Beyond Rig'ing for the Animator and Independent Director

David Woloz

Submitted in Partial Fulfillment of the Requirements
for the Degree of Master of Fine Arts in Animation
at
The Savannah College of Art and Design

____/_____/_______
Author – David Woloz Date

____/_____/_______
Committee Chair – Brett Rutland Date

____/_____/_______
Committee Member – Brian Schindler Date

____/_____/_______
Committee Member – Jose Luis Silva Date

Beyond Rig'ing for the Animator and Independent Director

The creation of an AutoRig focused on the non-professional users with unlimited options

March 2018

A Thesis Submitted to the Faculty of the Animation

Department in Partial Fulfillment of the Requirements

for the Degree of Master of Fine Arts

at

Savannah College of Art and Design

By

David Woloz

Savannah, GA

Acknowledgements

I would like to specifically thank my parents and boyfriend who stood beside me throughout the entire process, keeping me sane when I could not figure out what was wrong, which happened on far too many long nights.

I would like to thank all of the amazing professors at Savannah College of Art and Design who expanded my education and pushed me to question what can be produced in Animation.

And lastly all my friends and family who have supported and, in most cases, distracted me enough to not lose my mind throughout this process.

Table of Contents

//Abstract – 5

//Personal Background – 6

//Presentation Style – 8

//Construction – 9

//Process – 10

//Premade Physiologies – 14

//Part Specifications – 15

//Running Scripts – 16

//Packaging & Documentation – 17

//Under the Hood – 18

//Planning Conclusion – 19

//Beta Test – 21

//Graduate Experience – 22

//Glossary – 24

//Existing AutoRigs – 26

//Film Stills – 27

//Project Stills – 28

//Character Designs – 32

//Bibliography – 33

Abstract

Through the application of regularly updated components within Autodesk Maya program the ability to rig can be more easily achieved for all circumstances. The Beyond Rig began the process of creating a self-taught AutoRig for all levels of experience. All elements provided are integrated with explanations and thorough breakdowns of all necessary elements. Furthermore, while the AutoRig is minimized for ease of use, it was created with a secondary purpose of being boundless. The Beyond Rig can rig options as simple as a human, as fused as a Chimaera, and can even go as far as to rig a fractal-like form while covering everything in between. This AutoRig allows only one limitation, the user’s imagination.

Automated **Rigging**[1] Systems, or **AutoRigs**[2], are designed to replace the role of the **rigger** [3] in the animation process. However, all current AutoRigs require some amount of rigging knowledge. Based in personal experience and user input, Beyond Rig was constructed to be easily maneuvered while simultaneously having minimal limitations to its capabilities. The issue with existing AutoRigs of the past is that they fail to cater to the experience and perceived knowledge of their users. Furthermore, as technology expands, so should the ability of the AutoRig to create new and uninhibited forms. This project seeks to create a rigging system that at its core provide the user as many options and customizations for their needs. This is the culmination of everything I learned and came across during my many phases of education.

//Personal Background

There were many factors that have contributed to my Animation path, and specifically the Rigging subset. At a young age I studied Zoology, learning about the diversity of the living form throughout the Animal Kingdom and began my understanding of the varying forms of motion that exist. This led to my study of Biology, learning about the structural composition of living forms and understanding how the physiologies of even the most fantastical creatures. In my undergraduate education I studied Architecture which furthered my understanding of engineering and, on a conceptual level, how to create the illusion of structural stability. My minor in Classics took all of the varying aspects of my education and consolidated them into a single cohesive

---

[1] An underlying structural element constructed in order to move a rigid model for the purposes of animation. This is commonly compared to the strings on a marionet that allow someone to "animate" the otherwise still puppet.

[2] Allows the user to create a rig with limited knowledge about rigging. A means to expedite the rigging process.

[3] An underlying structural element constructed in order to move a rigid model for the purposes of animation. This is commonly compared to the strings on a marionet that allow someone to "animate" the otherwise still puppet.

study, understanding how the Mythological creatures could have evolved and been logically constructed in the physical realm. For this reason, when learning digital modeling, one of my first creations was that of a Gryphon, which merged a quadrupedal lion with and avian eagle into a symbiotic anatomical structure. The problem I found was that existing AutoRigs could only do one creature at a time, and a fusion such as this was not conceivable.

When doing further research into existing AutoRigs for a collaborative assignment it was learned that the variability of the AutoRigs was exceptionally scarce. The available options to rig were a biped, a quadruped, or bird which were a very limited set of options knowing the variability alone within the Animal Kingdom. Beyond the AutoRigs, I had always pushed my personal capabilities through unique anatomical forms during my rigging education, such as a shrimp, a winged bat and a myriad of inorganic forms. This taught me the varying needs between rigs, while also overlapping applications for the same rigging techniques. This would later help expedite rigging methodology by applying similar steps to analogous structures, but I had realized how much work would be necessary for anyone not as experienced with rigging to use even an AutoRig, intended to be designed for such a purpose.

When constructing my initial Thesis project, I created a multitude of character designs to push the limits of my rigging capabilities. One character was a simple bipedal character throughout his lifetime (Fig. 3.1), with which meant he had the same basic anatomical structure but varying proportions as he aged. For this reason, a rig set up that could be altered for proportional variances, while retaining the structure was necessary. Another creature I created was designed as an exaggerated Centaur, the structure based on the mythological stories. Using current AutoRig options, the two halves of the mythically fused creature would have to be rigged independently and manually fused. However, the most difficult creature to replicate through available rigging aids was a centipedal creature with tentacles for limbs (Fig. 3.2) which is numerous levels of rigging needs. These were problematic even for me, and I couldn't imagine the difficulty anyone with limited knowledge would suffer through if they wanted to break beyond what was currently available. Due to the amount of difficulty I faced in my own designs, I began to create an AutoRig to expand the design opportunities for independents and non-riggers alike so that they may expand their character physiologies.

//Presentation Style

One of the greatest issues during the rigging process is the communication disconnect between the rigger and the client. Therefore, in the same manner by which the AutoRig seeks to communicate the necessary information as clearly as possible, so too through paper do I aim serve to aid the reader to understand and propose necessary requests of any rigger. One of the common flaws experienced between riggers and animators is that of communication. Within the rigging community simple use of necessary jargon permeates their understanding, but such terminology must be limited at times when working with those who use the rigs. For instance, when creating a rig for a user's express needs more colloquial language must be utilized to convey greater concepts for the technical construction of the rig. For the creation of the rig, this was a driving force to reach an audience beyond those with a technical mindset as is a common issue amongst existing AutoRigs. Additionally, in this written supplemental explanation of the design of the project, the intended audience will not be limited to the rigging community.

This written piece, not unlike the **scripts**[4] that govern the AutoRig construction, aims to guide those seeking a rig creation to interact as clearly with a rigger as possible. Therefore, terminology has been simplified to convey all necessary concepts to the reader. Furthermore, a step-by-step description of how the AutoRig is used goes against the nature of its construction itself. Beyond AutoRig was designed to be as user-friendly as possible by limiting all extraneous material necessary to explain itself. The following description of the structure will instead serve to provide an understanding of the work involved to create an extensive while effective rig, while allowing any user a descriptive understanding of the available mechanisms that can be integrated into any rig. It is with logical understanding that an AutoRig provides a quick opportunity for any user to integrate a rig onto any model, but every rigging need will vary and not every AutoRig may provide the exact needs for their project.

//Construction

Anyone beyond a professional studio, such as students and independent animators, shouldn't be limited by their own knowledge of rigging. Many films and independent pieces are restricted by the tools available and one obstacle for many comes with rigging step. It is therefore the need of an AutoRig to merge the separation of the artistic and logical realms, allowing imagination to be limitless while still allowing it to be controlled within the technical capacity of the program, in this case, Autodesk **Maya**[5].

Unfortunately, most, if not all, rigging tools are limited to the bipedal and quadrupedal forms, with some additional variation thrown in for alternately existing anatomical parts. When it

[4] Similar to a code, is a compilation of instructions that are understood by a specific program to generate a given set of written actions.

[5] The program used by the public for Animation purposes due to its focus on the varying needs within the medium.

comes to the mythical beasts which combine anatomical structures of disparate creatures, or the gods of classical and contemporary creeds which extended beyond the banal humanoid, there isn't anything readily available. The rarity of creatures as shown in *Zygote* by Neill Blomkamp (Fig. 1.1) is staggering and albeit a horror piece, there should not be anything to prevent the regular appearance of unique creations. The only limitation for artists should be their imaginations when the technology exists to expand their options and therefore the Beyond Rig was born.

//Process

The initial concept began where others had started, a biped, but personally designed characters fell far outside the generally available AutoRig anatomies. Students constantly simplified their designs because of the limited AutoRig options available[6]. Additionally, editing and adjusting the characters, that even the slightest adjustments reverberated down the pipeline and would require the rigging to be updated accordingly with every structural adjustment. For this reason, I sought a project that could roll with the waves of change smoothly and seamlessly in between the modeling and animation processes. The last thing anyone wants to do is recreate the same character rig every time the model changes, and I wanted to include the functionality to make quick alterations without requiring a complete overhaul.

When handling a project with infinite possibilities it was imperative to focus on simplicity for the users to be able to navigate the program efficiently. One method to remaining minimal was to provide as few components as possible while promoting enough functionality to cover all possible needs. So many previous variations rely on separate windows, broken down

[6] Personal Observation during Collaborative Course, Winter 2014

into tabs, or other framing methods, but these also restrict the users. When dealing with infinite creature variations, there are infinite "windows" that are necessary in order to control and edit all features of the rig as necessary. For this reason, the first step was to create an alternative means of setting up and updating the body parts and **joints**[7] therein. By connecting the list of editable elements directly to each body part, it would ensure that the user could easily work on a component basis without having to worry about clear and unique naming specification to keep them separate. Furthermore, by allowing the direct connection, one is only required to find the component visually in the panel and edit the **attributes** [8]of the base **controls**[9] in the **Channel Box**[10] which update in real-time. Now while all of this is clearly visible to the user, a necessary set of nomenclature attributes were concealed for a clear and concise means for the script to locate all the elements as needed.

Due to the possible complexity of the rig, it was imperative to focus on a means of uniquely naming and categorizing elements for ease of locating and organizing them. The basic consideration was to keep a taxonomical naming system which was built on a Dewey Decimal - based set of numerical data. The first component was the body part, which outwardly is an alphanumeric name derived by the user, but as more components are added to a base joint, the higher the body part number. For instance, the spine and legs extend from the base hip joint, and therefore they would be assigned a 0, as is common coding baseline, followed then by 1 based on

[7] Named after the anatomical feature, this is a rigging element is used to construct to represent a skeletal form to allow for the manipulation of a given geometry.

[8] A defining aspect of a given element, such as a spatial reference, visual appearance, as well as a user defined feature.

[9] A feature necessary in animation for the manipulation of the skeletal joint structure to move and pose the character.

[10] A side panel where a user can view and edit the visible attributes of a selected object.

which is created first. In this way, beyond the personalized name of the appendage, there is a simple means for the script to categorize each newy created component.

The next step focuses on the directionality of the body parts which extends beyond standard mirroring as many rigs have. The arrangement is based on the construction of **NURBS**[11] objects, which use “Sections” and “Spans” that represent coordinates for grid mapping on a surface and “Sweeps” which represents how far around a given surface extends like a paper fan opening. Through these three specifications a unique array of similar elements may be created with equal spacing, for example Medusa’s head, which is a rounded surface, that has snakes spaced evenly across it. However in the Beyond Rig in place of the internal Maya jargon, the terminology was altered to reflect a more user-friendly translation with the use of *Longitude*, *Latitude*, and *Degree Range*.

The longitude element revolves perpendicularly around the part control, similar to a paper fan locked to a given point. Therefore, in a simple arrangement using mirroring, the longitude number would be 2 where the 1st would denote the “right” side, and 2nd denotes “left”. Furthermore, the degree range would be 180 because the first body part is always at 0 while the second would be at the highest degree which would be 180. However, when handling a creature like an octopus with tentacles rotating equally spaced in 8 directions, each tentacle is assigned a number 1-8, starting with the “rightmost” tentacle at 0 and extending around with the understanding there isn’t a duplicate at 0 and 360. In addition, just like on a sphere, directionality isn’t simply two-dimensional and therefore a matrix system is necessary to apply a second dimension to the duplication. This way the user can automatically create duplicates that are

[11] Short for Non-Uniform Rational B-Spline. Simply understood as 3-Dimensional design based on curves, using mathematical degree measures to create the surface between them.

equidistant and rotated evenly while not being required to do calculations or know extensive commands within Maya. with the opportunity to adjust as necessary after creation.

The next level for the body parts are described as *Duplicates* because they are same element but unlike the Direction level they are intended to be manually edited in vary forms. The most common example would for fingers on a hand because they are the same basic structure but each finger has variable lengths. One thing to note is that the fingers are a separate element from the thumb because the thumb has the ability to orbit at its base. Additionally, by connecting all of the fingers under a single hierarchy they can be controlled as one with options such as curl and spread, that are automatically assigned to all of them. Lastly are the individual joints, placed separately including the tips of the appendages, with increasing numbers from the base.

At this point every joint has a unique numerical name ending with five numbers denoting the body part, directionality in longitude and latitude, set number and lastly the joint. Preceding this numerical set is the numerical set assigned to the joint to which the body part is connected. For instance, along the spine the head and neck would extend from the topmost joint, ending the spine joint numerical name in the highest number, while the arms may extend from the second to last, therefore the numerical name preceding the arms would be one less than that of the head and neck.

The next difficulty is with potential for repeating types of body parts that are not part of the same set. For example, a character with arms protruding from their heads as well as their chests, such that they act the same but would be linked to separate base joints. For this reason, to expedite similar functionalities across both instances, this similarity must be noted. Within the attributes will be a list of premade body part types from which to choose, such as spine, finger, tentacle, or even wing. This also allows for necessary elements to be provided to determine how certain capabilities of the body part will occur. Simply, for fingers, the knowledge that they can spread away from one another as a unit, or tentacles that move freely, but can get locked down when suctioned to along the body.

//Premade Physiologies

Although the formatting for the AutoRig reimagines the production and layout methods, in many circumstances an expedited rig is necessary. Taking notes from the existing rigs set-ups which provide a limited selection of predesigned structures and applying personal understanding of atypical anatomical structures both existing and mythical, a premade selection of existing rigs was implemented. When beginning the rig, the user is given the option to start from "Scratch", but if not they have alternative options to chose from inducing but not limited to Humanoid, Quadruped, Avian and Centipedal. It is encouraged that the user chose the closest option to what can be used with their model because it is edited before completion, while also being made like the rest of the so that it can be manual adjusted as well.

Once a premade option is selected, a similar procedure is implemented through the Channel Box specifying the necessary body parts along with important details for arrangement. Basic details include the number of said body part, the joints for each, and if they will be symmetrical in any manner. Now to provide unique editing and construction specifications the user can first specify from which existing body part it will extend such as the arms from the spine, and the fingers from the arm. Furthermore, because numerous body parts may extend from the same parent part, further joint specification is necessary. Therefore, from the spine, for example, the neck and head construction would extend from the last joint while the arms would commonly extend from an earlier joint along the **chain**[12]. By preemptively arranging these details, this reduces the number of adjustments needed later for the user, further expediting his or her use of the AutoRig.

//Part Specifications

When properly setting up the joint chains, it was necessary to include a means for the rig to understand how the part would behave. Therefore, beyond simply naming each “body part”, internal nomenclature was added to be assigned to each newly created part. The base application for this specification is for the chains themselves to be created accordingly. Applying the same type of Inverse Kinematic, or IK[13], to every type of chain would result in uncontrolled results. For instance, a spine should not act in the same way as an arm unless the character is expected to break their back regularly. However, the arm may act similarly to a spine in the case of a bendy joint set-up, where the bones are not always rigid elements but rather act more like rubber hose animation. This is the first necessity when applying controls to the created joint chains.

The specification of the body parts is not only imperative for the join chain assignment, but also allows for a better controlled joint chain by the animator. The secondary provision from specification is the addition of supplementary animation controls for the manipulation of the joint chains. For instance, quadruped spines can use a center of gravity modification for the shifting of weight between the fore and hind legs. Another application of this feature is the fine-tuned animation of feet rolling on all four sides all while being able to keep the toes planted as

[12] Connected joints linked hierarchically used to represent structures like body parts.

[13] Controlling the joint chain when both ends of the chain are placed such that all the spacing between the joints is unaltered, but the chain as a whole contorts accordingly between the two points.

well. Every body part has a unique set of controls and by integrating all necessary animation needs into the rig creation allows for the users advanced manipulation capabilities.

//Running Scripts

Expressions[14] are commonplace element in many rigging scripts, but the more there are the more memory intensive the file becomes and with infinite possibilities this is a serious problem. For this reason, the script initially had a single expression to which all elements were connected attempting to limit the number of **nodes**[15] that could be running simultaneously. The first issue when working with expressions is their failure to run "On Demand" when values change. This causes the script to run uncontrollably and therefore either too late for the need or diminished the computer's available memory when run unexpectedly even if precautions to prevent actions to occur out of turn are in place.

Similarly, there is the option of **Script Nodes**[16], or nodes that contain a script within it that can be run during very specific instances, such as the opening of the GUI or file. Do to the limited options attributed to running the script, it does not have much value during the use of a file. Alternately, is the implementation of **Script Jobs**[17] which unlike the previous options are hidden scripts, that are more rigid in their application and operation. The difficulty is that they remain applicable during a Maya session similar to non-global procedures and stated variables

[14] Controlling attribute values through mathematical equations driven by other attributes, time and/or other numerical value input.

[15] A simplified representation of all attributes that determine a given objects existence within the file. Can be linked together by like attributes wherein one controls another through a downstream transfer of data.

[16] This node is specifically designed to retain a given script and run the script at a specified point such as the file opening or close.

[17] This script is linked to a given element, such as an attribute, and automatically is run when a specified condition is met.

but can be permanently attached like global procedures but are far more difficult to remove due to the invisible nature.

For this reason, combining Script Nodes and Script Jobs can yield positive results. The file contains scripts comprised of procedures, which will run every time the file is opened. Furthermore, within the Script Node is an embedded Script Job which initially connects current objects with the appropriate script to run when the attribute is changed. This is accomplished most easily by connecting the applicable nodes to the Script Node and listing out incoming connections to send the internal script back. This can also be applied to newly created elements by simply constructing the script run to link to the Script Job initially and connect the objects to the Script Node so that it will be reconnected upon file opening. This double layering allows for everything to happen currently and remain connected as is necessary for the user to save and reopen the file without loss of updating.

//Packaging & Documentation

The standard method by which to relay information to users is a text file opened separately while with scripts, the creator sometimes integrates descriptions into the beginning of the **code**[18] the user runs. This code's unique structure, allows for the actual explanation of the elements to be integrated seamlessly into the visual structure of the code. Just as with all other elements, the descriptions are Channel Box attributes beside their active counterparts that can be called for explanations and examples. The options can be hidden during regular use, but easily shown at any time through a generalized "Show Descriptions" attribute under the main rig start element.

[18] A written language component providing a instruction that is understood by the processor or a program to generate a given action.

One common issue non-technically minded users come across is the use of scripts, either in the **Script Editor**[19] or scripts folder. I was constantly contacted by non-rigging students with issues involving the loading of scripts for varying purposes[20]. For this reason, the script is embedded into a saved file with all scripted elements preloaded and integrated into the file. Additionally, to assist users in understanding the basic construct of the rig, the initial visible objects in the file begin the process of instructing the user on how the rig builder is structured. A simple structural example is provided, with descriptors provided in both annotative titles as well as extended descriptions on a per attribute basis located alongside the attribute.

//Under the Hood

The entirety of the AutoRig is geared towards the user including many components that help keep the program functional with potential user interference at times. One common issue witnessed with any rig was the accidental **parenting**[21] or, more often, unparenting of certain elements associated with any rig. For this reason, the design of the AutoRig has included some hopefully unchangeable elements to retain the structure of the proxy set-up. All body are organized into **sets**[22] which are then separated into component and **child**[23] sets. The first set refers to any elements for a singular specified body part, including the part control, directional and duplicate controls, and finally all proxy joint objects since they will all be treated as one cohesive unit in later stages. Conversely, any additional elements that are directly connected to

---

19 An internal window that allows for the writing and running codes written in either MEL or Python. A secondary view panel also shows a textual history of everything that occurs.
20 Personal Observations, Fall 2012-Summer 2014
21 The hierarchical term for an object being in control over another, referred to as its child.
22 An organizational tool that can quickly link specified elements for easy referencing when necessary.
23 The hierarchical term for an object being controlled by another, referred to as its parent.

the specified body part, such as fingers to an arm, will be located under the second folder so that, as necessary, they may remain linked. By using this secondary hierarchical structure, it is hoped that any potentially accidental rearrangement of elements will not damage the integrity of the rig.

Additionally, the interconnectivity of components served another use. Through the use of the object nodes, information could easily be shared and link all related components. Though certain attributes are not visible to the user per the Channel Box view, they are still represented in the node view. For instance, standard alphabetic data such as names exist in relation to an object but are not available through basic means of editing. For this reason, the rig utilizes the inability of the user to reach this form of data to store necessary details for the object. For instance, in case the name of a given component is changed after the fact, the prefix for the specified character rig as well as it's part specification are stored within all of the applicable objects. By doing this, the structure will always be able to locate and separate all rigs and their components for later steps of production without relying on potentially user editable attributes.

//Planning Conclusion

To further the variability of the project, all potential animatable features will be integrated into the script. By integrating their potential into the options, every project will be made to the unique specifications of every character or animator. There is a basic set up already, applying simple **Forward Kinematics**, FK[24], into the initial joint creation. The further options will be broken into two methods of application. In one option, the specified body part from the **proxy**[25] set up will be used to apply commonly added capabilities to the given body part. For example, feet will

[24] When the control of a chain is on a joint-by-joint basis, focusing on the rotations of each one affecting the joints down the chain.

[25] For the purposes of an AutoRig, this feature is used to stand in for joint locations until the arrangement is complete, after which the joints are generated at the given points.

have tipping edges and roll options built in with additional markers as needed to apply such features. The second application will be fore elements such as bendy features, which are not part specific but by the needs of the rig itself. These will be made available once the basic rig structure is laid out, so as not to bog down the user's options throughout the construction phase.

The construction of the overall script is geared towards editability, and this is not strictly limited to during its creation. Many of the editable features are for variability between users, but sometimes the project itself will be edited during the process. For this reason, the final feature to be integrated will be the transfer of the skinning from one geometry to another. The rig will take into account how it manipulates the original geometry and if placed similarly with the new geometry the influence will be transferable to a new model. This will be accomplished through the use of the control vertices along with proxy skinning objects to match previous forms to the edited anatomy, such as lengthened or moved body parts, are used for retaining the skinning work that was previously done.

Similarly, the simple geometric forms will be used for less confident users to create a basic skinning application that can be stored in the background for reapplication during editing. More confident users can apply the skinning more manually, allowing some personalization and editing from the users. When the geometry is edited, the prior geometry's control vertices will be logged in reference to the joint locations, so that when the joints are shifted the proximity can relocate the skinning values. With elements like this the user will never be limited by the rigging process for their continual design editing throughout the production as the need arises. That is the final and necessary step to provide an AutoRig that is fully functional and allows designers full technical capabilities along with uninhibited design conceptualization.

//Beta Test Results

A beta test was run during which a few alterations were found to be necessary for the success of the Beyond AutoRig. When designing the AutoRig the user began the file with an example appearance featuring all of the rig components as they would be laid out during the construction of the rig. Although this, coupled with a few prompts and annotative markers, was believed to help guide the user, the Beta Test proved it was not sufficient. Therefore, the expected adjustment will be comprised of a step by step introduction for new users. They will be prompted to specify if they were comfortable with the rig and if not will be led through each component of the rig and its unique attributes and purpose. By doing this it will ensure that the user is sufficiently acquainted with everything. Coupled with the existing structure of the AutoRig, further questions throughout the rigging process may be answered by the explanations integrated into the components themselves.

The test was also illuminating to some uniquely minor clerical errors throughout the rigging script. For example, the introductory window which helps name a rig for separation of different rigs within the same file, mistakenly requested a body part name which is a reference to a window accessed later for creation of appendages. Additionally, beside the existing attributes shown in the side, it was made known that their appearance wasn't bold enough and therefore became lost amongst the standard appearance of the other attributes present. Therefore, a color coding system will be applied to clearly outline the AutoRig specific attributes and their particular function. For example, all component descriptions may be slightly greyed to lower their integration in the standard attributes but still be present to be accessed. One noticeable issue was the appearance of all the Script Nodes under all objects they were linked to. Any alteration to the names would detract from any updates to the rig and therefore a simple lock will be necessary. Overall, once introduced to the rig and its components, the beta test went smoothly. All future versions will serve to benefit the user's understanding and interaction with the overall rig structure.

//Graduate Experience

This project was necessary, and that is why found myself creating it before I had even chosen to do so. In every project through both my undergraduate and graduate level work, I have attempted to find a purpose beyond myself. My desire within such a creative field was to present information and experiences in a way that traditional film could not. It is for this reason my initial thesis topic focused on promoting Animation over Live-Action by way of showing how the Live-Action remakes were unsuccessful by comparison to their usually Animated originals. I wanted the field, which at the time appeared to be rivaled by Live-Action with Visual Effects, to have a clear future.

In my process I had to pushy the bounds of Animation and create elements that were very clearly Animated, while also providing a promotional undertone. This proved more difficult the more I strove to push the limits of Animation itself within a more independent production framework. I was working alone and although I had the basic skillset to do most of the production myself I began to question how anyone else could even fathom doing this. As a side project of sorts I began writing scripts that I designed for my own personal use but would practically be applied for others in the future. I created small elements commonly found in AutoRigs but knew that the available options could not be applied to my characters.

I had created non-standard character forms trying to push my personal understanding of plausible physiologies but became aware that rigging would require a unique set-up for each character. But I questioned why it was so difficult to create a rig for anything beyond a humanoid and quadruped with some additional variants available. It was at this point that I realized creating the scripts for myself would be selfish even if I made them publicly available. Many of the students beyond the rigging courses that I interacted with had varying levels of difficulty with the scripts that could have assisted them in their work, and therefore it was clear providing my scripts would only assist those capable of making their own. I began to weave the individual scripts into a cohesive singular element which would require progressively less experience to execute. As I did so I learned to overcome what I found to be an unstable balance of simplicity for ease of use and structural manipulation and control for variability.

Having already experienced the expansive potential of character design within my own work I had already considered what a potential rig may need. The first consideration came from the design process itself, having edited the characters regularly based on committee input. Having been the Technical Director in my Collaborative Course, we had specifically agreed to use an AutoRig because of the potential for editing, but the need to change the rig regularly became more of a nuisance. I therefore wanted to create an AutoRig that was constantly able to be constantly altered based on wherever the design process took the character. Furthermore, I wanted the rig to not be limited by anything, just as I had designed my characters without fear of the rigging process, so to did I hope for this project to do the same for those less experienced than I. For this reason, I initially designed the Beyond AutoRig without any structure, allowing the design of the character to fully dictate where and how body parts were connected. Additionally, all parts could be defined as existing types that it understands so that during the final steps the applicable controls and construction may be applied.

Glossary

AutoRig

Short for Automatic Rig. Allows the user to create a rig with limited knowledge about rigging. A means to expedite the rigging process.

Attribute

A defining aspect of a given element, such as a spatial reference, visual appearance, as well as a user defined feature.

Chain

Connected joints linked hierarchically used to represent structures like body parts.

Channel Box

A side panel where a user can view and edit the visible attributes of a selected object.

Code

A written language component providing a instruction that is understood by the processor or a program to generate a given action.

Expression

Controlling attribute values through mathematical equations driven by other attributes, time and/or other numerical value input.

Child

The hierarchical term for an object being controlled by another, referred to as its parent.

Control

A feature necessary in animation for the manipulation of the skeletal joint structure to move and pose the character.

FK

Short for Forward Kinematics. When the control of a chain is on a joint-by-joint basis, focusing on the rotations of each one affecting the joints down the chain.

IK

Short for Inverse Kinematics. Controlling the joint chain when both ends of the chain are placed such that all the spacing between the joints is unaltered, but the chain as a whole contorts accordingly between the two points.

Joint

Named after the anatomical feature, this is a rigging element is used to construct to represent a skeletal form to allow for the manipulation of a given geometry.

Maya, Autodesk

The program used by the public for Animation purposes due to it’s focus on the varying needs within the medium.

MEL

Short for Maya Embedded Language. The internal coding language upon which Maya is run and allows an external user to program features and elements within the program.

Node

A simplified representation of all attributes that determine a given objects existence within the file. Can be linked together by like attributes wherein one controls another through a downstream transfer of data.

NURBS

Short for Non-Uniform Rational B-Spline. Simply understood as 3-Dimensional design based on curves, using mathematical degree measures to create the surface between them.

Parent

The hierarchical term for an object being in control over another, referred to as its child.

Proxy

For the purposes of an AutoRig, this feature is used to stand in for joint locations until the arrangement is complete, after which the joints are generated at the given points.

Rig

An underlying structural element constructed in order to move a rigid model for the purposes of animation. This is commonly compared to the strings on a marionet that allow someone to “animate” the otherwise still puppet.

Rigger

A position in animation where a person creates skeletal structure for a provided character form, or model, in order for it to be manipulatable during animation.

Script

Similar to a code, is a compilation of instructions that are understood by a specific program to generate a given set of written actions.

Script Editor

An internal window that allows for the writing and running codes written in either MEL or Python. A secondary view panel also shows a textual history of everything that occurs.

Script Job

This script is linked to a given element, such as an attribute, and automatically is run when a specified condition is met.

Script Node

This node is specifically designed to retain a given script and run the script at a specified point such as the file opening or close.

Set

An organizational tool that can quickly link specified elements for easy referencing when necessary.

Existing AutoRigs

Advanced Skeleton – Animation Studios

Hermes - Jonah Reinhart

Mixamo – Adobe Systems Inc.

Rapid Rig – Dustin Nelson

The Setup Machine – Anzovin Studio Inc.

Film Stills

Fig. 1.1 *Zygote.* Short Film. Blomkamp, Neill. 2017. Canada: Oats Studios, 2017.

http://za.ign.com/neill-blomkamp/108786/feature/neill-blomkamp-talks-his-new-sci-fi-horror-short-zygote

Project Stills

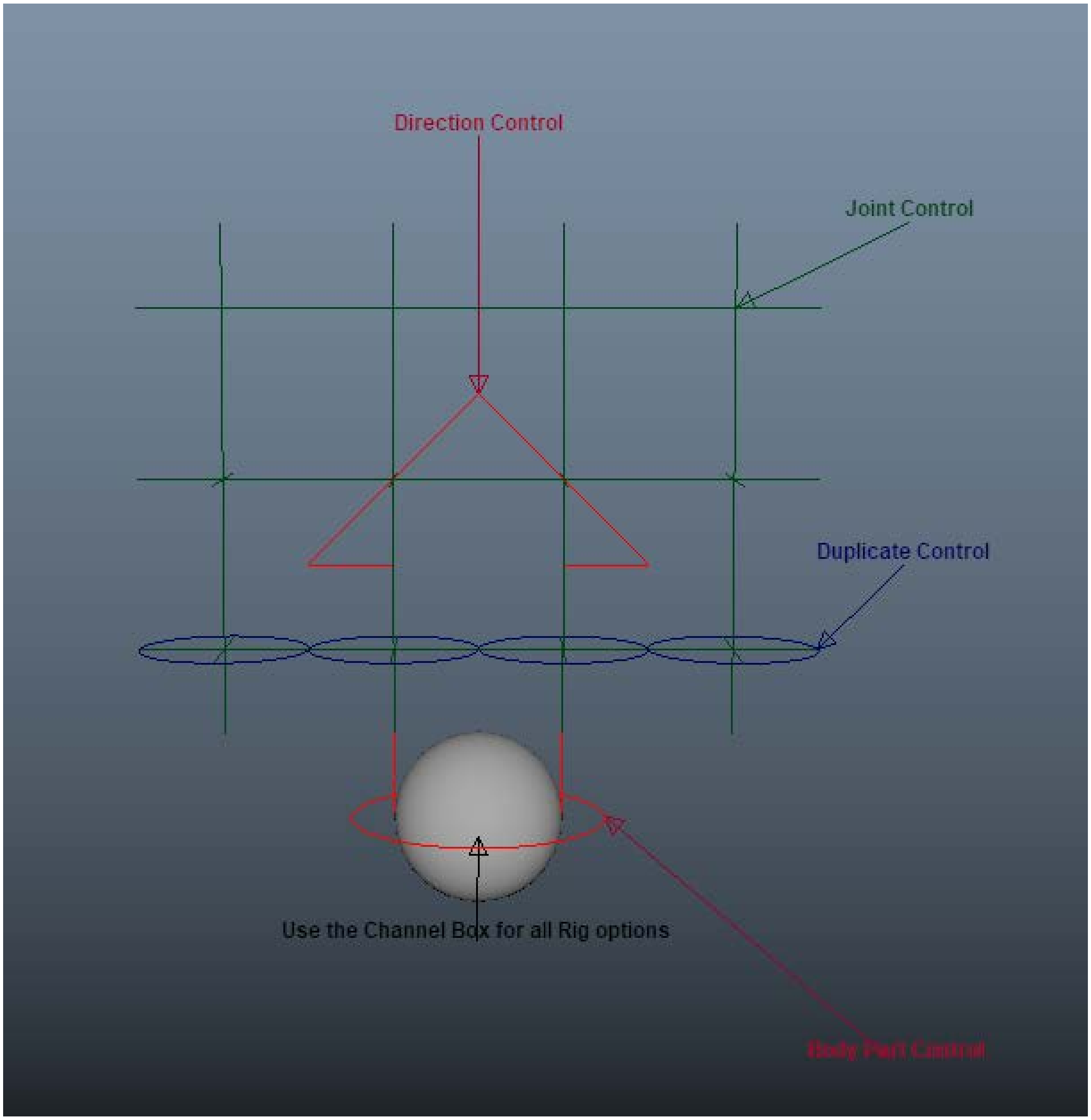

Fig. 2.1 – Introduction View

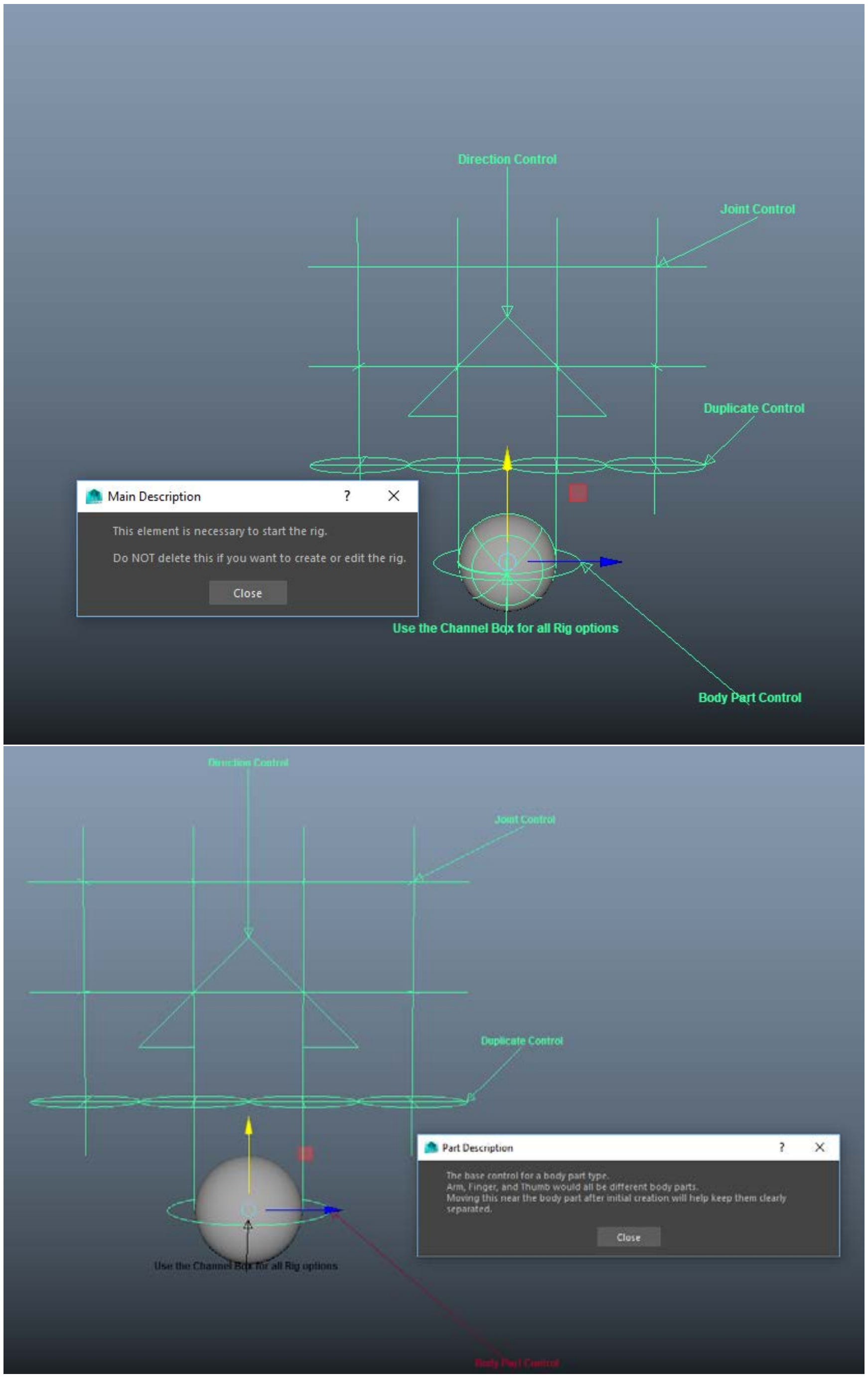

Fig. 2.2 – Descriptions

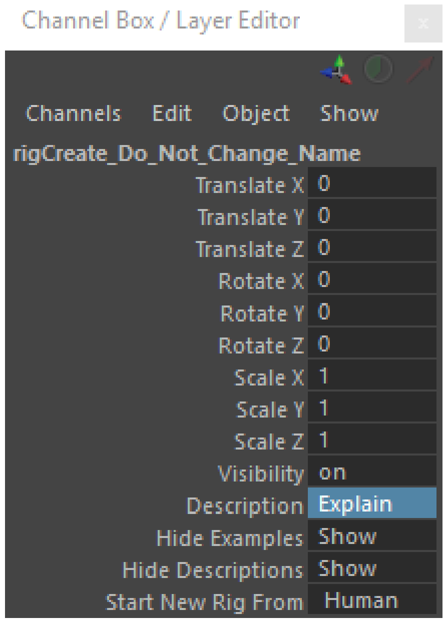

Fig. 2.3 – Channel Box Attributes

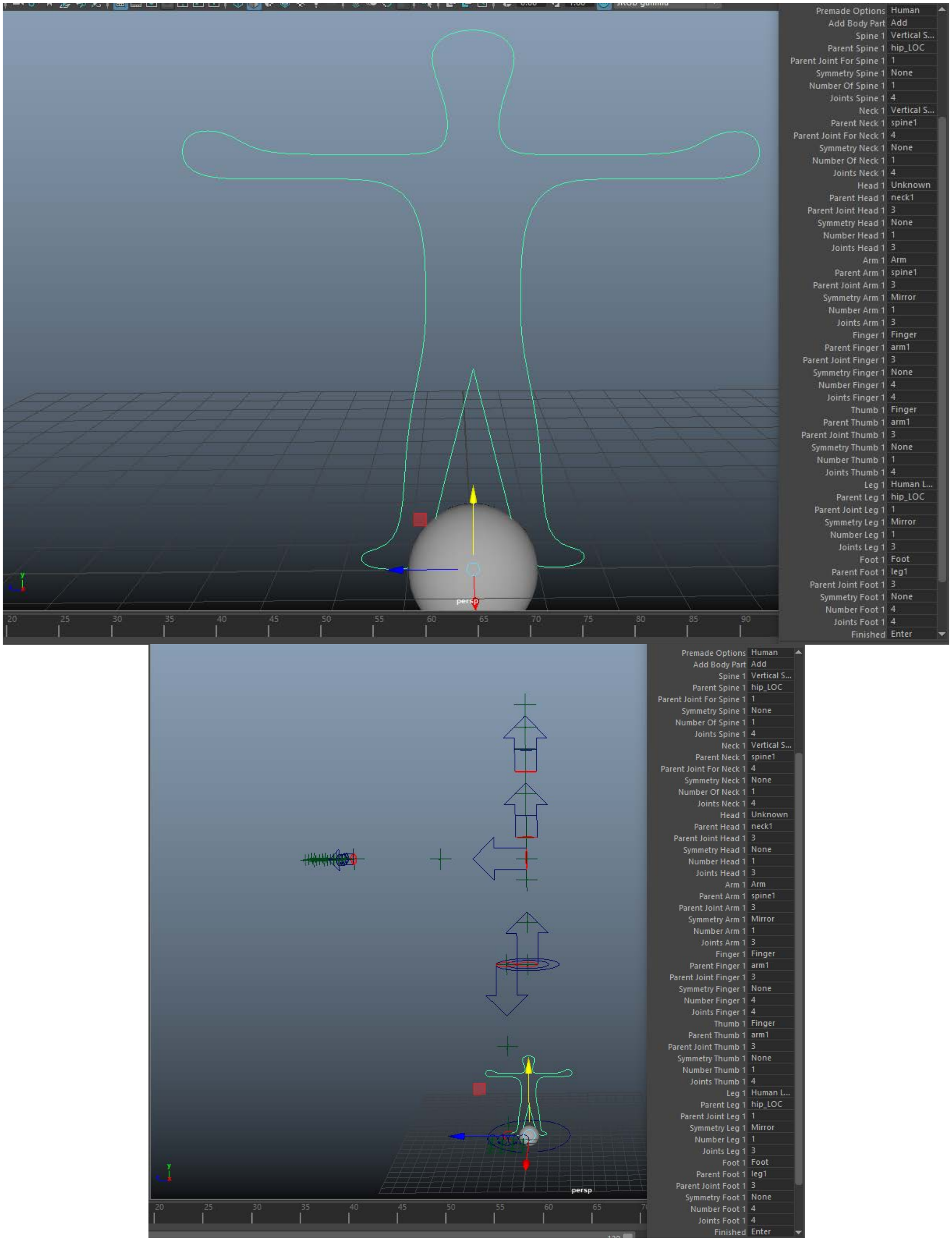

Fig. 2.4 – Premade Rig Set-Up

Character Designs

Fig. 3.1 – Humanoid Model

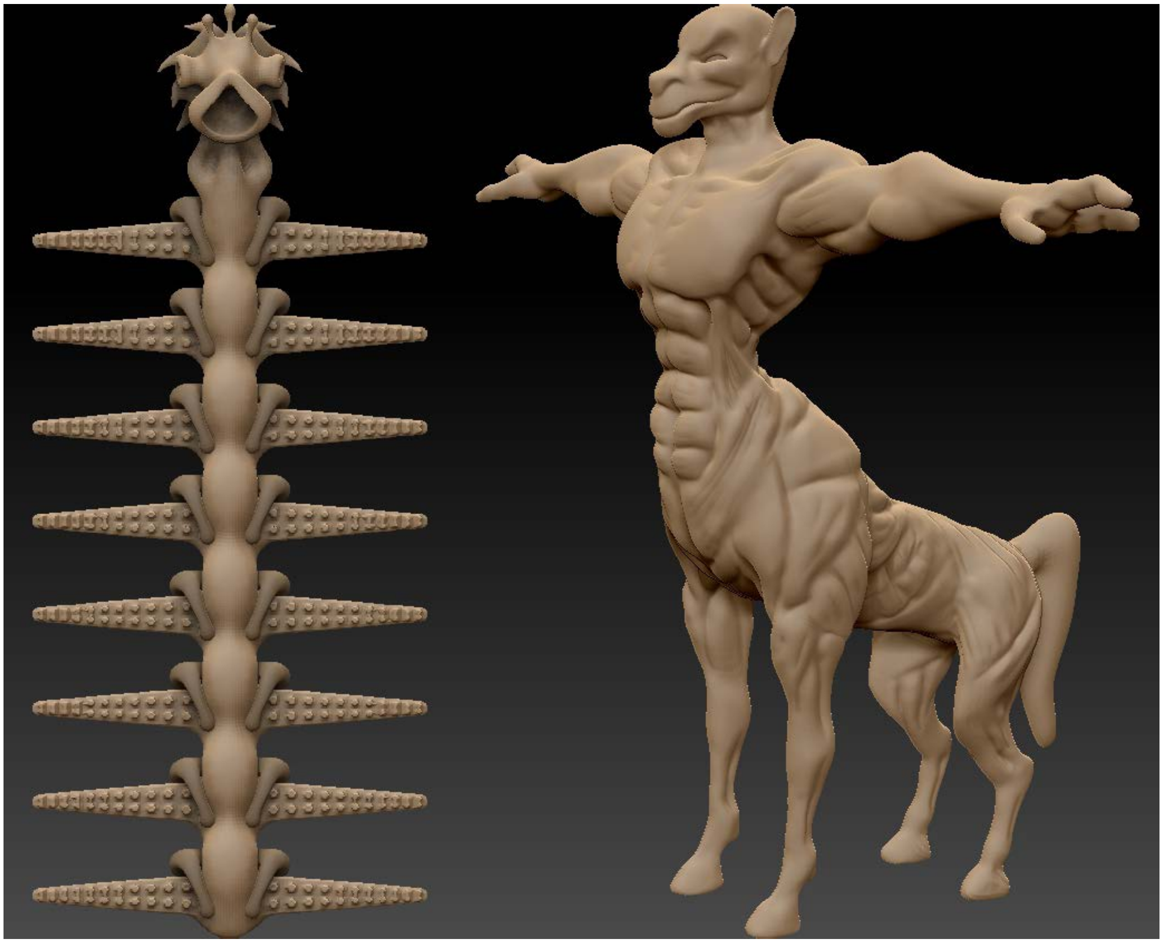

Fig. 3.2 – Non-Human Models

Bibliography

Maya Technical Documents. Autodesk, 2018. http://help.autodesk.com/cloudhelp/2018/CHS/Maya-Tech-Docs/Commands/index.html.

O'Hailey, Tina. Rig It Right!: Maya Animation Rigging Concepts. New York: Focal Press, Taylor & Francis Group, 2013.

Pisca, Nicholas. *Ysyt: Maya MEL Basics for Designers*. Los Angeles, CA: 0001d Publishing, 2008.

Wilkins, Mark R., and Chris Kazmier. *MEL Scripting for Maya Animators*. San Francisco (etc): Morgan Kaufmann Publishers, 2005.