A WEB-BASED ANIMATION AUTHORING APPLICATION FOR QUADRUPEDAL CHARACTERS

A Thesis

by

KRISTA LEA MURPHY

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Chair of Committee,	Tim McLaughlin
Committee Members,	Ann McNamara
	John Keyser
Head of Department,	Tim McLaughlin

December 2014

Major Subject: Visualization

ABSTRACT

Creating animation for quadrupedal characters via key-framing is time consuming; furthermore, motion capture is expensive and cannot easily be applied to animals. Procedural animation can combat these limitations. However, procedural animation requires a certain amount of technical and mathematical prowess. In order to address these issues, this thesis delivers a web-based application capable of creating expressive animation using a procedural approach. The web-based interface allows the user to create a gait cycle for any quadrupedal shape through a graphical user interface (GUI) that enables the animator to easily modify character representation and gait parameters. Once the user has obtained their desired animation cycle, the animation data from the new web-based application can be exported directly into a preferred animation pipeline. This system provides a lightweight tool that saves the user time and requires minimal expertise. It also does not add to the cost of software and/or hardware and facilitates animation authoring from any computer with internet access and a web browser.

To My Family

ACKNOWLEDGEMENTS

I would like to thank my committee chair, Prof. Tim McLaughlin, and my committee members, Dr. Ann McNamara and Dr. John Keyser, for their guidance and support throughout the course of this research. I also appreciate the contributions of Spencer Cureton, Junze Zhou, Jorge Cereijo-Perez, Jacob Zimmer, and everyone else involved in the Perception Based Animation research group who had a hand in developing the foundation for this thesis.

Thank you to all the faculty, staff, and students of the Visualization Lab for providing such a unique, friendly, and helpful environment to study. I've never once questioned whether I made the right choice by enrolling in this program. The knowledge I've gained throughout my time here is invaluable.

Finally, thank you to my family and friends for always supporting me in everything I do and providing me encouragement when I needed reassurance.

This material is base upon work supported by the National Science Foundation under Grant No. 1016795. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect the views of the National Science Foundation.

TABLE OF CONTENTS

Page

LIST OF FIGURES

FIGURE Page

1. INTRODUCTION

The demand for highly realistic quadrupedal motion is on the rise. The Croods, Brave, and World of Warcraft, are just a few examples of recent film and game productions that feature multiple quadrupedal characters. While the growth of quadrupedal characters remains steady, the need for both a quick and successful method for authoring quadrupedal animation is evident.

There are three main approaches to creating animation: key-framing, motion capture, and procedural animation.

1. Key-framing requires the animator to manipulate the controls of a character rig to create specific poses over and over again throughout the performance. This process is often successful, yet, the animator must invest a lot of time perfecting their work and must be very familiar with the motion they are portraying.

2. Motion capture, on the other hand, has the ability to provide very realistic animation quickly, however, a clean-up process is almost always needed after the recording session to prepare the data for application to a character. It also requires a dedicated set of equipment and is very difficult to apply to quadruped animals [22].

3. Procedural methods can be challenging to use for character animation, but with the correct mathematical model it can avoid these pitfalls.

Along with the use of quadrupedal characters, the availability of web-based 3D content creation tools has also grown in recent years. Unfortunately, existing web-based animation tools generally focus on bipedal characters and have not taken

advantage of procedural methods for authoring quadruped character locomotion cycles.

1.1 Overview

In the application developed within this thesis, a quadrupedal character is represented by a dynamic rig. The rig is driven by mathematical functions extracted from video footage of animal locomotion. The components of these functions, as well as other gait parameters are stored in an online database to be accessed by the application. A graphical user interface (GUI) is provided to enable the user to manipulate the shape of the rig to match their target character. The interface also allows the animator to adjust the gait type, speed, and characteristics portrayed in the cycle animation. This thesis uses data gathered during a previous study to provide age and weight as the characteristic controls. With the collection of more data, the system may be expanded represent more characteristics. Finally, when the user has completed designing a gait cycle, they are able to export their animation to be used in their own pipeline.

1.2 Contributions

This thesis addresses the issues mentioned above by producing a web-based animation authoring system that focuses solely on creating quadrupedal motion through a procedural method. It aims to decrease the amount of time and expertise needed to create believable and expressive motion for quadrupedal characters. It also removes any requirements for additional equipment. Many advantages are inherent in an online tool. These tools are lightweight and require virtually no setup by the user. A web-based tool is available at any time and anywhere there is internet access, giving the user the freedom to work wherever they want on almost any computer and encouraging remote collaboration.

2. BACKGROUND

2.1 Methods for Authoring Animation

2.1.1 Key-frame Animation

Key-frame animation refers to the process of creating an animation sequence by starting with the key frames. In traditional 2D animation, the lead animators were responsible for drawing the characters in their key poses throughout the sequence. The less experienced animators were in charge of drawing the frames known as in-betweens. These frames depict the movement of the character that occurs in between the key poses, hence the name. It takes 24 frames to produce one second of animation in a film. Even if the film is shot "on twos", meaning one drawing is shown for two frames, the amount of work required to create a complete animated film is astonishing. In the 1970's, computers were brought into the process to interpolate the motion between key-frames. Eventually computer animation became an art of it's own, and in 1995, Pixar released *Toy Story*, the first ever feature length, computer animated film.

This type of animation requires animators to have a thorough knowledge of animation principles in order to convey personality. In 1981, Ollie Johnston and Frank Thomas, two Disney animators, outlined a set of rules widely known as the 12 Principle of Animation to help animators increase the appealing and believability of their characters [24]. In 1987, John Lasseter published a paper applying these rules to computer animation [15].

Although key-frame animation often produces very successful results, even with the aid of computers, key-frame animation is a time-consuming. The animator must manually interact with each of the controls on the rig to form key poses throughout

a shot. At times additional key-frames are needed between large motions to correct the computer's interpolation. Overall, the process that requires a large amount of knowledge and skill.

2.1.2 Motion Capture

Motion capture refers to the process of recording the movement of an object or body and is most easily applied to humans. This practice is popularly used in both the film and game industries to obtain the performance of actors for the purpose of being applied to computer generated characters. Generally, markers are placed on the target body where transformational data is needed. Either after the performance is finished or in real-time, the movement of each marker is mapped to a character. This mapping can be one-to-one, such as human performance mapped to a human character, or indirect, such as hand patterns controlling emotional state.

The use of motion capture began in the late 1970's, but has become widespread since the mid 1990's [23]. In the early 1980's, Tom Calvert attached potentiometers to a body and used the output to drive computer animated figures for choreographic studies and clinical assessment of movement abnormalities [4]. Jim Henson Productions along with Pacific Data Images gave birth to Waldo C. Graphic in 1988. Waldo was a computer generated pupped that could be controlled in real-time in concert with real puppets [26]. The technology continued to progress until it was picked up by both the game and film industries. In 1995, *Highlander: The Last of the MacLeods* was the first video game to use this process, and *Final Fantasy: The Spirits Within* became the first widely released movie to be made primarily with motion capture in 2001. Characters such as Gollum from Peter Jackson's *Lord of the Rings* trilogy (2001-2003) and, more recently, Caesar from Matt Reeves's *Dawn of the Planet of the Apes* (2014) are among this animation method's most successful results.

Motion capture is a great resource for achieving realistic animation; however, the amount of quadrupedal motion capture data available for use in animation is much smaller than it is for bipedal data. The process of recording the motion of an animal is much more difficult than it is for a human. A large amount of quadrupedal animals are wild, and therefore, require training or special precautions in order to set up a capture session. The equipment used for capture needs to be set up methodically. Commonly, dedicated studios are set up to permanently house the system. Large animals may require even larger spaces or mobile systems. Finally, once the data has been captured it is not instantly ready to be used for character animation. The data ordinarily contains a large amount of noise that must be filtered out before using the motion in production.

2.1.3 Procedural Animation

Procedural animation generates motion in real-time using mathematical functions which allows for much more involved actions than could otherwise be created with predefined animation. This method is not commonly used for character animation because defining characteristic components in biological motion is so challenging.

Despite it's difficulty, methods for mathematically defining biological motion are however being researched. In 2002, Nikolaus Troje explored the idea of using linear combinations of sinusoidal basis functions to synthesize human walking motion. In his study, psychophysical experiments of gender recognition were conducted based on a set of holistic motion capture data. The results were then sent through Fourier Analysis and Principal Component Analysis (PCA) to extract the male and female characteristics from the motion capture data [25]. By applying multilinear data analysis techniques to the extracted data this study created a model for mathematically describing character motion. Similar methods have been applied in later studies of

gait patterns associated with sadness and depression [17] and human female fertility cycle [19]. Troje's method [14] forms the basis for the animation authoring technique described in this thesis, however it is adapted in order to be applied to quadrupedal figures.

2.2 Web-based Animation Authoring Systems

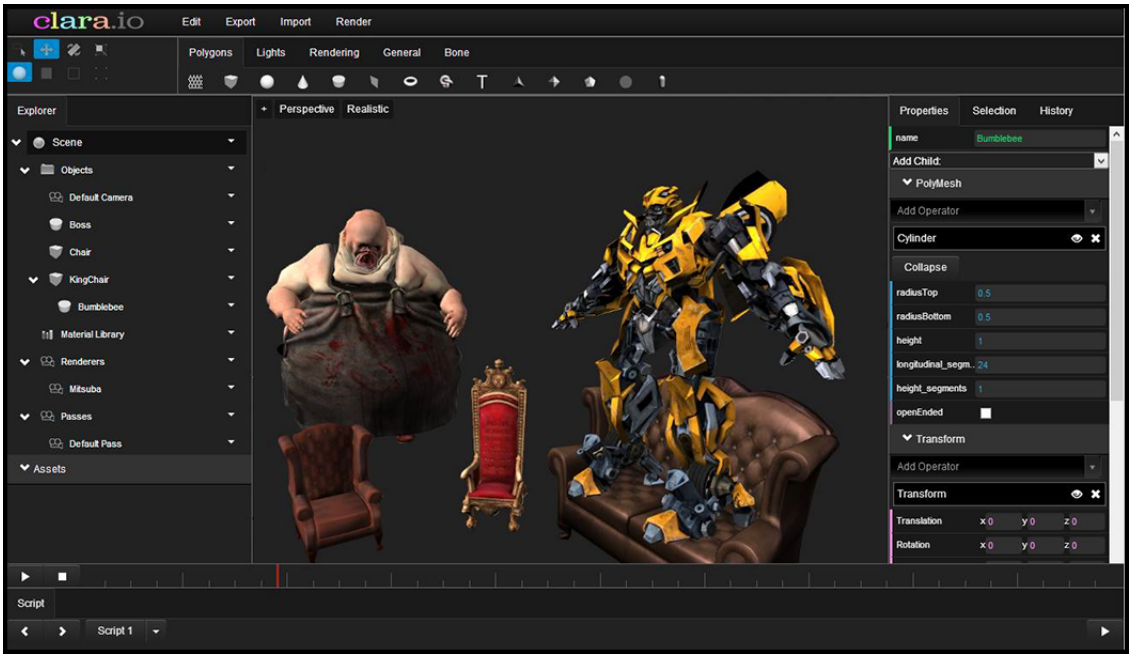


Figure 2.1: The Clara.io Interface [8]

2.2.1 Clara.io

The beta version of a new tool called Clara.io [5], shown in Figure 2.1, was demonstrated and well-received at the Special Interest Group on Computer Graphics and Interactive Techniques, commonly referred to as SIGGRAPH, convention in 2013 [8]. This package is very similar to Autodesk Maya, the current industry standard, in

that it provides tools for several areas of the production pipeline, including modeling, animation, and rendering. It is built using WebGL and aspires to provide a similar user-experience to a desktop-based tool, but with added benefits. These benefits include the fact that storage of data is handled in the cloud rather than local disk space, and the tool requires no installation, configuration or manual upgrading. Because this system provides tools for the animation pipeline all the way through rendering, it is not necessary for the user to export their work to local software, however the ability to export scenes to multiple 3D formats is offered. In Clara.io, animation is created via key-framing and therefore excludes the capacity to make use of motion capture and procedural methods.

2.2.2 Mixamo

Another online tool intended to speed up 3D production by aiding in the creation of animation is Mixamo [19], presented in Figure 2.2. The Mixamo website offers the ability to upload or customize your own character, send it through the Mixamo Auto-Rigger, and apply motion capture animation. In contrast to Clara.io, this tool is not meant to be used through to entire production pipeline. Once the user has completed their character, they will have to export it to be used within their preferred pipeline. Although Mixamo offers a large selection of options when building characters, as well as a multitude of pre-existing animations to choose from, the user is limited to only motion capture animation and bipedal characters. There are no options for creating key-frame or procedural animation, and currently no way of developing a quadrupedal character.

2.3 Perception Based Animation

Spencer Cureton's thesis implements a method similar to Troje's to describe the movement of quadrupedal animations. This solution is beneficial because it

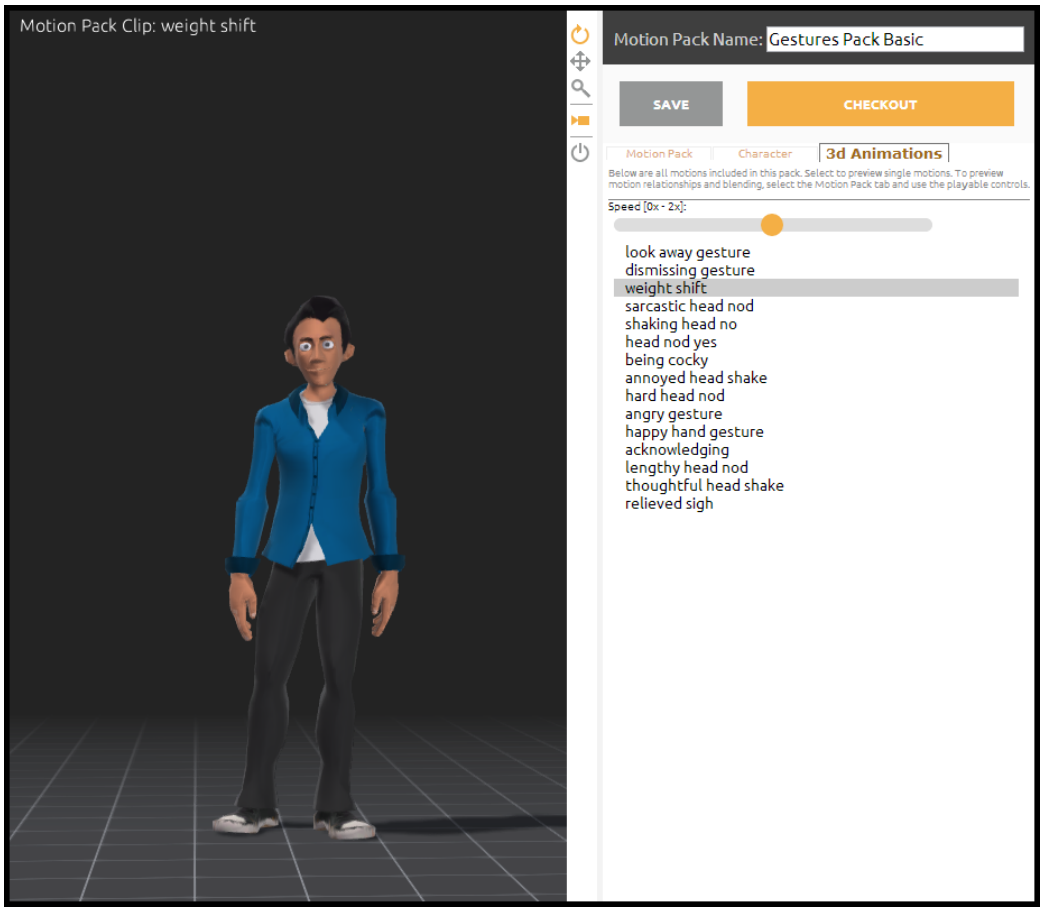


Figure 2.2: The Mixamo Interface [19]

avoids the pitfalls of key-frame animation and motion capture that come along with animating four-legged animals. Whereas Troje utilized motion capture data to gather motion examples, this method extracts an animal's motion from video footage. The joints of each leg were tracked throughout the film clip then the rotations between each of the joints were recorded for each frame. Fourier analysis on these rotation values was used to produce a sinusoidal function describing the animal's motion.

Junze Zhou's thesis [29] built upon Cureton's by isolating certain characteristics within the animal's motion to be used as animation controls themselves. In his final system, the output gait is a result of how much the user wants the gait to display

a specific trait. For example, a slider is given a range from light to heavy and manipulating that slider will influence the gait to portray the animal as lighter or heavier.

In this thesis, these works serve as a foundation for it's animation authoring system. Using a new approach, the system provides intuitive control over the character's form as well as it's perceived characteristics. As a web-based tool, this new application also shares benefits of the Clara.io and Mixamo systems. The methods mentioned in this section will be described in further detail in the next chapter.

3. METHODOLOGY

3.1 Animating Quadruped Gaits

When animating a quadruped it is important that the animator is very familiar with a few key aspects of their creature. These aspects have a large amount of influence on the believability of the final animation. They also need to be heavily considered during the construction of the animation authoring application. This section provides an overview of each area.

3.1.1 Gait Types and Footfall Patterns

The gait of an animal refers to the pattern of their limb movement during locomotion [7]. Both as a group and as individual animals, quadrupeds use a wide variety of footfall patterns. A quadruped's gait can loosely be categorized into one of three basic gait types: walk, trot, or run. More specific gaits can be found in-between these categories, however they will not be described here for the sake of generalization. An animal will select a gait based on a mixture of reasons, such as speed, environment, energy level, and health status. Ultimately, the most energy efficient gait will be chosen given the circumstances.

The walk is the slowest and least strenuous of all the gait types. It is a four-beat gait, meaning each foot strikes the ground individually. At any point within a single stride either two or three feet are on the ground. Each foot steps at generally the same speed as the others. The sequence starts with one of the hind legs, next comes then the front leg of the same side, then the opposite hind leg, and finally the opposite front leg. This gait is generally use for leisurely travel.

A trot is faster than a walk but slower than a run. It is a two-beat gait, meaning the animal's feet only strike the ground at two points during the stride. In this step

sequence, the front right limb moves in time with the left hind limb and vice versa. A quick trot has two points of suspension. Suspension is a moment during the stride in which all four feet are off the ground. These suspensions occur between the points at which one set of diagonal feet is supporting the animal. Suspension never occurs within a slower trot. This gait is often used to travel long distances with a reasonable amount of speed.

There are two different types of runs: the transverse gallop and the rotary gallop. A transverse gallop is the fastest gait that large ungulates use. Smaller ungulates, digitigrade animals, and some plantigrade animals can also use this gait, but the rotary gallop is their fastest option. The difference between unguligrade, digitigrade, and plantigrade will be discussed in the section 3.1.3.

The transverse gallop is a four-beat gait that includes one suspension. In this gait, the footfall pattern starts with one of the hind legs and is followed by the opposite hind leg. Both front legs come after, starting with the same side as the first hind leg. The front leg that supports the body just before the suspension phase is known as the leading leg.

The rotary gallop is the fastest of all the gaits, and also the most energy expensive. It is a four-beat gait as well, however, it contains two suspensions. As the name implies, the step sequence is rotational. For example, if the right hind leg is first, the left hind leg follows, then the left front leg, and finally the right front. The first suspension follows the hind legs and the second comes after the front legs are lifted. Just as it is in a transverse gallop, the leading leg is the second front limb to leave the ground. Figure 3.1 depicts the step sequence of each of the gaits discussed in this section.

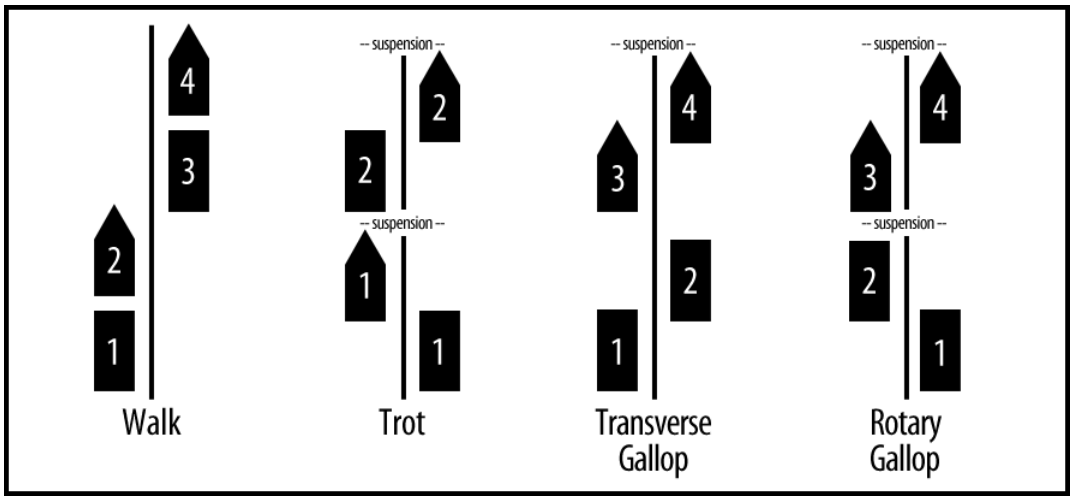


Figure 3.1: Footfall Pattern Diagram

3.1.2 Cycle Lengths

In animation a cycle refers to a series of frames that can loop continuously, resulting in an animation that can play for any length of time. A walk cycle is an extremely common example. Animators often produce walk cycles for a character before anything else. In this example, one cycle describes the motion for a full stride. This fact holds true for both bipedal and quadrupedal characters.

The number of frames in a cycle is known as the *cycle length*. This thesis also refers to this number as the *gait speed*. It is important for the animation authoring system that interpolating between motion examples can be done with gait samples that have differing gait speeds. Another issue to consider when blending multiple samples is that a cycle length may become a non-integer value. Because the animation is generated in real-time within the system, a non-integer cycle length will not cause any issues. However, should the final animation have a non-integer cycle length at the time of export the user will be given the option to modify the gait speed slightly for the purpose of looping.

3.1.3 Bone Structure

Quadrupeds can be classified into three categories: unguligrade, digitigrade, and plantigrade. Unguligrade animals walk on the tips of their toes and typically have hooves. Horses, deer, and cattle are examples of this class. Digitigrade quadrupeds walk on their digits, or toes. Dogs and cats fall under this category. Finally, plantigrade animals walk with their feet, podials and metatarsals, flat on the ground like humans. Two examples of this type of locomotion are bears and mice. Figure 3.2 depicts each of the three bone structures.

Although the application will eventually be expanded to include all three classes of quadrupeds, this thesis focuses on unguligrade locomotion. Animation is created for each of the four legs, the head, and the spine.

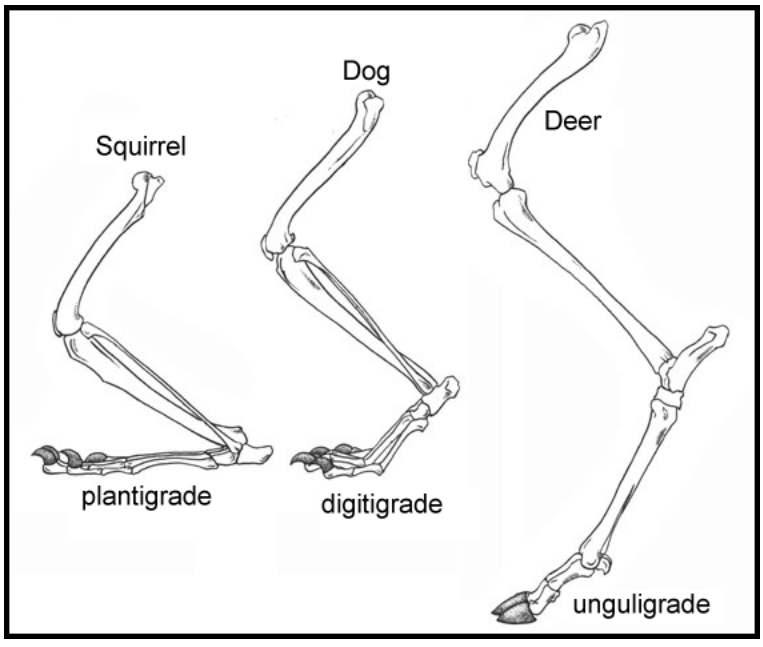


Figure 3.2: The Three Types of Quadrupedal Leg Anatomies [20]

3.2 Expressive Gaits

This section describes the process of preparing the motion samples used to define specific characteristics within an animal's gait.

3.2.1 Extracting From Video Footage

As mentioned in the last chapter, the motion examples used in this system were originally extracted from video footage. This process began with collecting a number of film clips showing various animals walking, trotting, or running. These clips had to meet specific qualifications in order to have the potential of providing usable data. The animal must be moving perpendicular to the camera as a side view supplies the maximum motion information for quadrupedal animals. Also, three full strides were required to ensure a full, clean cycle was recorded.

After the footage was collected, each video was rotoscoped. The process of rotoscoping consists of an animator tracing over footage, frame by frame, for use in a film. In this project, the position of each joint was manually tracked throughout the clip's duration. Fortunately, because the front limbs and hind limbs generally share the same pattern during a cycle, only one side of the animal needs to be extracted. A time shift can later be applied to the legs on the opposite side of the body to create the proper footfall pattern. Unfortunately, the usage of 2D video footage to attain motion data limits the final animation to the same two dimensions; however, should data for a third dimension become available, through a process like motion capture, the system can be expanded to execute it's transformations without any changes in technique.

The information taken from these videos serves two purposes: creation of a point-light display representation and extraction of joint rotations. The point-light display is used for biological analysis, whereas, the joint rotations are used during the pro-

cedural generation of new gaits. Just as in Troje's method, the rotations are sent through Fourier analysis to create a sinusoidal function describing their values over time. The function's components, amplitudes and phases, are stored in an online database along with other gait parameters such as gait speed and timing shifts.

3.2.2 Point-Light Display (PLD)

Point-light displays (PLDs) are commonly used as the stimuli for biological motion experiments because they isolate the motion of a subject from it's form. This allows for investigation of how characteristics are portrayed. A PLD is composed of moving dots which reflect the position of the key joints in the subject and was invented by Gunnar Johansson in the early 1970's [12]. This technique has been used in research to identify human gender from walking [13] and to test if the human brain might have detectors tuned to the characteristic signal generated by the feet of an animal in locomotion [27]. Figure 3.3 shows an example of a PLD of a trotting lion next to the full image it was created from.

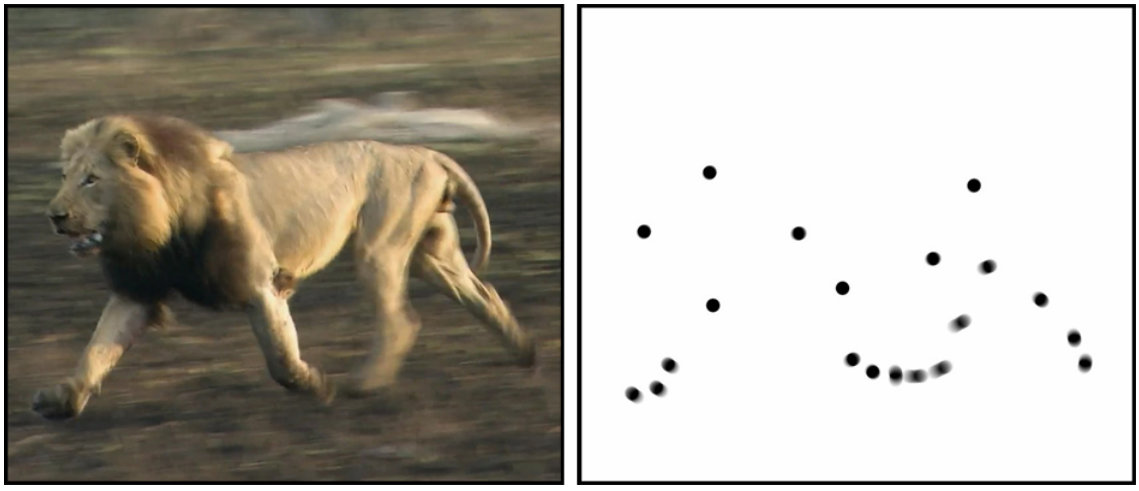


Figure 3.3: Full Image Versus PLD of Trotting Lion Footage

In 2009, Tim McLaughlin and Ann McNamara conducted a study to show that animal motion can communicate relative age, relative weight, and the identity as predator or prey [16]. In this study, a set of animal motions were shown to participants in their full footage and PLD form. Each clip was given a rating for each characteristic in the experiment. For example, the extreme cases for the age characteristic were young and old. The rating was chosen based on where the participant felt the motion sample belonged within the spectrum. Results from this study are used to decide which motion examples will represent each characteristic in the final system.

3.2.3 Translational Motion

The previous works discussed thus far focused solely on rotational movement of a quadruped's leg during locomotion; however, this is not the only type of motion that occurs during an animal's gait. The animal as a whole moves up and down during a gait cycle. This is especially apparent during suspension phases. The forequarters and hindquarters of the quadruped also translate relative to the central mass of the body each time they take a step.

Although the process previously described provides the position of these regions for each frame, an additional mechanism needs to be added to the process in order to differentiate the translation of the forequarters and hindquarters from the overall translation of the animal as it moves in space. This additional data enables the system to generate motion for the spine and head along with the legs. This development increases the believability of the final animation, as well as, the system's ability to portray desired characteristics.

3.3 Animation Authoring

This section investigates what is required to proceduralize motion based upon what is perceived and how control over those features is given to the author.

3.3.1 Procedural Gait Generation

The overall purpose of this system is to author animation for quadrupedal characters. The process of generating transformations for each of the character's joints on a time basis is the backbone of the entire application. The code responsible for this work is referred to as the *gait generator*.

The last section gave an overview of how motion samples for the system are developed. It is the job of the administrator of this tool to populate an online database with the resulting sample data. The first thing that the gait generator needs to do is request the sample data from the server and store it within it's own memory for quick access.

When the user begins playback, the gait generator plugs all the necessary values into a specific sinusoidal function. The current time along with the gait parameters will produce the transformation, whether it be a rotational or translational value, for one of the joints in the rig. The resulting value is used to update the joint's orientation or position. This calculation is done for each joint on every frame. For animation purposes, the playback speed is set to 24 frames per second.

Age and weight are the two characteristics demonstrated in this thesis. Through a previous experiment, the motion samples have already been given a rating of how much each characteristic is perceived within the sample [16]. The user is given control over characteristics through a method similar to the weighted interpolation developed by Nikolaus Troje in 2002 [25]. Sliders are provided to the user to change the influence of each characteristic on the animation. The value of a characteristic slider informs

the gait generator of how much weight a certain motion sample, one representing the chosen trait, has on the final transformation value. The gait type parameter uses this same method to interpolate between walk, trot, and run motion samples. The linear transaction between different gait types resembles a gait transition in the real world.

A final issue to consider is the interdependent relationship that speed shares with gait type and characteristics. For example, a light creature tends to move faster than a heavy one. Also, speed changes are directly linked to transitions in gait type. For this reason, disagreements can arise from having separate controls over each parameter. Desired traits may cancel each other out. A solution to this issue is to provide an option for the user to manually define the speed of the animal so that the characteristics no longer influence speed. Otherwise, the speed is determine based upon the chosen characteristics.

3.3.2 Flexible Rig

Standard rigs normally consist of a motion system, a control system, and a deformation system [21]. The motion system is responsible for defining points of articulation within a character and consists of a hierarchical set of "joints" or "bones". The control system provides an interface for the animator to manipulate the motion system and pose the character. Finally, the deformation system defines how the joints within a character's rig influence it's form.

Because this system enables the user to create countless varieties of character configurations, it is difficult to provide a geometrical form to represent every possibility. For this reason, the deformation system is not included in this thesis, however, one can be put in place once the final animation has been exported to a local 3D animation package. The rig instead utilizes primitive shapes to represent the character's

form.

For the purposes of this project, the rigs has two functions: it gives the user the ability to visualize the animation they are creating and it changes shapes in order to best represent the target character. The hierarchy of the rig is imperative to be able to complete these two functions without interfering with each other. Multiple layers will need to be created within the hierarchy to receive transformations from the gait generator and define the joint positions.

3.3.3 Graphical User Interface (GUI)

An intuitive graphical user interface (GUI) is crucial for enabling the user to interact with the system. The GUI is located next to the view port and houses all of the instruments available to the user for animation authoring. It is made up of three parts: the shape control panel, the gait control panel, and the rig control panel.

It is essential that this system can provide for many various quadrupedal configurations. The shape control panel addresses this issue. It provides the tools needed to manipulate the rig's form. Ideally, the user will have the option of choosing a shape from a set of predefined shapes or creating their own. If the user decides to design their own configuration, they should be able to upload a reference image to inform their process. After creating a custom shape, user should be able to save it's description to their own library and load previously stored shapes as well.

The gait control panel is the most important panel for creating the desired movement for the character. It provides controls for the gait's type, speed, and characteristics. This is where the function that generates the animation receives input. The most intuitive way for the user to control these values is through sliders. For example, the two characteristics presented in this thesis are age and weight. The weight slider ranges from one extreme case to the other with light on the left end

and heavy on the right.

Finally, the rig control panel provides the user with options for controlling the size and visibility of the rig. These options do not have any influence on the final animation and are meant to help the user analyze their work. For example, the animator may need to view the animation as a PLD or isolate specific limbs for further investigation.

3.4 Web Accessibility

The most unique feature of this application is that it is web-based. This section discusses how this will be achieved and also highlights the key benefits that are associated with tools being accessible online.

3.4.1 Achieving Web Accessibility

After the development of the animation authoring system is completed locally, it is compiled into one small file that is stored online. A web page is created through the same general process as most other web pages. HTML is used to set up the structure of the page, and CSS defines it's style. The key factor in this method is that the compiled system file is embedded into the web page. Upon loading the page, the web browser executes the file which initiates the animation system.

In order for the system to author animation it needs to have access to previously prepared motion samples. This data is stored in an online database so that it is available to the application at any time. Web scripts are also stored online for the purpose of transporting data back and forth between the application and the database.

3.4.2 Benefits of Being Web-Based

3.4.2.1 Lightweight Tool

As a web-based tool this system is inherently lightweight. Although a small plug-in is required in order for the browser to run the application, there are no large downloads or any installation needed to use it. No additional maintenance is required. Web based systems only need to be installed on the server placing minimal requirements on the end user. This makes maintaining and updating the system much simpler because it can all be done on the server. Any updates can be deployed via the web server with relative ease and fall completely under the administrator's responsibility. Simply accessing this tool's website ensures the user is using the most current version. This tool can run on any platform with the exception of Linux and mobile platforms, and is compatible most web browsers. It is also completely open to the public. There are no licenses associate with the application.

3.4.2.2 Online Storage

Everything needed to use this tool is stored online. This fact is extremely significant. Because the tool is readily available wherever an internet connection can be made the user is not confined to a specific workspace. This means a user can work from virtually anywhere on almost any computer. Also, this tool does not take up any space on the user's hard drive. User accounts are not within the scope of this thesis, however their concept adds even more potential perks for a web-based tool. Given that the user can create an account they could produce their own personal library of animations and custom shapes that would be preserved even after a session is closed. The user could log back into their account and start working on a previous animation by loading the gait settings, shapes, and even reference images with a click of a button.

3.4.2.3 Remote Collaboration

This tool can be accessed from anywhere an internet connection is available. This opens a large number of opportunities for collaboration between multiple people whether they work in the same building or across the world from each other. The method for exporting animation provides a text file with all the settings needed to recreate a gait cycle. These files can easily be shared between collaborators in order to perfect an animation together. As previously mentioned, user accounts can stretch the existing benefits of a web-based tool even further. A user could share items from their personal library with other users identified as a colleague or teammate. To continue with this idea, users could add their animations or custom shapes into an open database accessible to all other users of the tool for everyone's benefit.

3.5 Local Software Compatibility

The benefits of this tool's accessibility on the web are great, unfortunately, unless the full production pipeline is online, it is necessary to be able to bring the final animation into a local 3D animation package. This requires a process of transporting the animation data between tools.

An export entails formatting data in such a way that it can be used by another application. When a user has completed a gait cycle, this system gathers all the current animation settings and writes them to a text file. At this point, the file exists on the web server, but a download will automatically initiate to move the file to the user's hard drive. Unlike some other formats, a text file can be opened and read in a text editor. This can help the user recreate a gait in the web application if the user so desires.

The import process is the opposite of an export. It allows for data produced by another application to be used within the current application. In this case, the

current application is a local animation package. A script will allow the package to locate the animation file and use it's contents to build and animate a rig. The configuration and movement of this rig is an exact copy of what the animator created inside the web application except for one feature. This rig includes additional animation controls for each joint. These controls allow the animator to layer supplemental movement on top of the authored gait cycle.

4. IMPLEMENTATION

This chapter will outline the system composition. Each building block will be explained including the techniques and technologies used. Figure 4.1 visualizes how these building blocks work together.

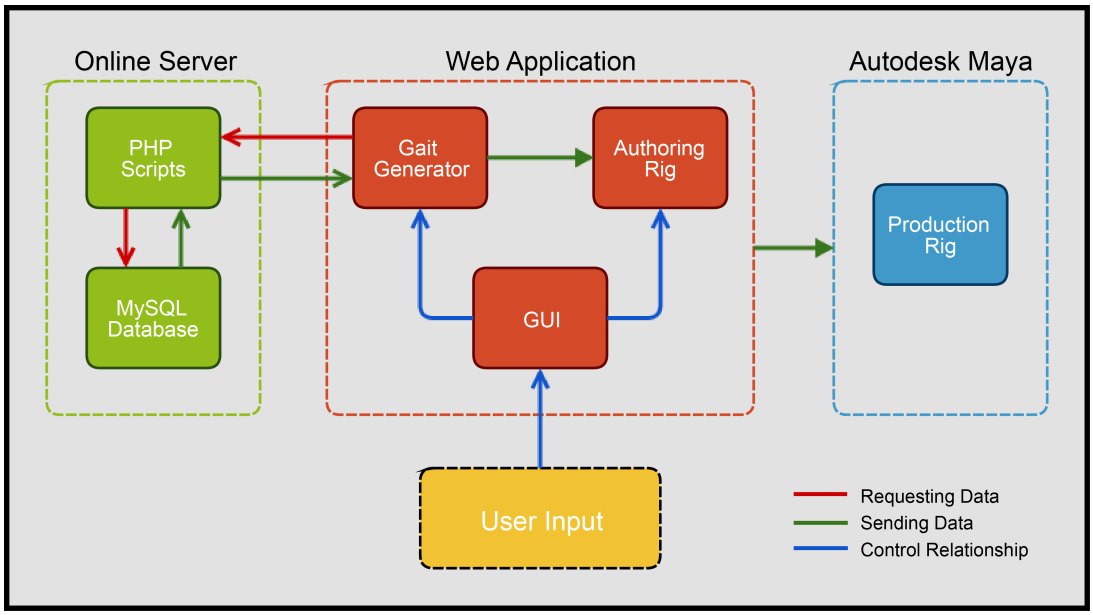


Figure 4.1: The System Composition

4.1 Perceptible Characteristics

The ability of this system to portray specific characteristics comes from the collection and conditioning of sample motion data that has been proven to strongly display those traits. This section outlines the process of obtaining gait samples.

4.1.1 Producing Translational Data

In order to extract an animal's motion from video footage this thesis uses a process of rotoscoping. The video sequence is brought into Autodesk Maya as an image plane and viewed through an orthographic camera. The position of each joint is manually track for each frame in the clip. Previous implementations of this method stopped once the position of all joints were correctly marked throughout the clip because it provides enough information to extract the orientation of each joint; however, this stops short of what is needed to obtain the translational movement of the animal.

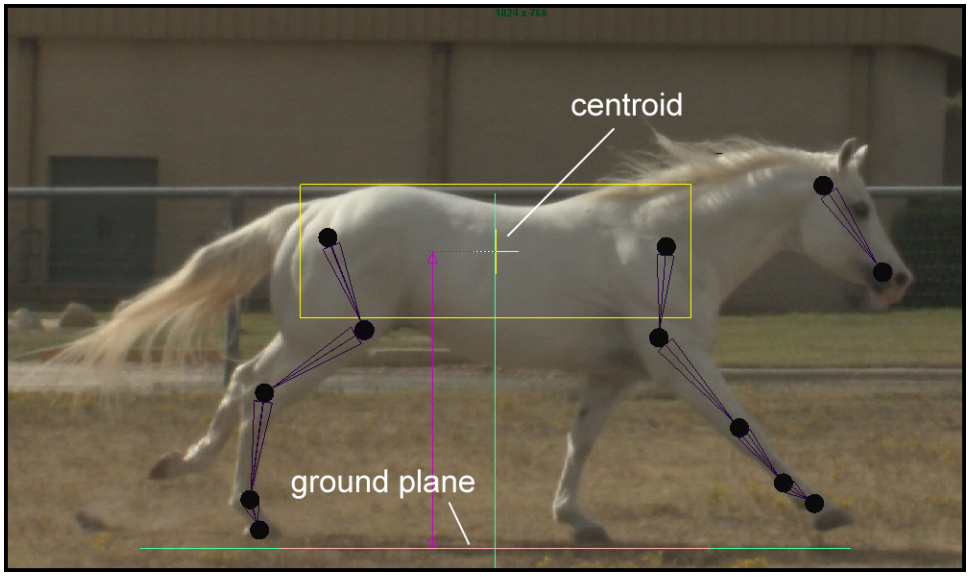


Figure 4.2: Mechanism For Extracting Translational Data

Figure 4.2 shows the mechanism that was constructed to differentiate the translation of the forequarters and hindquarters relative to the centroid of the animal. In this thesis, the centroid refers to a point found in between the forequarters and hindquarters of an animal which moves directly with the animal's center of mass.

The centroid is tracked, just as the joint positions, on each frame using the yellow box as a guide. The tool also provides a method for defining where the ground plane is. This is important because the position of the centroid should be recorded relative to the ground plane. For example, the centroid may translate upwards within the video because the animal is walking up a hill or because the animal is in the suspension phase of a gait cycle. The distance between the ground plane and the centroid is the value that is recorded for gait generation.

Using the centroid's position as a marker for the animal's motion as a whole, the relative translation of each quarter region can be determined. The addition of the spine and head are also made possible with this piece of data.

4.1.2 Motion Data Conditioning

The process of conditioning the motion data, whether it is rotational or translational, is the same. A Python script is executed inside Maya's Script Editor that extracts the rotations, and translations if applicable, of each joint of the front and hind legs, as well as the centroid. The values are graphed and analyzed for the purpose of choosing the specific set that best represents one complete gait cycle. Once a set has been selected for each body part, the data is ready for Fourier analysis. Another Python script is responsible for this step. Preset variables within the script are set equal to the chosen transformational values, along with the cycle length, time shift between the left and right sides of the animal, and initial bias of the joint's position. This process is executed for each body part. The result is a sinusoidal function for each joint describing it's motion throughout the cycle. This data is stored in the online database to be access by the gait generator.

4.2 Application Development

4.2.1 Constructing the System

The majority of the construction of this system was done in Unity 3D. Unity 3D [2] is a game engine popular with both professional and casual game makers because of its intuitive interface [3]. In recent years, the makers of Unity 3D have released the Unity Web Player plug-in allowing users to build applications and make them available online. This is accomplished by compiling the project within Unity 3D and embedding the build onto a web page. Virtually every operation of the animation authoring system is defined within the Unity 3D build except for online scripting and storage.

For this project, setting up the scene within Unity 3D is very similar to setting up a shot for a 3D animation. Cameras, lights, and backdrops are placed according to the character's position; however, the most important element created is the character animation rig. The hierarchy of the rig is crucial because the rig must be able to receive input from the gait generator as well as be able to change shape according to presets or user input. It consists of six joint chains: one for each leg, one for the head, and one for the spine. Figure 4.3 shows an example of the three layers needed for one joint, in this case the front left elbow joint.

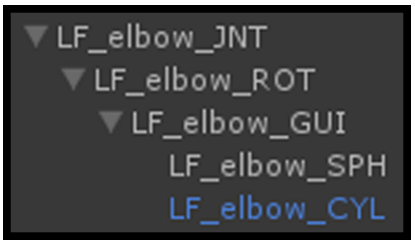


Figure 4.3: Rig Hierarchy Within Unity 3D

The top layer for a joint, or *JNT* layer, contains it's position relative to it's parent. This layer never receives any values from the gait generator, and is meant to be used only to allow updates to the rig's overall shape. The *ROT* layer is where the procedural animation influences the rig. This layer receives rotation values calculated by the gait generator. The first joint in each chain has an additional layer, the *POS* layer, to account for generated translations. Finally, the *GUI* layer is responsible for the display of the rig. The *SPH* object is the sphere used to represent the joint. It is colored based on which region of the body it belongs to. The left side has blue spheres, the right side has red spheres, and the center spheres are green. Not only are the spheres necessary to show joint positions, they can also be selected and moved to make new rig shapes. The *CYL* object is a piece of geometry that represents the joint orientation. This mesh is a grandchild of the *ROT* layer, and therefore, is oriented along with it. A child joint is positioned at the very end of it's parent's *CYL* mesh. These meshes move and scale according to the mouse during the creation of a new rig shape. The process of creating a custom shape is discussed in section 4.3.

Every event that occurs within the system during a session is a result of one of the system's scripts. The system contains 14 scripts, all of which are written in C#. The scripts vary in length and complexity, but their purpose is either to allow the user to interact with the system, communicate with the online database, or author animation to be placed on the rig. Although their names are not listed, the specific functionality of each script is described throughout this chapter.

4.2.2 Applying Motion to the Rig

In this system the final joint rotations and translations defining the gait cycle are computed by one very important C# script. The *compute* function within this script is commonly referred to as the gait generator. Upon initialization of the

application, the script requests all the necessary gait sample data from the online motion database. This data consists of a series of components required to form the sinusoidal function used to compute joint transformations, such as amplitudes and phases [29]. These values are stored in several arrays for quick access throughout the use of the application so that the database only needs to be queried once.

Once the values from the database have been stored, the gait generator queries the gait type, speed multiplier, user-controlled gait speed switch, and characteristic settings from the *Gait* tab. These parameters have been set by the user. The gait speed refers to how many frames it takes for the creature to complete one full gait cycle. If the user-controlled gait speed switch is enabled, then the gait speed is set directly to the value the user specified. If the switch is turned off, the gait speed will be determined by interpolating between the gait speeds of the motion data samples from the database according to the characteristic values. For example, the gait speed of the sample representing a heavy walk will be weighted much higher if the *Weight* slider is set closer to the heavy side. The resulting gait speed is then adjusted by the speed multiplier value.

The system extracts a gait sample from the motion database to represent each of the two extremes for both characteristics, age and weight, for each of the three gait types. This results in 12 motion examples overall. Using the computed gait speed, along with the side shift, amplitude, and phase values previously stored in arrays, the gait generator computes the rotation and/or translation of each joint for each of the 12 gait samples for the current frame. This is done by plugging the variables into a specific sinusoidal function. Now that the desired transformation value has been calculated for all samples, another interpolation process is used to determine the final output value. Similarly to the gait speed interpolation, the current gait type and characteristic settings are used to blend the transformation values. Figure

4.4 outlines the interpolation process.

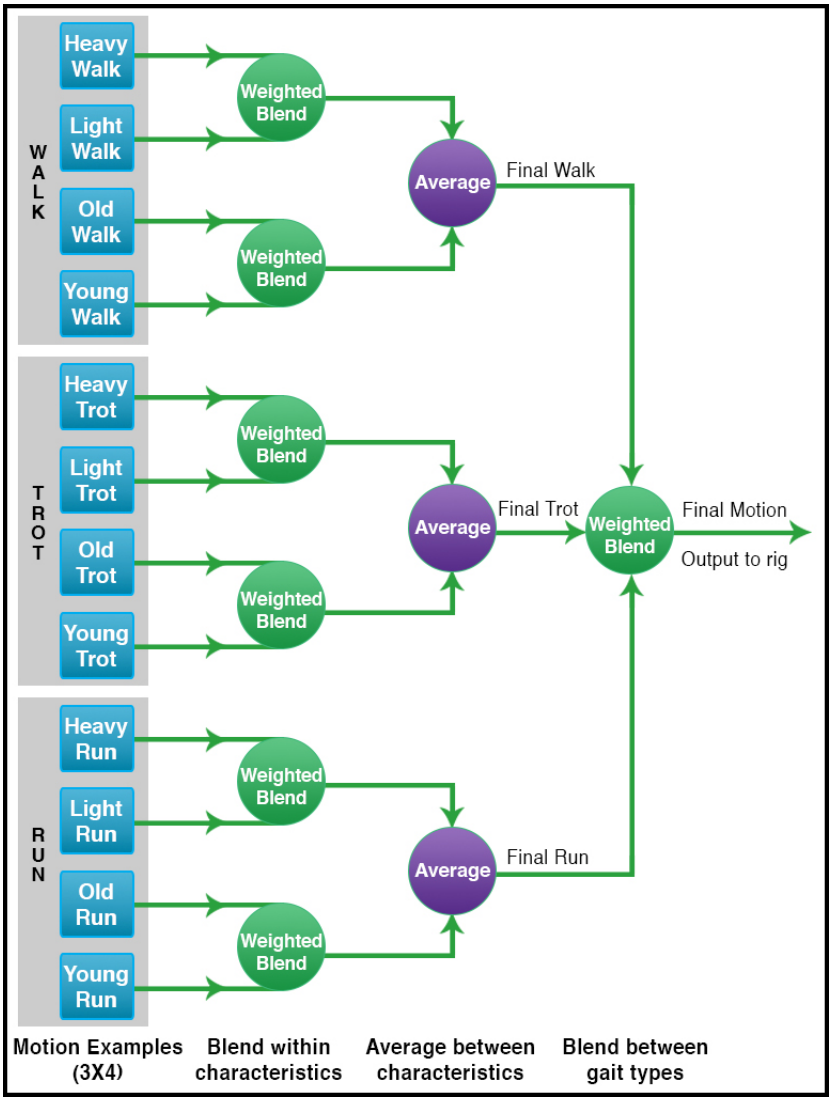


Figure 4.4: The Main Interpolation Process [29]

It is important to note that because Unity 3D is a game engine, it attempts to run at the highest frame rate possible, generally around 60 frames per second (FPS). This rate is higher than the rate that is typically found in the video clips used to

generate motion samples. The conditioning process responsible for extracting motion data assumes a frame rate of 24 FPS. Another C# script forces the application to maintain the desired rate of 24 FPS.

4.3 Controlling the Animation

The Graphical User Interface (GUI) is what allows the user to interact with the system. It is made up of three panels: Shape Control Panel, Gait Control Panel, and Rig Control Panel. These panels will be the topic of this section.

4.3.1 Shape Control Panel

The ability to change the shape of the rig is necessary for this application to be used for a wide variety of quadrupedal characters. The Shape Control Panel consists of two tabs: Presets and Custom.

4.3.1.1 Preset Shapes

To enable the animator to create an animation as quickly as possible, this system provides a set of preset shapes for the rig. These shapes include a horse, zebra, lion, cheetah, tiger, and leopard, as shown in Figure 4.5. These buttons are great because they allow the animator to switch the rig into a common shape without interacting with the rig themselves. When a new preset is selected, all the current rotations on the rig are set back to zero and the joints are translated according to the preset definition. These shapes can also be used as a basis to start creating a custom shape.

4.3.1.2 Custom Shapes

The rig within the application is built in a way that it can be transformed and continue to display generated animation. In order to change the shape of the rig the *Custom* tab must be active. This restriction is made to prevent any accidental joint shifts while working on the gait cycle. In addition to enabling joint updates, entering

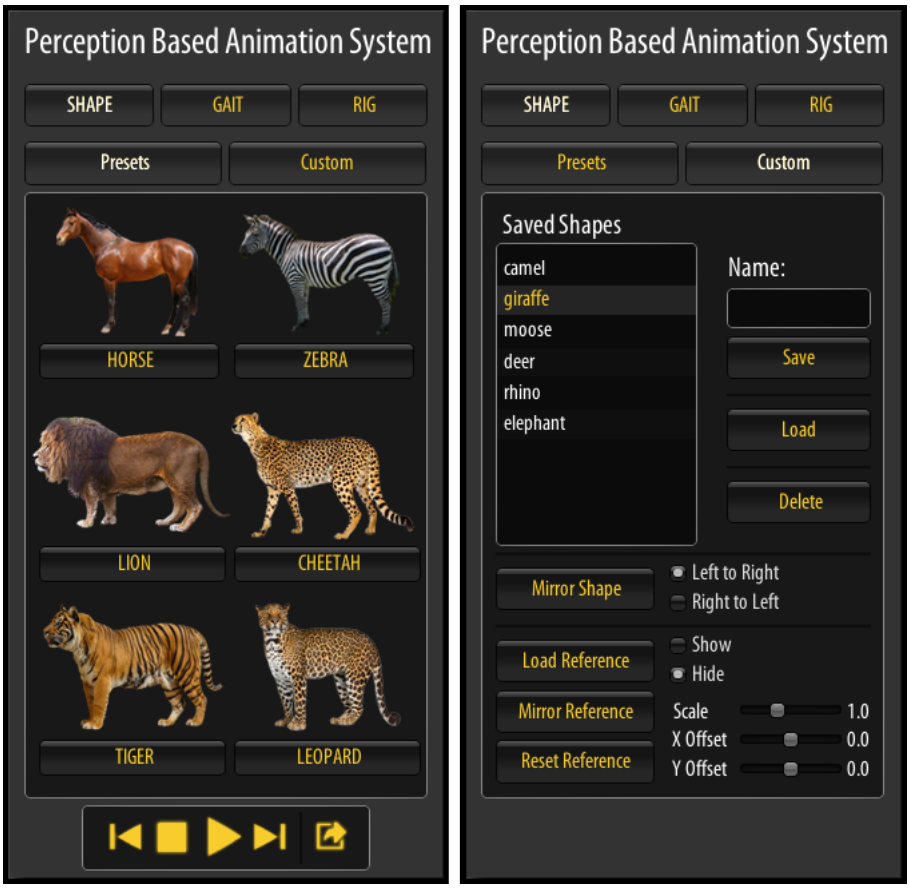


Figure 4.5: The Shape Control Panel

the *Custom* tab also stops playback. This returns the rig to it's neutral pose where there are no rotations on any of the joints. At this point the rig's shape is ready to be adjusted. Similar to other 3D software packages, the user can select a single joint by clicking on it or grab a group of joints by dragging the mouse over a region. The user can then drag the selection to a new position using the mouse. Each time a joint is moved the new offsets are accounted for by updating the joint's position and removing translations from the joint's children. The majority of quadrupedal characters are symmetrical, therefore a *Mirror Shape* button is provided to mirror the joint positions from one side of the rig to the other. However, joint shapes are

not required to be symmetrical.

Once the user is finished creating their new shape they can name it and save it to their library. Clicking the *Save* button gathers all the joint positions and sends them to the online database, as well as, adds the name to the current library. The library is a list of all the shapes saved during the session. If a user wants to return to a previously saved shape, they can select it from the library and click the *Load* button. The system will query the online database for the shape's definition and update the rig accordingly. A shape that is no longer needed can be removed from the library using the *Delete* button. A more in depth description of passing data back and forth to the online database will be provided in section 4.4.

4.3.1.3 Using a Reference Image

Using an image as reference when creating a custom shape may be helpful to ensure the proportions are correct. The *Load Reference* button allows the user to choose an image from their computer or online to be displayed in the system. In this application, the creature must be facing the left side of the screen. The *Mirror Reference* button is available to flip the image if needed. The scale and offset sliders are provided to help center the reference shape on the screen to account for the position of the creature in the image, and the *Reset Reference* button will set them back to their initial settings. The image will only be visible in the side view as this is the most conducive view for creating quadrupedal shapes. Finally, the Show and Hide toggles enable the user change to the visibility of the image depending on whether the reference is still needed. Figure 4.6 shows an example of a user utilizing a reference image for shape creation.

Unfortunately, internet security issues prohibit the ability of the Unity Web Player to use an image directly from a user's hard drive. In order for an image

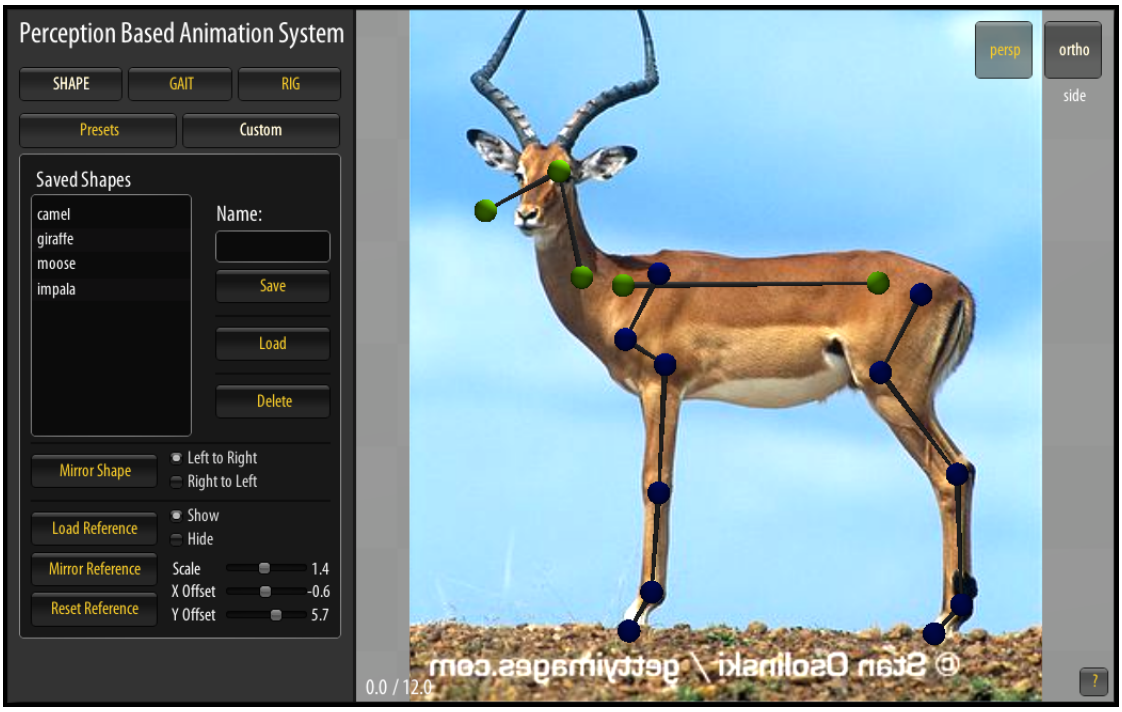


Figure 4.6: Displaying an Image for Reference

to be used by the system it must be placed on an online server. For the user's convenience this process is done for them. As mentioned before, clicking the *Load Reference* button enables the user to select an image to use in the system. Specifically, it notifies the web browser to open up the File Browser so that the user can choose either a.jpg or .png image file. PHP is used to upload that image to the server where it can then be used by the Web Player. This process is described further in section 4.4.

4.3.2 Gait Control Panel

The Gait Control Panel, presented in Figure 4.7, provides the instruments used to define the gait cycle. It consists of three sections: Gait, Speed, and Characteristics. The variables within each panel directly influence the gait generation process described in the last section.

Figure 4.7: The Gait Control Panel

The *Gait Type* slider is used to transition the gait between a walk, trot, and run. The range is set from 0 to 20, where walking results from a setting of 0, trotting from 10, and running from 20.

The Speed section includes a *Speed Multiplier* slider and a check box used to enable user-controlled gait speed. The *Speed Multiplier* ranges from 0 to 2 and can be used to increase or decrease the overall speed of the animation. The speed and characteristics of a gait are interdependent, therefore changing the value of a characteristic may result in an unwanted speed. Checking the previously mentioned check box results in three additional sliders. These sliders allow the user to define a

specific speed, in frames per cycle, for each of the three gait types. When this switch is enabled, the *Gait Type* and characteristic sliders will not influence the speed of the gait. Otherwise, the creature's speed is determined solely by those inputs.

Finally, the Characteristics section provides control over the application's two characteristics: Weight and Age. Just as the *Gait Type* slider transitions the animation between gait types, these sliders interpolate between the extremes of each characteristic. The *Weight* slider ranges from -50, representing a light gait, to 50, representing a heavy gait. The *Age* slider uses the same numbers to change between young and old.

Figure 4.8: The Rig Control Panel

4.3.3 Rig Control Panel

The settings found within the Rig Control Panel do not have any effect on the final animation. These options are provided for the user so that they can more easily observe their animation. The first set of options scale different aspects of the rig. Updating the size of the spheres, representing joints, and the bones may help when interpreting the motion. The user can also scale the entire rig larger or smaller to better fit the view point. The second set provides the ability to hide the spheres, bones, or even certain regions of the rig. For example, the animator may like to observe the animation in PLD form. To do this they can simply hide the bones. They can also isolate specific parts of the animation, such as the front left leg, to analyze its motion during a step. The Rig Control panel is shown in Figure 4.8.

4.4 Server Side Tools

A significant feature of this system is its accessibility on the web. There are several specific tools needed to make this happen. Those tools will be the topic of this section.

4.4.1 Web Development

In order to create a web-based application it is necessary to have access to an online server. Heroku is a cloud-based development and application-delivery platform [11] that provides the virtual server and other services used for this thesis. *Heroku: Up and Running* compares Heroku to the operating system (OS) on a computer. Just as a game developer can rely on a large amount of functionalities that the OS provides, a web developer can rely on Heroku to set up the server and keep it up to date [18]. This frees up the developer to focus on the construction of their own application. The specific usage of Heroku on this project can be divided into two

subject: PHP Communication and MySQL Database.

4.4.1.1 Online Communication

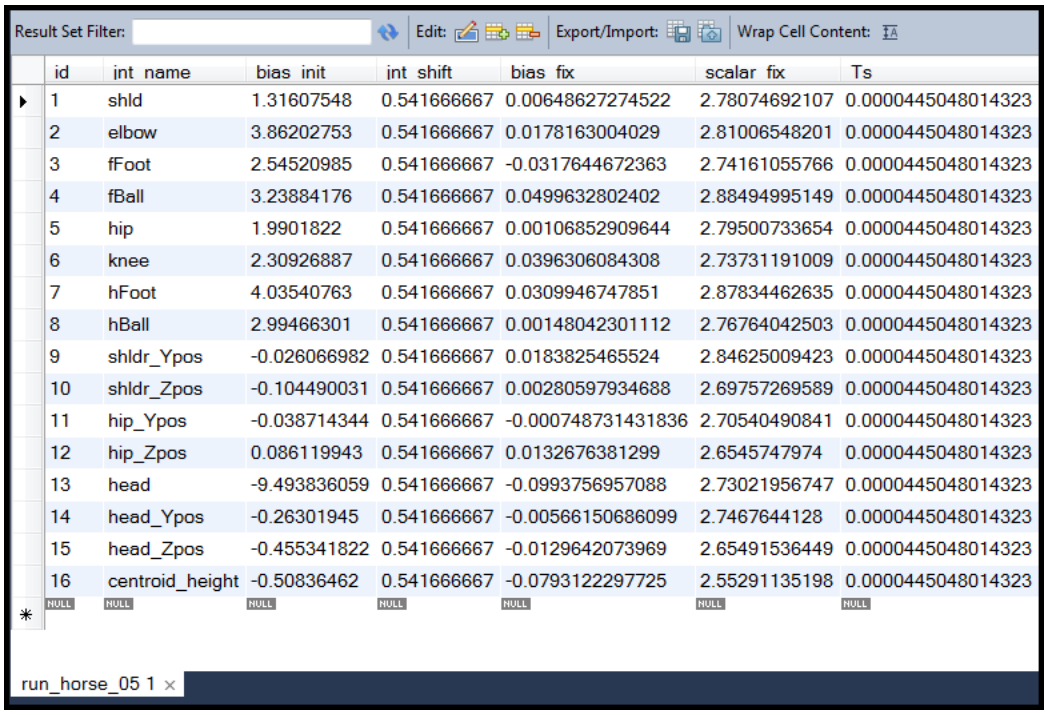
Without communication between the system and the server this project would not be possible, making PHP an indispensable factor. PHP is a popular server-side scripting language optimized for web development [10]. It provides the necessary passage of information between the application and stored data on the server. This means that it is involved in multiple processes throughout the application's use. PHP scripts are implemented at the system's initialization to load all the necessary motion data, when uploading an image to use as a reference, to save and load custom shapes, and to download an animation file. It is also used to view the current list of gait samples, as well as, add new gait samples to the database through the web browser.

These scripts are stored on the virtual server provided by Heroku and are called from inside the system and the web browser. Their processes can be generalized into three groups: extracting data from the database, adding data to the database, and writing/storing files on the server for other uses. The majority of the PHP scripts used in this thesis are simply a request from the system for specific data from the database. For example, at the start of the session the system needs to store all of the motion data values into memory for use in gait generation. It calls a PHP script, which runs a query against the database and sends the resulting values back to the system. Saving a custom shape is an example of adding new data to the database. The application sends all of the joint positions to the script, which plugs them into a MySQL insert statement, then calls that statement to add the new shape to the custom shapes table. The final script type is used for uploading to and downloading from the server. When the user decides to export their work, all the data is packaged together and sent to a PHP script. The script writes all the values into a text file

stored on the server, then another script is used to download it.

4.4.1.2 Motion Database

The method used for gait generation in this project requires a large amount of data. A MySQL database is used to store this data. Specifically, the ClearDB MySQL [6] add-on within Heroku was used to expedite the initial set up. Once the database was in place, MySQL Workbench [28] was used to visually design and manage it.



id	int_name	bias_init	int_shift	bias_fix	scalar_fix	Ts
1	shld	1.31607548	0.541666667	0.00648627274522	2.78074692107	0.0000445048014323
2	elbow	3.86202753	0.541666667	0.0178163004029	2.81006548201	0.0000445048014323
3	fFoot	2.54520985	0.541666667	-0.0317644672363	2.74161055766	0.0000445048014323
4	fBall	3.23884176	0.541666667	0.0499632802402	2.88494995149	0.0000445048014323
5	hip	1.9901822	0.541666667	0.00106852909644	2.79500733654	0.0000445048014323
6	knee	2.30926887	0.541666667	0.0396306084308	2.73731191009	0.0000445048014323
7	hFoot	4.03540763	0.541666667	0.0309946747851	2.87834462635	0.0000445048014323
8	hBall	2.99466301	0.541666667	0.00148042301112	2.76764042503	0.0000445048014323
9	shldr_Ypos	-0.026066982	0.541666667	0.0183825465524	2.84625009423	0.0000445048014323
10	shldr_Zpos	-0.104490031	0.541666667	0.00280597934688	2.69757269589	0.0000445048014323
11	hip_Ypos	-0.038714344	0.541666667	-0.000748731431836	2.70540490841	0.0000445048014323
12	hip_Zpos	0.086119943	0.541666667	0.0132676381299	2.6545747974	0.0000445048014323
13	head	-9.493836059	0.541666667	-0.0993756957088	2.73021956747	0.0000445048014323
14	head_Ypos	-0.26301945	0.541666667	-0.00566150686099	2.7467644128	0.0000445048014323
15	head_Zpos	-0.455341822	0.541666667	-0.0129642073969	2.65491536449	0.0000445048014323
16	centroid_height	-0.50836462	0.541666667	-0.0793122297725	2.55291135198	0.0000445048014323
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Figure 4.9: Horse Run 05 Joint Data Table

The database is made up of four table types: motion data, joint data, sinusoidal components, and custom shapes. There is only one motion data table. It houses one entry per gait sample and serves as a comprehensive list of motion samples to

pull from. The columns of this table consist of gait name, gait type, side shift, t shift, and footfall pattern. All of these values describe the motion as a whole. The next two tables hold values that apply to each joint individually. For organizational purposes, each gait sample needs two tables to store it's data: the joint data and sinusoidal components tables. The joint data table, shown in Figure 4.9, holds the variables that come from the conditioning process and describe each joint in relation to each other. The sinusoidal components table is by far the largest table and stores the amplitude and phase for each joint in the rig for all 250 components of the sine function. Automation of adding a new gait sample to the database will be discussed in section 4.4.3. Finally, the custom shapes table is used to store and load rig descriptions that the user creates within the application.

4.4.2 Browser and Application Communication

For the most part the Unity Web Player functions on it's own while occasionally sending and receiving data in the background hidden from the user, but there are two exceptions to this rule: downloading and uploading. In order for the download and upload dialogs to occur the web browser needs to be notified. JavaScript is the key to communication between the Unity Web Player and the web browser [1]. As mentioned in the previous section, a PHP script is responsible for writing the animation file; however, once the file has been written the application tells a JavaScript function on the web page to click the download link. In the case of uploading an image to use as reference, the communication flows both ways. The system sends a message to the browser to display the upload button which is done through another JavaScript function. Clicking this button allows the user to browse for and select their desired image. The web browser uploads the image to the server using another PHP script, but it also sends the system the filename so that it can use the uploaded

image.

4.4.3 Administrator Tools

Although the MySQL Workbench interface makes it much easier to interact with the database, it can still be time consuming to write an insert statement with the large amount of data needed for gait generation. In order to save time for the person in charge of maintaining the system, this project includes three additional web pages that collaborate with the database.

The first page allows an administrator to add a new gait sample to the database by simply uploading a motion data file. The motion data file is a text file which is the final product of the conditioning process. It contains every value needed to replicate the given gait. Once the administrator selects a motion data file, a PHP script parses through it and stores each of it's values. After all the values have been stored, the script uses MySQL statements to add the new gait sample to the motion data table as well as create and populate both the joint data and sinusoidal components table for the gait.

The motion data table is the best way to monitor the available gait samples. The second page displays the current state of the motion data table which enables the administrator to quickly view all available gait samples without having to use additional software.

Finally, the third page enables the administrator to update which samples represent the extremes of each characteristic. If these choices were hard-coded into the system's scripts, then the administrator would have to rebuild the system each time a characteristic needed to be updated. Instead, the application looks to the database at the start of the session to determine which samples it will use.

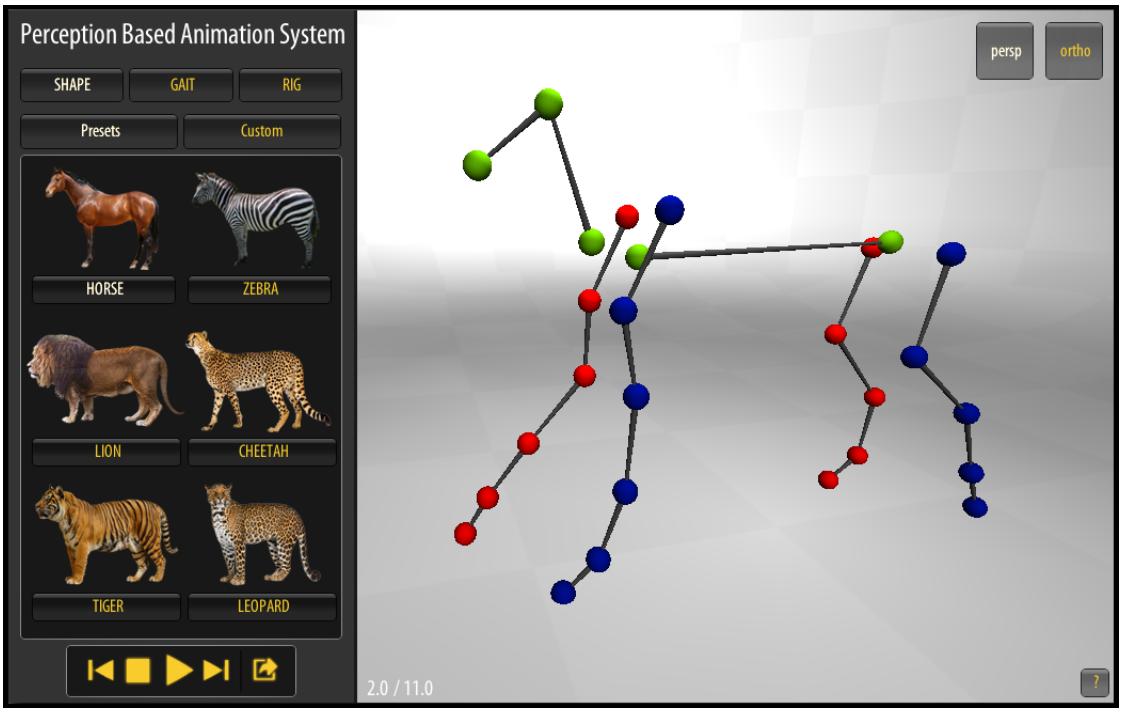


Figure 4.10: Completed Animation System

4.5 Using the Final Animation

The purpose of this tool is to allow an animator to quickly and intuitively create animation, but once this step is done the animation will need to be accessible in a production pipeline. This section discusses the process of transporting the animation from the web application to a local software package.

4.5.1 Exporting the Final Animation from the Web Application

When the user finishes creating a gait cycle, their work can be exported by clicking the Export button. As mentioned in the previous chapter, if the gait speed is currently not an integer value the user will be given the option to modify the animation slightly so that it will loop properly. If the user chooses to adjust the animation, the gait speed is rounded to the nearest integer value and the transformations are

recomputed; otherwise, no changes are made to the animation data. At this point the system gathers all the necessary data and sends it to a PHP script. The browser will automatically begin downloading the animation text file through the process discussed previously in section 4.4.2. This file can be stored locally wherever the user feels best. As the file is a simple text file, it can be opened and read in a text editor. The user can clearly see what the gait settings, joint positions, animation data, and reference images settings were set to at the time of export. A current minor limitation of the system is that these settings are not stored once the user exits the system, but reading the animation file can help to recreate gait cycle. The file can also be used to bring the animation into Autodesk Maya. An example animation file is show in Appendix A.

4.5.2 Importing the Final Animation into Autodesk Maya

For the user, importing an animation is made simple through the animation importer script provided on the web application's home page. The script is written in Python [9] and allows the animation to be brought into Autodesk Maya. It can be run directly through Maya's Script Editor or placed in Maya's script directory. When called, the script opens a dialog box, depicted in Figure 4.11, for the user to browse for their desired animation file. Clicking the *Import Animation* button begins a short preparation process.

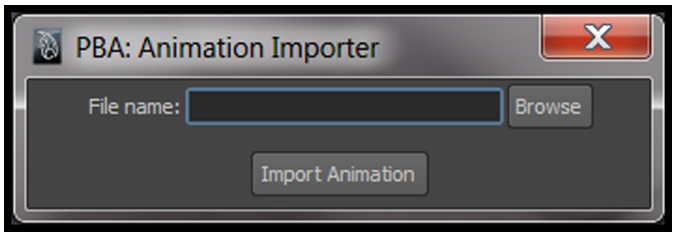


Figure 4.11: Autodesk Maya PBA: Animation Importer

The script begins by parsing through the file and store the skeletal and animation data. Once all the data has been stored, the next step is to use the skeletal data to construct a rig. Creating a rig with Python is a very popular approach because it can save a large amount of time and in the case of this project it ensures the animator will be provided with a rig that can correctly receive animation exported from the web application. As mentioned in chapter 3, standard rigs normally consist of a motion system, a control system, and a deformation system [21]. As this thesis focuses on skeletal motion it only includes the motion and control systems.

Constructing the motion system is the first step in building the rig. The system is made up of six chains of joints, one for each leg, one for the head, and one for the spine. The joints are set up in the correct hierarchy and positioned according to the values in the animation file. A spherical mesh is constrained to each joint purely for visual aid. These spheres are color coded by region of the body; the left side is blue, the right side is red, and the middle is green.

The control system is next. Each joint, with the exception of the joints at the end of the chains, has a spherical shape drawn around it to be used as a control. The are two methods of moving joints within a joint chain: Forward Kinematics (FK) and Inverse Kinematics (IK). Forward Kinematics refers to rotations on a set of joints that are applied down the chain. In this method only parent rotations effect child joints, whereas, in Inverse Kinematics the end of the chain is translated in space and the parent joint rotations are solved for. The rig in this project uses Forward Kinematics (FK). It is important to note that an extra layer is placed above each control within the hierarchy to account for the rotation and translations from the animation file.

Additional attributes are added to the rig to enable the user to update scale and visibility similar to the Rig Control Panel in the web application. Once the rig setup

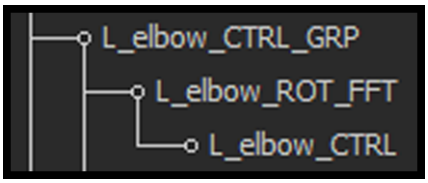


Figure 4.12: Rig Hierarchy Within Autodesk Maya

is complete, the script applies the animation data to the rig. This is accomplished by applying the rotations and translations to the extra layer above the each control. The layer structure can be seen in Figure 4.12. A key-frame is set on this layer for each frame in the gait cycle. At this point the animation is ready to be used in production or the FK controls cans be used to layer on addition animation to the rig. Figure 4.13 depicts the finished product of the import process.

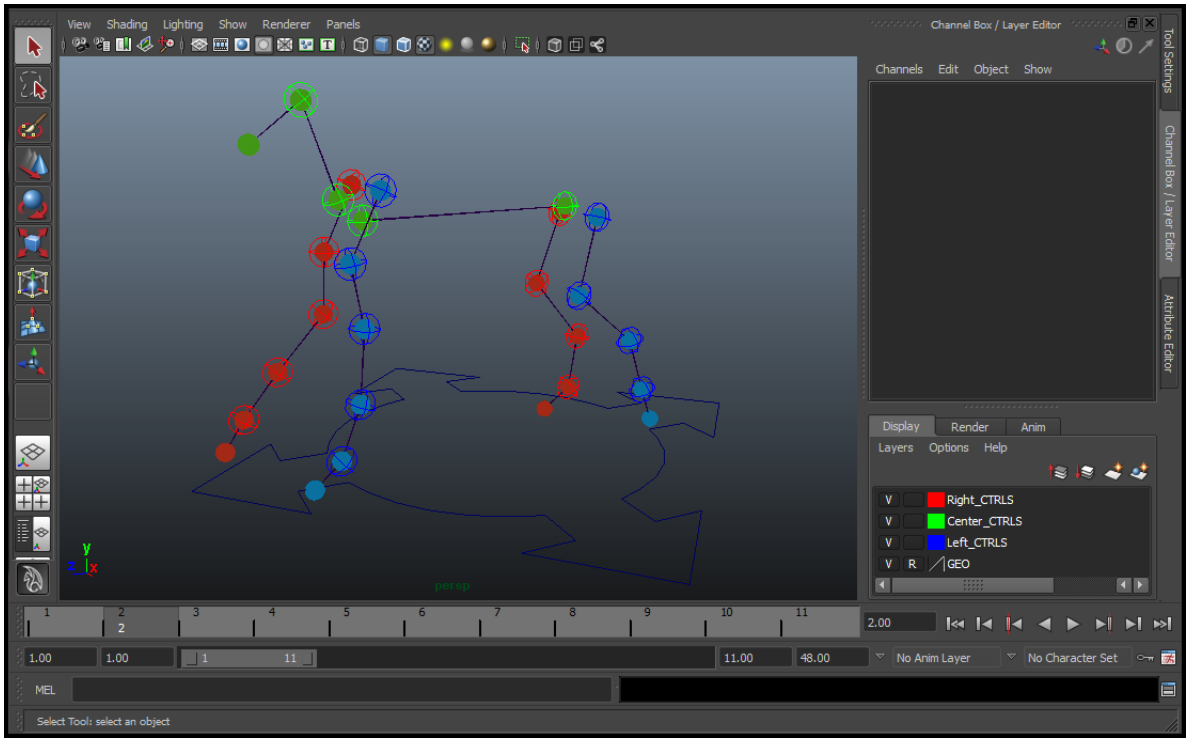


Figure 4.13: Imported Animation in Autodesk Maya

5. CONCLUSION

The web-based animation authoring application presented in this thesis provides a simple, intuitive way to author expressive animation for quadrupedal characters in real-time. The additions of the head and spine representations, along with translational movement of the legs improve the believability of the final animation beyond the limitations of past implementations of this method. The flexible rig and GUI enable the animator to achieve countless possible gaits and shapes, and the export feature allows the user to bring their work into Autodesk Maya, a popular 3D animation package. Ironically, the large amount of software packages and programming languages used during the implementation of this system allows for the animator to interact with just one lightweight tool that can be used almost anywhere.

This system provides many possible positive impacts in the film and game industries. The tool is easy to use, therefore enabling inexperienced artists to create convincing animation for quadrupedal animals. Character gaits can be infused with the specific traits of weight and age through the manipulation of specific sliders making production faster. Finally, web accessibility may encourage collaboration among animators working at distant locations and in separate production pipelines.

5.1 Future Work

5.1.1 The Third Dimension

The application in its current state is limited to motion in only two dimensions as the motion data was collected from video analysis. A method for integrating motion capture data would allow for the final third dimension to be added to the rig. This could potentially increase the believability of the final animations.

5.1.2 Additional Quadrupeds & Characteristic Expansion

Section 3.1 discusses the different bone structures between unguligrade, digitigrade, and plantigrade quadrupeds. The current state of the system only accounts for the unguligrade configuration. It also is limits the animator to only two characteristics: weight and age. A method that will incorporate the multiple bone structures, along with the addition of new expressive traits, such as danger, will greatly expand the possibilities for the final animation.

5.1.3 A Normalized Gait Sample

While the multilinear interpolation method for increasing and decreasing characteristics within a final gait has been proven, it only allows for one gait sample to serve as the epitome of each characteristic. For a system providing two characteristics as animation controllers with three gait type choices, only twelve gait samples can be represented. A method for normalizing multiple gait samples into one to represent a specific characteristic would allow for many more samples to be represented in the final gait cycle. This method also has the potential to save on loading time during the initialization of the plug-in. However, the method must be careful not to take away from what makes the characteristic perceptible during the process.

5.1.4 User Accounts

As mentioned in chapter 3, the ability to create user accounts for this application would be an extremely helpful addition. Unfortunately, a user can only save their custom shapes throughout one session. Once the user has closed the plug-in, if the animator would like to go back and rework an animation they must reconstruct their work based on the data saved in the animation file. A user account would allow a user to log in and out as they please while preserving any progress they save to their

account. An account would also allow for easier collaboration with other users by adding the ability to share animations and shapes with colleagues or even with an open database for everyone's use.

5.1.5 Ability to Import into Multiple 3D Packages

The goal of this application is to create animation that will be used in a production. The current method of exporting the animation is software agnostic as it simply writes the data to a text file; however, the animation importer script provided with the application currently only imports the animation into Autodesk Maya. A method that provides for the animation to be brought into multiple 3D software packages, such as Autodesk 3dsMax, Maxon Cinema 4D, or Blender, would be a significant improvement.

REFERENCES

[1] Unity 3D. *Unity Web Player and browser communication*. Available at `http://docs.unity3d.com/Manual/UnityWebPlayerandbrowsercommunication.html`.

[2] Unity 3D. *Unity 3D*. San Francisco, CA, USA. Available at `http://unity3d.com/`.

[3] Sue Blackman. *Beginning 3D Game Development with Unity: All-in-one, multi-platform game development*. Apress, 2011.

[4] T.W. Calvert, J. Chapman, and A. Patla. Aspects of the kinematic simulation of human movement. *IEEE Computer Graphics and Applications*, 2:41–50, November 1982.

[5] Clara.io. *Clara.io*. Ottawa, Ontario, Canada. Available at `http://clara.io/`.

[6] ClearDB MySQL Database. *ClearDB MySQL Database*. Available at `https://addons.heroku.com/cleardb`.

[7] Vicki L. Datt and Thomas F. Fletcher. *Gait Foot-Fall Patterns*, 2012. Accessed: 2014-09-26.

[8] Ben Houston et al. Clara.io: Full-featured 3d content creation for the web and cloud era. In *ACM SIGGRAPH 2013: Studio Talks*, SIGGRAPH '13. ACM, 2013.

[9] Python Software Foundation. *Python*. Available at `http://python.org`.

[10] The PHP Group. *PHP: Hypertext Preprocessor*. Available at `http://php.net/`.

[11] Heroku. *Heroku*. San Francisco, CA, USA. Available at `https://www.heroku.com/`.

[12] Gunnar Johansson. Visual perception of biological motion and a model for its analysis. *Perception and Psychophysics*, 14(2):201–211, 1973.

[13] Lynn T. Kozlowski and James E. Cutting. Recognizing the sex of a walker from a dynamic point-light display. *Perception and Psychophysics*, 21(6):575–580, 1977.

[14] Bio Motion Lab. BMLwalker Demo. http://www.biomotionlab.ca/Demos/BMLwalker.html. Accessed: 2014-10-19.

[15] John Lasseter. Principles of traditional animation applied to 3d computer animation. In *Proceedings of the 14th international conference on computer graphics and interactive techniques*, SIGGRAPH '87, pages 35–44, New York, NY, USA, 1987. ACM.

[16] Meredith McLendon, Ann McNamara, Tim McLaughlin, and Ravindra Dwivedi. Connecting the dots: discovering what's important for creature motion. In *ACM SIGGRAPH 2009: Talks*, SIGGRAPH '09, New York, NY, USA, 2009. ACM.

[17] Johannes J. Michalak, Nikolaus F. Troje, Julia J. Fischer, Patrick P. Vollmar, Thomas T. Heidenreich, and Dietmar D. Schulte. The embodiment of sadness and depression - gait patterns associated with dysphoric mood. *Psychosomatic Medicine*, 71:580–587, 2009.

[18] Neil Middleton and Richard Schneeman. *Heroku: Up and Running*. O'Reilly Media, Inc., 2013.

[19] Mixamo. *Mixamo*. San Francisco, CA, USA. Available at `http://mixamo.com/`.

[20] P. Myers, R. Espinosa, C. S. Parr, T. Jones, G. S. Hammond, and T. A. Dewey. The animal diversity web (online). http://animaldiversity.org, 2014. Accessed: 2014-09-26.

[21] Rob O'Neill. *Digital Character Development: Theory and Practice*, chapter 5. Elsevier, Burlington, MA, USA, November 2008.

[22] Ljiljana Skrba and Carol O'Sullivan. Human perception of quadruped motion. In *Proceedings of the 6th Symposium on Applied Perception in Graphics and Visualization*, APGV '09, pages 130–130, New York, NY, USA, 2009. ACM.

[23] David J. Sturman. A brief history of motion capture for computer character animation. In *ACM SIGGRAPH 1994 courses*, SIGGRAPH '94, Orlando, FL, USA, 1994. ACM.

[24] Frank Thomas and Ollie Johnston. *The Illusion of Life: Disney Animation*, chapter 3. Hyperion, New York, NY, USA, 1981.

[25] Nikolaus F. Troje. Decomposing biological motion: A framework for analysis and synthesis of human gait patterns. *Journal of Vision*, 2:371–387, 2002.

[26] Graham Walters. The story of waldo c. graphic. In *ACM SIGGRAPH 1989 Courses*, SIGGRAPH '89, Boston, MA, USA, 1989. ACM.

[27] Cord Westhoff and Nikolaus F. Troje. The inversion effect in biological motion perception: Evidence for a "life detector"? *Current Biology*, 16(8):821–824, 2006.

[28] MySQL Workbench. *MySQL Workbench*. Available at `http://www.mysql.com/products/workbench/`.

[29] Junze Zhou. Perception based gait generation for quadrupedal characters. Master's thesis, Texas A&M University, College Station, TX, USA, December 2013.

APPENDIX A

ANIMATION FILE (PARTIAL)

```
cheetahRun

# GAIT SETTINGS #
shape name: cheetah
gait type: 0
speed multiplier: 1
User-controlled gait speed
 walk: 25
 trot: 20
 run: 15
Characteristics
 weight: 0
 age: 0
 danger: 0

# SKELETON #
LEFT
 shldBlade: 2,12.61,-6.97
 shld: 2,9.98,-8.54
 elbow: 2,6.53,-5.28
 fFoot: 2,1.78,-7.06
 fBall: 2,0,-8.33
 fToe: 2,0,-9.66
 hip: 2,11.65,6.81
 knee: 2,6.69,4.65
 hFoot: 2,1.953,8.75
 hBall: 2,0,7.14
 hToe: 2,0,5.83

RIGHT
 shldBlade: -2,12.61,-6.97
 shld: -2,9.98,-8.54
 elbow: -2,6.53,-5.28
 fFoot: -2,1.78,-7.06
 fBall: -2,0,-8.33
 fToe: -2,0,-9.66
```

```
 hip: -2,11.65,6.81
 knee: -2,6.69,4.65
 hFoot: -2,1.953,8.75
 hBall: -2,0,7.14
 hToe: -2,0,5.83

CENTER
 neck: 0,11.88,-8.02
 head: 0,14.54,-11.22
 nose: 0,13.02,-13.58
 chest: 0,11.85,-6.52
 spineEnd: 0,12.31,5.87

# ANIMATION #
gait speed: 15
FRAME 1|18.78144|-36.16807|.....|-0.227105|-0.07696445|
FRAME 2|15.59292|-38.99955|.....|-0.232919|-0.07035204|
FRAME 3|12.76854|-31.06155|.....|-0.2673897|-0.08092302|
FRAME 4|10.71897|-13.90714|.....|-0.3164665|-0.0930302|
FRAME 5|9.093679|9.962521|.....|-0.3369529|-0.1054948|
FRAME 6|4.134848|37.53711|.....|-0.348442|-0.1216538|
FRAME 7|4.134932|51.71909|.....|-0.3178185|-0.1329156|
FRAME 8|12.92459|47.07108|.....|-0.2363529|-0.1384605|
FRAME 9|25.20415|33.67828|.....|-0.3306955|-0.382847|
FRAME 10|36.76882|19.33528|.....|-0.42067|-0.5737641|
...

# REFERENCE IMAGE SETTINGS #
filename: cheetah.jpg
mirrored: False
scale: 1
X offset: 0
Y offset: 0
```