detail so someone else could use the documentation to create their own work. Working with a program like Blender relies heavily on the user's ability to remember quick keys and navigate the user interface of the program so creating a tutorial requires every action to be explained. The task was not made any easier by how customizable Blender's interface is as one could never know if the reader has the windows laid out the same way the writer did. Still, the final guide should work as a step-by-step guide for someone who wishes to create their own 3D character, regardless of whether that character is simple or complex.

Due to the time being limited, certain areas of the creation of Aroleir's model needed to be left in a rough state to leave time for writing the guide itself. For this reason, Aroleir's textures are very simple flat colours without feel for material. Creating proper textures would require quite a lot of time while providing very little

material to write about for the guide. The textures for Aroleir will be finished later as a personal project. Similarly, creating a face rig was left for a later time as the process would likely require a lot of time to create, polish and finally explain in a way that would be informative.

The images and explanations used in the thesis are also used to create Moodle-course material on 3D character modeling for the Kajaani University of Applied Sciences. The course material is created in a PowerPoint slide format which allows the focus to be on the images instead of the text. This is more effective in terms of allowing the reader to create their own character while reading the guide even if in that method not always explain why exactly certain areas are created the way they are.



REFERENCES

Internet Pages

Autodesk. (2016, 15. November). UVW Map Modifier. Read: 23.3.2017, on the site Autodesk Knowledge Network. Link address: <https://knowledge.autodesk.com/support/3ds-max/learn-explore/caas/CloudHelp/cloudhelp/2017/ENU/3DSMax/files/GUID-78327298-4741-470C-848D-4C3618B18FCA-htm.html>

Blender 2.78 Manual. Docs >> Rigging >> Armatures >> Bones >> Introduction. Read: 11.5.2016, link address: <https://docs.blender.org/manual/de/dev/rigging/armatures/bones/index.html>

Engländer, F. (2015, 26. May). The T-pose – all about this mighty blueprint. Read 9.2.2017, on the site Animator Island. Link address: <http://www.animatorisland.com/the-t-pose-all-about-the-mighty-blueprint/>

Galand, R. (2016). Overwatch – Junkers. Read 8.2.2017, link address: <https://www.artstation.com/artwork/EzBI4>

Lague, S. (20.2.2013). Blender Tutorial: Basics of Character Rigging. Link Address: <https://www.youtube.com/watch?v=cGvalWG8HBU>

Pluralsight 1. (2015, 17. January). Create 3D Models More Quickly with a Model Sheet. Read: 11.11. 2016, on the site Pluralsight. Link address: <http://blog.digital-tutors.com/create-3d-models-3x-faster-with-a-model-sheet/>

Pluralsight 2. (2015, 12. February). How to Create Your First Character Rig in Blender: Part 2 – Weight Painting. Read: 10.5.2017, on the site Pluralsight. Link address: <https://www.pluralsight.com/blog/tutorials/weight-painting-in-blender>

Polycount. (2016, 3. March). Normal Map. Read 23.2.2017, on the site Polycount Wiki. Link address: http://wiki.polycount.com/wiki/Normal_map

Polycount. (2015, 14. toukokuuta). Limb Topology. Read 11.11. 2016, on the site Polycount Wiki. Link address: http://wiki.polycount.com/wiki/Limb_Topology

Polycount. (2011, May). T-pose – What are people using and why?. Read 9.2.2017 on the site Polycount. Link address: <http://polycount.com/discussion/84508/t-pose-what-people-are-using-and-why>

Polycount Del. (2011, January). Face Topology [Breakdown Guide]. Read: 23.3.2017, on the site Polycount. Link address: <http://polycount.com/discussion/80005/face-topology-breakdown-guide>

Rinaldi, D. Modeling Quality Checklist. Read: 23.3.2017. Link address: <https://danterinaldidesign.com/3d-modeling-quality-checklist/>

Roger, M. (2009, 17. December). Joan of Arc: Modeling the Head. Link address: <https://www.3dtotal.com/tutorial/1344-joan-arc-modeling-the-head-3ds-max-by-michel-roger-character-arc?page=1>

Silverman, D. (2013, 5. March). 3D Primer for Game Developers: An Overview of 3D Modeling in Games. Read: 22.3.2017, on the site Envato Tuts+. Link address: <https://gamedevelopment.tutsplus.com/articles/3d-primer-for-game-developers-an-overview-of-3d-modeling-in-games--gamedev-5704>

Slick, J. 1 (2016, 20. October). 5 Common Pitfalls of Beginning Modelers. Read: 23.3.2017, on the site Lifewire. Link address: <https://www.lifewire.com/common-pitfalls-of-beginning-modelers-2052>

Slick, J. 2 (2016, 7. October). Surfacing 101 – The Basics of Texture Mapping. Read: 23.3.2017, on the site Lifewire. Link address: <https://www.lifewire.com/texture-mapping-1956>

Slick, J. 3 (2016, 2. November). What is Rigging? Read 9.5.2017, on the site Lifewire. Link address: <https://www.lifewire.com/what-is-rigging-2095>

Unity User Manual (5.5). (2017, 8. March). Modeling characters for optimal performance. Read: 22.3.2017, from the site Unity Documentation. Link address: <https://docs.unity3d.com/Manual/ModelingOptimizedCharacters.html>

Van Gumster, J. (2016, 19. February). Version control isn't just for programmers. Read: 24.3.2016, on the site opensource.com. Link address: <https://opensource.com/life/16/2/version-control-isnt-just-programmers>

Ward, A. (2011, 20. September). How to create character models for games: 18 top tips. Read: 22.3.2017, on the site Envato Tuts+. Link address: <http://www.creativebloq.com/how-create-character-models-games-18-top-tips-9113050>

Zagobelna, M. (2014, 29. December). Design and Draw a Model Sheet of a Werewolf Warrior. Read 11.11. 2016, on the site Envato Tuts+. Link address: <https://design.tutsplus.com/tutorials/design-and-draw-a-model-sheet-of-a-werewolf-warrior--cms-22834>

Image Sources

Image 2. The difference between the perspective and orthographic projection. Source address: <https://www.script-tutorials.com/webgl-with-three-js-lesson-9/>

Image 3. Witcher 3 environment with simple textures. Source address: <https://www.gamespot.com/articles/witcher-3-mod-downgrades-graphics-to-look-like-3ds/1100-6444492/>

Image 5. Examples of 3D hands. Source address: <http://www.creativebloq.com/how-create-character-models-games-18-top-tips-9113050>

Image 6: Different ways of triangulating a pentagon. Source address: <https://gamedevelopment.tutsplus.com/articles/3d-primer-for-game-developers-an-overview-of-3d-modeling-in-games--gamedev-5704>

Image 7. Using poles in modeling a face. Source address: <http://polycount.com/discussion/80005/face-topology-breakdown-guide>

Image 8. The joint type featured on the Polycount Wiki page regarding limb topology. Source address: http://wiki.polycount.com/wiki/Limb_Topology

Image 16. Human body proportions. Source address: <http://www.creativecomi-cart.com/measuring-human-proportion.html> AND <http://www.idrawdigital.com/2009/01/drawing-tutorial-anatomy-and-proportion-1/>

By Andrew Loomis.

Image 82. 3D hair example from Final Fantasy XIII. Original image by Square Enix. Fetched from Polycount. Source address: <http://wiki.polycount.com/wiki/HairTechnique>

MODELING COMMANDS

1.	Shift + S → "Cursor to Center"	Move 3D Cursor to the center of the scene
2.	Shift + Ctrl + Alt + C → "Origin to 3D Cursor"	Move an object's Origin to the 3D Cursor
3.	Shift + Mouse Click	Select multiple parts
4.	Alt + Mouse Click	Select a loop
5.	A	Select all visible parts
6.	L	Select island (All connected vertices/edges/faces)
7.	B	Box Selection tool
8.	C	Circle Selection tool
9.	G	Grab and Move the selection
10.	G + G	Move the selection along the object's surface
11.	G + X/Y/Z	Move selection along an axis
12.	G + Alt + X/Y/Z	Move selection along two axes (Disable one axis)
13.	H / Alt + H	Hide/Unhide selection
14.	S	Scale the selection
15.	S + X/Y/Z	Scale along an axis (Scaling to 0 can be used to align the selection to an axis)
16.	R	Rotate the selection
17.	R + X/Y/Z	Rotate around an axis

	X	Delete or Dissolve the selection (Delete removes faces, Dissolve merges them)
	Numpad 1 / Numpad 3	Front view / Side view
	Numpad 9	Rotate view 180 degrees
	Numpad 5	Toggle Perspective / Orthographic view
	,	Pivot Point to 3D Cursor
	.	Pivot Point to Bounding Box Center (Default)
	Ctrl + J → “Join Selected Meshes”	Combine objects into one (Object mode only)
	E	Extrude edges/faces
	F	Create a face between selected edges
	J	Create an edge between vertices (Requires a face to function)
	S	Subdivide a face or an edge
	Alt + M	Merge Vertices
	Y	Separate faces / Split edges
	Ctrl + B	Bevel selected edges
	P → “Selection”	Separate the selection to a new object

UNWRAPPING COMMANDS (UV / Image Editor only)

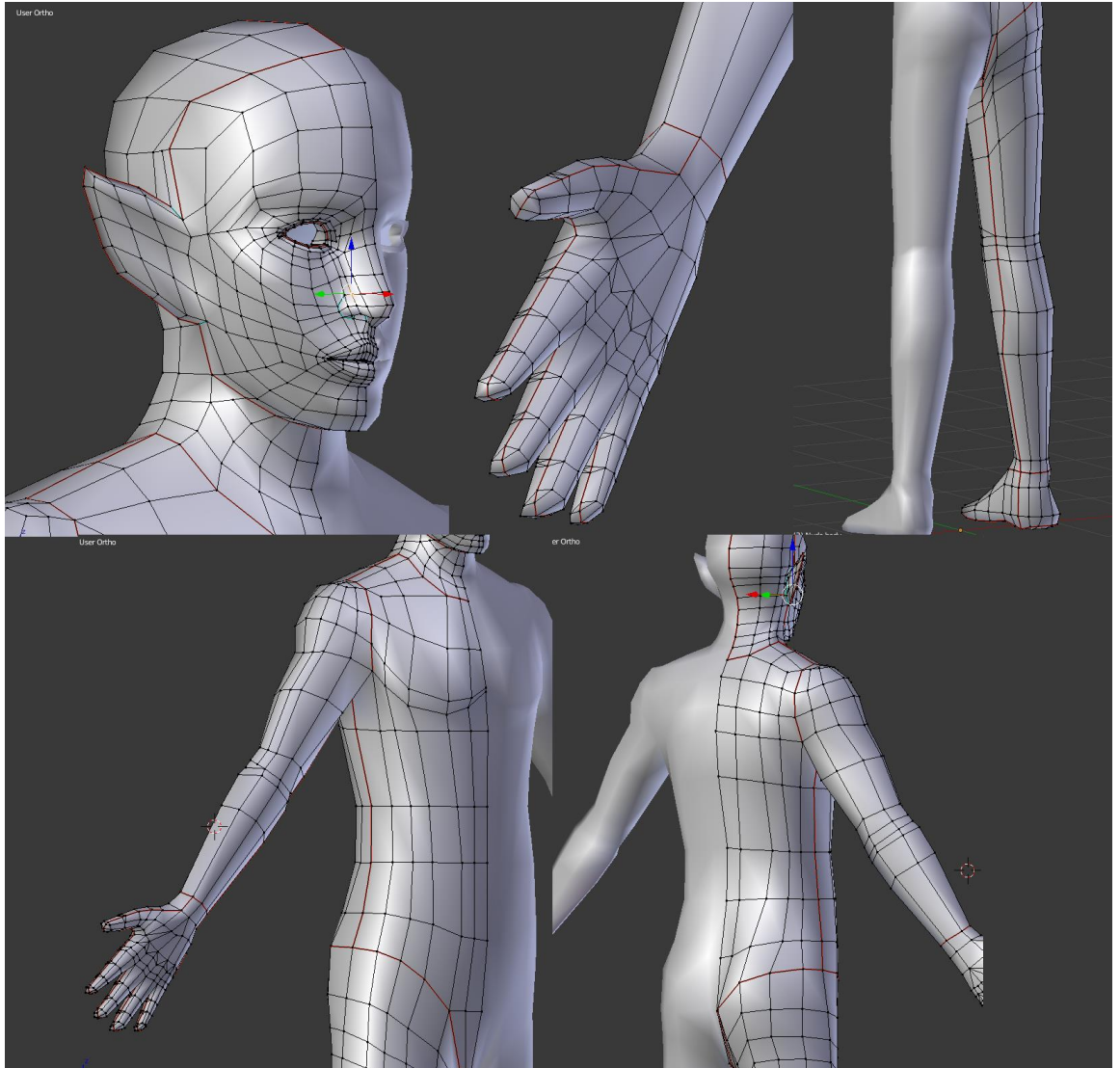
	P / Alt + P	Pin / Unpin
	W → Weld	Merge selected vertices

RIGGING COMMANDS

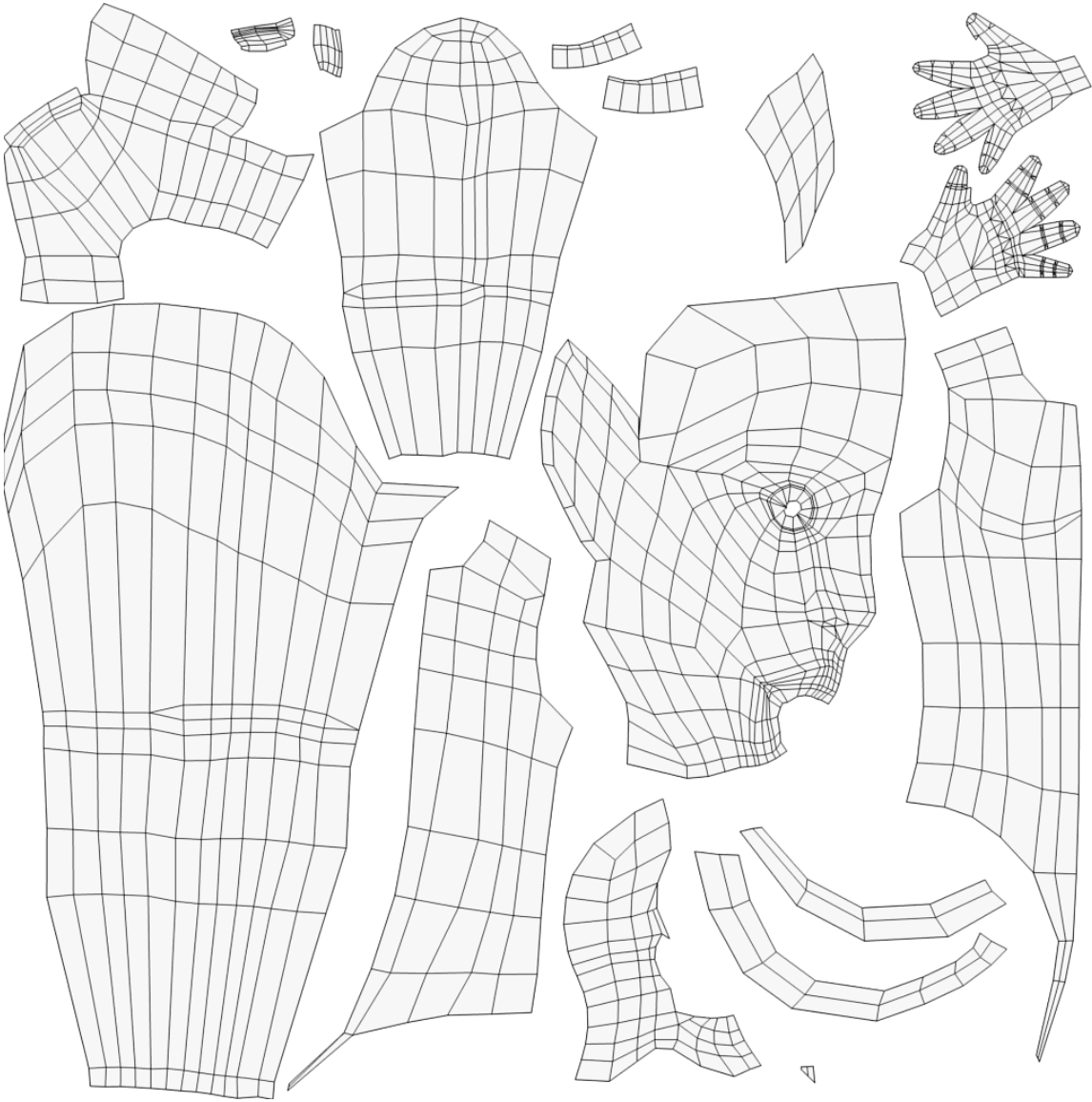
	Shift + A → Armature → Single Bone	Create a new bone
	Alt + P → “Clear Parent”	Separate a bone from its parent
	Ctrl + P → “Keep Offset”	Parent the selected bone/s to the last-selected bone
	Ctrl + Shift + C → “Inverse Kinematics”	Add IK constraint to a bone
	Ctrl + P → “With Automatic Weights”	Attach selected meshes to an armature (Object mode only)



Examples of UV layouts on Aroleir.



Body seams.



Body UV layout.