GROUP BASED RIGGING OF REALISTICALLY FEATHERED WINGS

A Thesis

by

HEATHER VERNETTE HOWARD

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

MASTER OF SCIENCE

December 2011

Major Subject: Visualization

GROUP BASED RIGGING OF REALISTICALLY FEATHERED WINGS

A Thesis

by

HEATHER VERNETTE HOWARD

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

MASTER OF SCIENCE

Approved by:

Chair of Committee,	Tim McLaughlin
Committee Members,	Ergun Akleman
	Michael Morrison
Head of Department,	Tim McLaughlin

December 2011

Major Subject: Visualization

ABSTRACT

Group Based Rigging of Realistically Feathered Wings. (December 2011)

Heather Vernette Howard, B.A., Texas A&M University

Chair of Advisory Committee: Prof. Tim McLaughlin

Digital birds are used in computer graphics to replace live animals both for the safety of the animal and to allow for more control over performance. The current treatment of avian wings in computer graphics is often over-simplified which results in a loss realism due to the incorrect form and motion of the feathers. This research attempts to address this problem by using the structure and motion of real bird anatomy to inform the creation of biologically accurate kinematic motion for wings. The hypothesis of this thesis is that a wing rig which follows biological accuracy will appear realistic in motion and facilitate efficient animation. This thesis describes the creation of a rig generation tool, called *WingCreator*, usable in 3D animation software to guide the construction of biologically accurate wings while maintaining a range of artistically-driven variability in form. The control system for the kinematic motion rig is designed to provide animators with intuitive control over wing behavior intended to result in efficient re-creation of realistic wing action including flapping and folding. *WingCreator* was tested by two riggers and one animator to gain feedback on the tools efficacy. The user feedback indicates that the resulting rig provides a control system that facilitates efficient animation while maintaining artistic control over the wing. Users reported that realism, however, could not be judged due to the numerous contributing outside factors, such as animation, lighting and texturing, that affect the perception of realism. *WingCreator* and its creation methodology is intended to be placed in the public domain for use by anyone and will add to the currently slim body of knowledge for creating realistic avian wings. Once placed in the public domain it is expected that this rig will be appropriated by animators who wish to create more accurate bird wing motion and by riggers who may use the biologically-driven methodology as a model for further exploration into depictions of other animals exhibiting complex form and structural motion behaviors.

TABLE OF CONTENTS

CHAPTER Page

LIST OF FIGURES

CHAPTER I

INTRODUCTION

In the use of computer graphics for visual effects, digital creatures are often used to replace animals. Sometimes they are used for ethical reasons: putting a live animal on a production set can be stressful to the animal or could cause possible harm to the animal, such as a scene where a horse needs to fall. There are also performance reasons: some animals cannot be trained to perform as well as others or act on cue, or the performance required is beyond the scope of a real animals ability, or perhaps the animal needed for the scene is endangered and therefore cannot be acquired to film. Sometimes the animal is a fantastical creature and doesn't even exist in real life, such as a gryphon.

While digital creatures provide benefits of safety and performance control relative to live animals, their use is not without fault. Digital creatures can sometimes look and move in ways that the audience perceives as unrealistic, particularly in live action movies where the digital creatures are see in comparison to real environments, actors and animals.

Within the kingdom of digital creatures computer generated (CG) avian creatures present a set of problems that are unique and challenging relative to their non-winged counterparts. Unrealistic representations of CG birds and creatures with bird-like wings are often the result of overly simplified approaches to addressing the biological

This thesis follows the style of *ACM Transactions on Graphics.*

complexity of avian wing structure and action. The skeletal structure of CG creatures and control over it, known as rigging in computer graphics terms, enables animators to articulate a digital character and create performances [Ford and Lehman 2002]. Effective rigging is essential to the look and movement of the character. For winged creatures in visual effects projects, where realism is key, rigging and control must have a strong connection to biological structure and action for the animator to be able to define a good, believable performance.

The primary contribution of this thesis is a tool that can be effectively used for rigging a range of avian wing structures and produces rigs that contribute to the animation of realistic wing look and behavior. Feathers play a large part of making a digital photo-real bird look believable and realistic. Wings have a wide range of motion, from outstretched soaring flight to folded tight against the body which can cause significant control and interpenetration problems for feathers. A deficiency with many previous approaches is simplification of the variety of feather form and action relative to the movement of the wing. This thesis takes an elegant procedural approach to feather placement, control, and definition of variations so that the resulting rig more closely resembles a real bird's wing. Another problem with many previous efforts has been that the resulting rigs only provide realistic appearance in one position- flying or folded. Many prior works include efforts to avoid the issue of wing folding entirely by employing visual tricks and cheats ranging from not showing the entire wing in frame, to strategical editorial cuts, and even render tricks that try to account for feather interpenetration resulting from movement from extended to folded wing poses [Hiebert et al. 2006]. The rigging method developed for this thesis provides a single rig and control solution for animation that includes both outstretched and folded wings. In deference to visual effects production environments, the approach defined in this thesis also addresses the factors of art direct-ability and efficiency as considerations. Art direct-ability is important because a rig that is too limited in its application to a variety of avian designs and performance requirements will constrain the creative capacity of animators. Efficiency is important because a rig with controls that are difficult to understand, or that impede quick definition of performances difficult to use will fail to be effective in a production environment where artist time is budgeted.

Most published work on the topic of CG avian wings has focused on feather generation, placement and rendering. Very little has been published on the control systems and animation of wings. This lack of available public, verifiable knowledge, and sometimes conflicting information, can make it difficult for anyone who is not currently in a production environment to attempt the creation of a CG bird wing that represents the contribution to the state of the art. To address this issue, this thesis uses user feedback and critique in an attempt to validate the thesis. Three Master of Science in Visualization students at Texas A&M University, two riggers and one animator, were chosen based on their experience in rigging and animation and asked to give feedback on the *WingCreator* tool. The riggers were tasked with using the system to build an avian rig, while the animator was asked to animate the resulting model to match the footage of a real bird. Their feedback on the usability of the system is provided in the Results sections of this thesis. Feedback from riggers and animators indicate that while it is difficult to judge the effectiveness of realism in the rig, the resulting rig provides a control system that facilitates efficient animation while maintaining artistic control over the wing.

This project was developed to make a contribution to more realistic presentations of CG birds and bird-like creatures. Both the tool and examples of its use will be distributed through on-line computer graphics forums that feature similar, visual effects industry aligned, applications of rigging and animation techniques. It is the author's intention to monitor feedback from potential users and hope that this contribution adopted by others and thus advances the art and science of creating more realistic computer graphics birds.

CHAPTER II

BACKGROUND AND PRIOR WORK

II.1 Realism and Fantasy

Computer graphics characters are often used to portray birds in film because they are put in situations no real bird would feasibly be in and asked to perform in ways that real birds do not naturally behave in and sometimes do not have the physical capability. For example: In *Legend of the Guardians: The Owls of Ga'hoole* the main character is a barn owl named Soren who has to fight to free his fellow owls, a situation no real owl would ever be in and thus have to give performances, such as fighting other owls to escape to freedom, talking and wearing armor. While the birds may be photo-real in style, their performance was unlike a real bird. This is the difference between something created for film that is photo-real vs. something created for scientific visualization. In a more scientific setting, the birds are limited to that which real birds can perform. Film is driven by the performance needed to tell a story and will stretch beyond the limits of a real bird.

II.2 Animation

The first person a rig caters to is the animator, as the animator is the one who will be using the rig to create a performance with. To create a good rig for an animator it is essential to understand how animators like to work. “There is no better way to get animators to produce good work than by presenting a rig built by a character TD

(Technical Director) who understands how animators like to work" [Ford & Lehman 2002]. A basic knowledge of the 12 Principles of Animation is very beneficial as well as understanding which ones apply specifically to the performance of realistic birds.

The 12 Principles of animation introduced by Walt Disney's animators Frank Thomas and Ollie Johnston in *The Illusion of Life* are a set of animation principles [Thomas and Johnston 1981]. These principles give the illusion that a character is alive, follows the laws of physics and adds appeal, both emotional and visual, to the character. While these rules were originally created to apply to traditional (hand drawn) animation, they were updated for use in computer animation by John Lasseter and while there is not universal agreement about the importance of each principle, the notions of the principles has become a standard to use for all animation [Lasseter 1987]. The principles and the reasons they apply to computer generated birds are:

- <u>Squash and stretch</u>: The principle of how an object will squash and stretch when in motion. An object does not lose its volume when it does this and this can be seen in a wing when it opens and closes. The muscle in the wing squashes (closed) and stretches (open) and the feathers on top of that collapse into a smaller area ("squash") and fan open ("stretch").
- <u>Straight ahead action and pose to pose</u>: These are two different approaches to animation. Straight ahead is successive drawing, frame by frame from one scene to the next whereas pose to pose is drawing out the key poses and then filling in between those key poses. Bird flight can be animated using either method. CG facilitates pose to pose animation

because the software can automatically interpolate between the key frames; however, often more work needs to be done to make the interpolation not look natural [Williams 2006].

- Follow through and overlapping action: Follow through is when parts of a body continue moving after the body has stopped and Overlapping action means that parts of a body move at different rates. For example: In birds these apply to the feathers dragging through the air, causing them to deform. Follow through would be when the main portion of the wing changes direction, that drag from the air causes the feathers to take a few frames to catch up to the main body and Overlapping Action would be the primary feathers moving at a different rate than the secondary feathers because of difference in flexibility of the feathers and location on the wing.
- Arcs: Most movement made by living creatures move along arcs. Using arcs when animating a wing gives more realistic movement to the wing.
- Exaggeration: To exaggerate an action or performance beyond what can realistically happen. Often action traced from live action can be accurate but it can look stiff or mechanical. Exaggeration is "like a caricature of facial features, expressions, poses, attitudes and actions" and can help give more appeal. In birds this can be an exaggeration on how far the wrist is bending during a frame or two in order to enhance the action of a wing beat [Lightfoot 2004].

II.3 Birds as Digital Characters

Within the realm of computer graphics birds, there are many uses of birds. Birds can be background characters, used to fill in the background of a scene. These background characters are only seen from afar and are therefore limited in performance and detail, sometimes so that only the profile is seen. These birds are out of the scope of interest for this project because for something seen only at a distance for a short amount of time does not need a great deal of complexity to it. The interest of this project is for what is sometimes referred to as a "hero character" - a character which is often seen close up, has a large amount of screen time, and a large range of performance (in the case wing behaviors) required.

II.4 A Brief History of Birds in Film

CG avian creatures as characters have a long history in film and covering all of them would be difficult. I would like to touch on some of the more notable digital avian creatures and discuss relevant work in regard to the rig, feather motion, and challenges faced in each case. Also, while many films have had birds in them, not all of them are relevant. The following brief history discusses avian creatures that fit the scope of this thesis- those that are realistic and not heavily stylized.

II.4.1 The Chronicles of Narnia: The Lion, the Witch and the Wardrobe (2005)

In *The Chronicles of Narnia: The Lion, the Witch and the Wardrobe* by Walt Disney Pictures with visual effects by Rhythm & Hues there are gryphons: half-lion half-eagle creatures. (See **Figure 1**). The gryphons have a small role overall but they are featured in shots, both flying and with wings folded, close up and full frame. This required that the gryphons be highly detailed and as anatomically accurate as a mythical creature can be. A big challenge to this was the look and actions of the feathers [Hiebert et al. 2006].

The feathers on the gryphons (coverts, and body feathers) were generated procedurally using Rhythm & Hues' in house fur software, except for two rows of flight feathers [Hiebert et al. 2006]. The implementation of feathers this way looked realistic for the gryphon when flying. However, the gryphon's performance required that he fold his wings. Rhythm & Hues' default set of tools worked well in general, with few interpenetrations perceptible; the problem came with the flight feathers.

In Rhythm & Hues' avian rig designs previous to the one used in *Narnia*, a spline along the trailing edge of the wing that the feathers pointed at was used to control the wing; this allowed the animator to have a lot of control with a minimum amount of control objects and worked well for flight motion. However, for wing folding, it caused the feathers to stack up incorrectly relative to the look and behavior of feathers on real birds. To solve this, the flight feathers were broken up into two separate groups of primary and secondary feathers with a separate spline to control each. It allowed the primaries and secondaries to stack up believably and an additional layer of control was added to allow the animator to control each feather individually and to clean up any feathers that were out of place or interpenetrating [Hiebert et al. 2006]. It works well for wing folding on the two main groups and something similar could be applied further to the separate feather groups in the wing for a more photo-real bird wing.

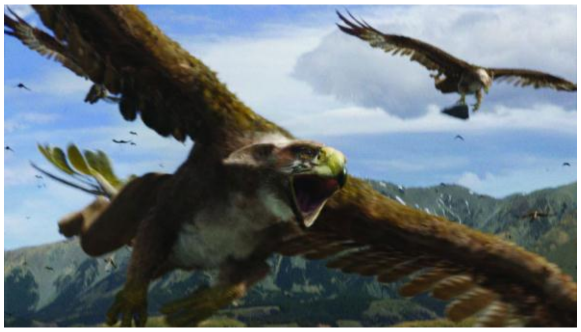

Figure 1. Gryphons from T*he Chronicles of Narnia: The Lion, the Witch, and the Wardrobe* © 2005 Walt Disney Pictures.

II.4.2 Harry Potter and the Prisoner of Azkaban (2004): Buckbeak

In the third Harry Potter movie, *Harry Potter and the Prisoner of Azkaban* by Warner Brothers Pictures with visual effects by Moving Picture Company (MPC), MPC developed highly detailed wings for the hippogriff, a mythical creature that is half horse half eagle, Buckbeak. He was in the film for about 10 minutes, and had several up close, full frame hero shots and, like the gryphons in *Narnia*, the hippogriff had performances with wings opened and folded. While published information on a bit vague on the production techniques used to make the wing, possibly due to the proprietary nature of the work, Buckbeak is a good example of a wing looking nearly photo-real when opened, but looking unrealistic when closed. (See **Figure 2**). When closed it loses believability because of the way the feathers are overlapping; the different types of feathers make angles to one another not found in a real birds wing or overlap incorrectly, causing the wings to look flat.

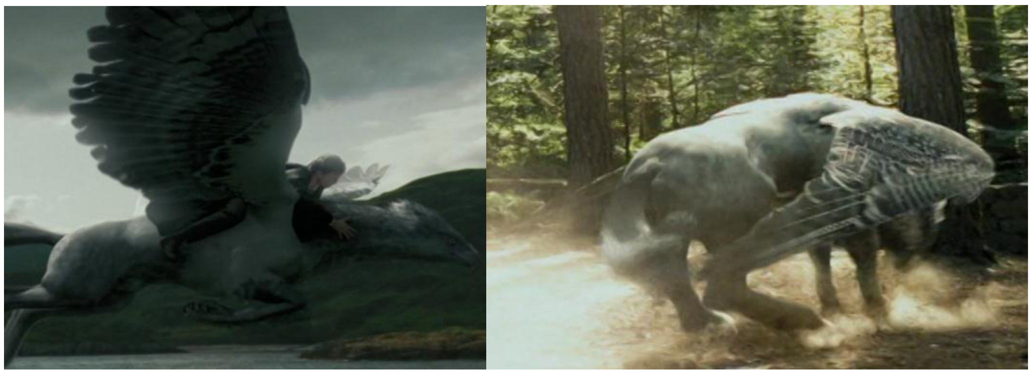

Figure 2. Buckbeak the hippogriff with wings open and folded from *Harry Potter and the Prisoner of Azkaban* © 2004 Warner Bros. Pictures.

Every feather on Buckbeak was created was modeled after a real feather in which it had geometry representing the central rib called the 'rachis' and filament like barbs and some feathers were even more detailed, using geometry to represent the little Velcro-like hooks holding the barbs together called 'barbules'. The feather models used depended on camera distance- the more detailed feather models were used for closeups and the less detailed ones were used for distance. A total of 16,500 feathers applied to the hippogriff's skin [Fordham 2004]. The feathers were driven by MPCs proprietary muscle system enabling skin jiggle. This allowed the feathers to move with the underlying musculature and skin for a more realistic look. The motion of the wings caused problems with the interaction of muscles and feathers. They needed a way for the feathers to stack up correctly when folding and to prevent intersection. The resulting rig prevented penetrations of the flight feathers when folding, using a program that resolved intersections and compressed feathers against one another in the closed wing.

II.4.3 Lord of the Rings trilogy (2001-2003): Gwaihir

For the movie *Lord of The Rings: Fellowship of the Ring* by New Line Cinema with visual effects by Weta Digital, Ltd. a large eagle named Gwaihir was scripted. He was a background character with relatively little screen time and the scenes he was in were short and from a distance or at night so little detail was required since it was not going to be visible in those conditions (See **Figure 03***)*. Unlike Buckbeak or the gryphons in *Narnia*, Gwaihir's performance never required him to fold his wings across his back; his performances were all aerial. “The fill eagle used for *The Fellowship Of The Ring* was a cheat – the bare minimum to final the shots [Aitken et al. 2004].” For *The Return Of The King* a much more detailed model was built because the eagle had a role requiring closer, more detailed shots. The feather setup for *The Return of the King* was done with layers differentiated types of feathers between “hero” feathers, feather code, and fur feathers. The hero feathers were nurbs patches and the animators could control them through the rig. The hair feathers were generated over a separate subdivision skin that was “like a sock around the original surface of the bird and was cleverly rigged not to interpenetrate the hero feathers when they moved [Aitken et al. 2004].” Gwaihir is an example of the simplified approach often taken toward wings in which there are two rows of feathers, and while this approach is adequate for the role of a background character, it may not be suitable for more detailed hero characters.

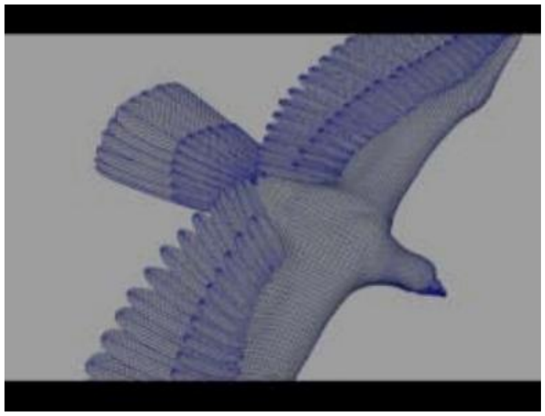

Figure 3. Gwaihir the eagle from *The Lord of the Rings: The Fellowship of the Ring* © 2001 New Line Cinema [Aitken et al. 2004].

II.4.4 Clash of the Titans (2010): Pegasus

The focus of the published information available on Pegasus in *Clash of the Titans* by Warner Brothers Pictures with visual effects by Moving Picture Company (MPC) mainly focused on the treatment of the feathers and touches only briefly on the wing. The winged horses had several hundred shots in the film and ranged in detail from background characters to highly detailed, full frame hero shots. Each feather was procedurally created, or generated via an algorithm, as curves in three dimensions. The feathers were generated at rendertime and used MPC's “Furtility” fur/hair utility. This gave the feathers a fluffy look to them, as well as allowing the groom artist to use the same tools they already had for hair grooming. Feathers were automatically distributed across the wing, as well as individually placed and sculpted. Since the actual feathers were created during render time a rig using simple NURBS (or 'Non-Uniform Rational Basis Spline') surface was substituted as proxy feathers in the wings for use by the animators. With this rig, additional animation and tweaks on the feathers could be done [Leaning and Fagnou 2010].

Like other films, the treatment of the wing and feather interaction overall was to place them as two main rows of feathers and procedurally generated the rest of the feathers in the wing, therefore when folding the wings tend stack up in an accordion fashion, very evenly. This gives the wings a very fluffy look, but also a slightly unrealistic overall. (See **Figure 4**).

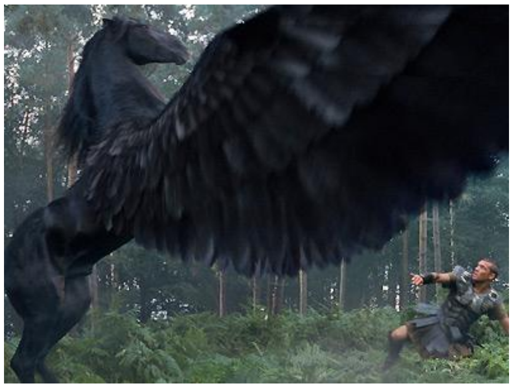

Figure 4. *Clash of the Titans* © 2010 Warner Bros. Pictures.

II.4.5 Legend of the Guardians: The Owls of Ga'Hoole (2010)

The owls created by visual effects company Animal Logic in *Legend of the Guardians* by Warner Brothers Pictures were highly detailed since the owls were the main focus of the entire film. (See **Figure 5**). This required that the wings be able to cover a wide range of performances such as flying, wing-folding, and anthropomorphic acting and and equally wide range of shots, from background to hero.

Much like real owls, the owls the Animal Logic owls without their feathers were skinny and chicken-like; all their volume and shape came from their feathers. The remiges, or main flight feathers, were connected to the wings and the animators had direct control over them; there were 24 in each wing. The animators had a lot of control over the wing and feathers: They had about 7 controls on each wing overall and another 10 to 15 on the feathers. They could rotate and translate individual feathers as well as bend, curl and cup them to simulate the force of wind [Robertson 2010].

Body and wing covert feathers were procedurally generated and controlled by the character effects team. Animal Logic's effects team had to re-write their feather system, Quill, from the ground up. Guide hairs were hand-placed on the models and had parameters for each feather that defined the look of the feather, from length and width, to how smooth or uniform the feather is from the base to tip.
Quill also added secondary motion such as wind effects on these feathers once animation and a first pass at simulation was in place [Robertson 2010].

Animation style was also an issue due to the realistic look for the film. “We did some early tests that were a lot more cartoony: squash-and- stretch, bigger arcs, heavier anticipations, but as soon as we placed these owls in our realistic world, they didn't belong.” The realistic style of the movie made this animations style not fit, however, Animal Logic wanted to avoid making it “a documentary”, so it became a balancing act using naturalistic animation for flying or walking, and using more traditional animation techniques for performances and talking [Desowitz 2010].

Figure 5. *Legend of the Guardians: The Owls of Ga'Hoole* © 2010 Warner Bros. Pictures.

II.5 The Form and Motion of Real Birds: Morphology of Aves

Bird wings are very diverse because of the various modes of flight seen in birds. There are, in general, four wing shapes attributed to flying birds: long and narrow (soaring birds, like albatrosses), short and round (like in grouse which are good for quick takeoff and maneuvering), slim and un-slotted (like in falcons for speed), and intermediate dimension slotted wings (like in hawks for gliding ability) [Gill 1995]. Despite the great diversity in birds and their wings, they still have a common overall bone and feather structure which can be seen in most flighted birds [Sibley 2000].

Bones: Bird wings are a modified forelimb, similar to the human arm: Scapula, humerus, radius and ulna, carpus and metacarpus, and the phalanges fused together forming the “hand”. (See **Figure 6**). The shoulder consists of the scapula (shoulder blade), coracoid (projecting part of the shoulder blade), and humerus (upper arm). The humerus joins the radius and ulna (forearm) to form the elbow. The wrist bones, carpus and metacarpus are fused together forming the carpometacarpals and the digits (fingers) are fused together in three digits. The alula (thumb) is also known as the “bastard wing” and moves independently of the rest of the wing [Gill 1995]. Due to the similarities with a human arm and for simplicity, the terms used for a human arm (“wrist”, “elbow” and “shoulder”) will often be used throughout this work.

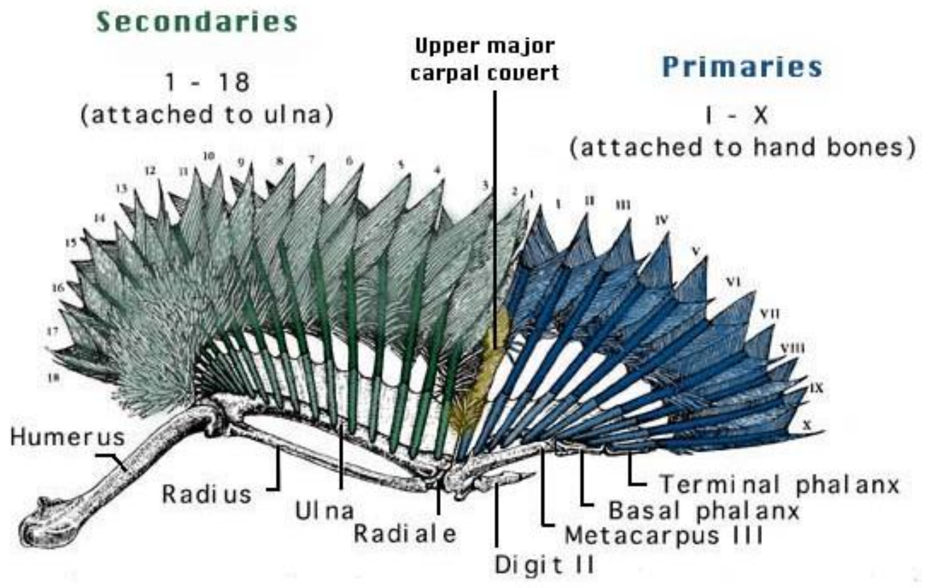

Figure 06. Bird wing anatomy [Lucas and Stettenheim 1972].

Feathers: There are three main types of feathers: vaned feathers, down feathers and filoplumes. Filoplumes are hair like and monitor movement and position of adjacent vaned feathers. These feathers are distributed throughout the plumage and are normally not visible due to their small size. Down feathers have no rachis (or central shaft) and provide insulation to the bird. Down feathers lie under the vaned feathers and thus are also usually not visible, except on baby birds whose vaned feathers have yet to grow in. Because of this, down feathers and filoplumes are not seen in computer graphics except for rare instances (like baby birds with down) and are not taken into consideration in this project.

Vaned feathers are the feathers that cover a bird’s body and have of a rigid center (rachis) and soft barbs on the side. There two different types of vaned feathers found in the wings: remiges and coverts. The remiges are the flight feathers and consist of the primaries, or outermost remiges, and secondaries, the innermost remiges [Gill 1995]. These feathers make up a large portion of the shape of the wing, so much so that the length and shapes of the primaries and secondaries are used as field marks to identify different species of birds [Sibley 2002]. Coverts are, as their name suggests, feathers that cover the wings. These feathers can visually be broken up into several groups. The arrangement of these feather groups is similar across all species of flighted birds even though the shape of individual feathers varies [Sibley 2002]. Often when trying to identify a bird by appearance, birdwatchers will use attributes of these feather groups (such as color and shape) to tell one bird from another. Within each group, feathers grow in orderly rows, overlapping like shingles on a roof, and plumage patters tend to follow the contours of these feather groups [Sibley 2000]. These feather groups are primaries, secondaries, greater primary coverts, greater secondary coverts, median secondary coverts, lesser secondary coverts, and alula [Sibley 2000]. (See **Figure 7**).

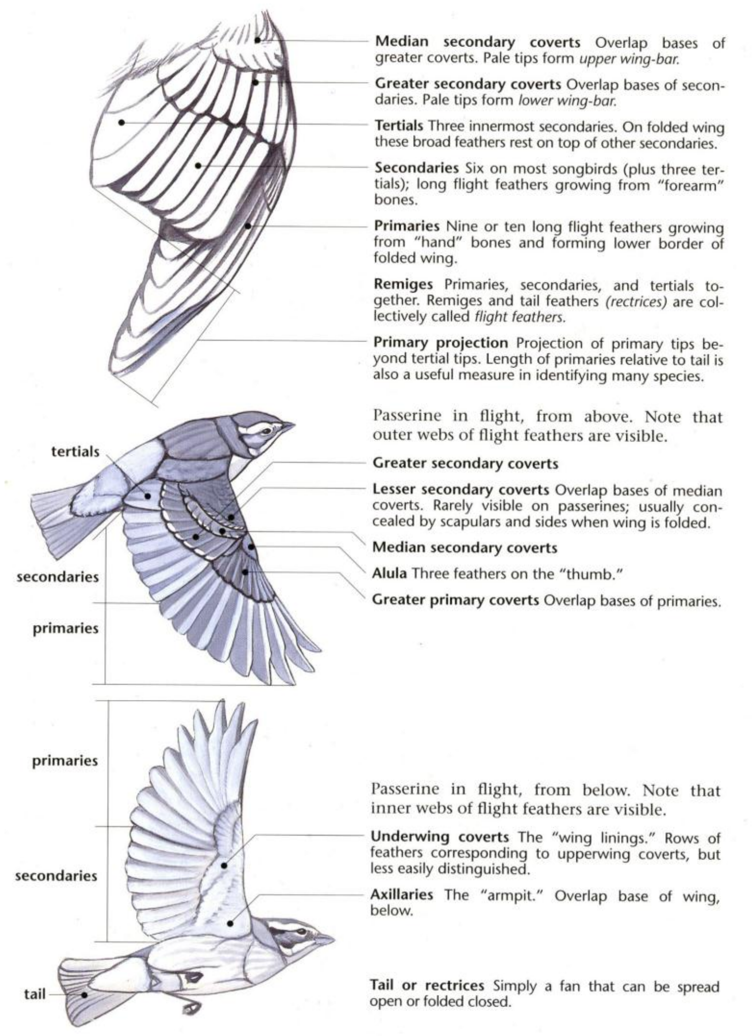
Median secondary coverts Overlap bases of greater coverts. Pale tips form *upper wing-bar.*
Greater secondary coverts Overlap bases of secondaries. Pale tips form *lower wing-bar.*
Tertials Three innermost secondaries. On folded wing these broad feathers rest on top of other secondaries.
Secondaries Six on most songbirds (plus three tertials); long flight feathers growing from "forearm" bones.
Primaries Nine or ten long flight feathers growing from "hand" bones and forming lower border of folded wing.
Remiges Primaries, secondaries, and tertials together. Remiges and tail feathers *(rectrices)* are collectively called *flight feathers.*
Primary projection Projection of primary tips beyond tertial tips. Length of primaries relative to tail is also a useful measure in identifying many species.
Passerine in flight, from above. Note that outer webs of flight feathers are visible.
tertials
Greater secondary coverts
Lesser secondary coverts Overlap bases of median coverts. Rarely visible on passerines; usually concealed by scapulars and sides when wing is folded.
Median secondary coverts
secondaries
Alula Three feathers on the "thumb."
Greater primary coverts Overlap bases of primaries.
primaries
primaries
Passerine in flight, from below. Note that inner webs of flight feathers are visible.
Underwing coverts The "wing linings." Rows of feathers corresponding to upperwing coverts, but less easily distinguished.
secondaries
Axillaries The "armpit." Overlap base of wing, below.
tail
Tail or rectrices Simply a fan that can be spread open or folded closed.

Figure 7. Bird feather groups [Sibley 2000].

The main area of interest in this project is feathers in the wing that interact like venetian blinds or fans. These are the primaries, secondaries, alulas, and primary, secondary and median coverts. These feather groups are the main focus because they are the ones with the most movement and expression to them, making up the main body of the wing. They are often the feathers that are over-simplified which degrade the realism of the wings. Feathers such as the marginal covert feathers act more as scales instead of fans or venetian blinds and have very little range of motion to them; their portrayal in film is often reasonably accurate and believable because of their simple nature and limited motion. There has also been previous work in this area such as in the papers by Seddon et al. [2008] and Weber and Goronwicz [2009]. Due to the fact that these feathers have a considerable amount of previous work on them and are often treated accurately in wings, unlike the main feather groups, they are considered outside of the scope of this project.

II.6 Development of CG Birds: Feathers

There are numerous papers which discuss methods of creating realistic feather geometry, feather coats, and solving for interpenetration of feathers. While this does not relate specifically to rigging, these aspects have an impact on the realistic look of a bird. These could be used in conjunction with a wing rig in order to improve realism of the digital bird. The following are just a few representations of this work

Bangay [2007] presents a technique to place feather coats on an animated object. Bangay's method uses a vector field in the space surrounding the body of an object which then deforms feathers to align with those field lines. This results in feathers that are consistent when the object is animated and ensures that the feathers are aligned and do not inter-penetrate [Bangay 2007]. This method generates realistic feather coats for covert feathers, such as those on the body of the bird and lesser secondary coverts- feathers that act more like scales. For non-scale feathers such as the feathers in the wing of a bird, this method is inappropriate due to the motion of these feathers, which is vastly different from that of scale-like feathers. This technique and the rig in this thesis project would complement one another; between them they cover the complete range feathers and feather motion in a bird.

Chen et al. [2002] offer a technique using a branching system to define realistic feathers. Different types of feathers can be approximated and different attributes can be changed by just altering the parameters in the system [Chen et al. 2002]. The strength of this system is that it can create a wide range of realistic feathers from these user given parameters, however, the downfall to this system is that these feathers are distributed across a bird in manner not biologically accurate to real birds causing the realism to be lost. A system like this used in conjunction with a feather generation program that follows how feathers grown on a real bird, much like the feather generation for the wings in this project, would lend itself to very realistic looking wings.

Seddon et al. [2008] describes a method of procedurally generating realistic feathers that allows them to be artistically groomed. They can be groomed on several attributes: splitting of feathers, scraggle (displacement of feathers), tangle (scraggle that accumulates down barbs), and clipping (cuts between the barbs.) This technique is similar to both Bangay [2007] and Chen et al. [2002] in that it is a method that focuses on the look of the individual feathers themselves and a method of generating and grooming a feather coat. Also similar to both methods, this method would complement a rig such as the one in this thesis project, helping to increase the realism on a CG bird through the implementation of realistic feathers and feather coats in addition to the realistic wing structure and feather movement resulting from this project.

An intuitive approach to automatically generating feather coats is offered by Streit and Heidrich [2002]. This paper particularly addresses the range of colors and patterns in a feather coat and different types of feathers in a feather coat where previous methods generated only one type. Like Chen et al. [2002] this also focuses on the branching aspect of the feathers as a method of parametrization. While this method is useful for generating a wide range of feathers, the feathers generated do not increase realism; the resulting feathers look more stylized, very soft with an airbrushed quality to it and not quite realistic.

Goronwicz and Weber [2009], two animators from Dreamworks Animation, describe the method of making the feathers on Crane from *Kung Fu Panda* using implicit constraint surfaces. Their method focuses on creating feathers that will not penetrate and will result in visually smooth animation [Goronwicz and Weber 2009]. Both this paper and Bangay's address the motion of the feathers when animated as well as solving for interpenetration of feathers. Also similar to Bangay's paper this method could be used in conjunction with the rig presented in this thesis project to solve interpenetration issues on the feathers, creating a visually more appealing and realistic wing.

II.7 Development of CG Birds: Production Environment

The production environment requires that the form is driven by an art director who is responding to the director's vision for a film. This necessitates choices about the strategy implemented when creating the rig for the film. Unlike the previous section which discusses methods for solving problems on the rig itself, the example found here focuses more on past methods used in production and how to incorporate them into future work.

Hutchings [2007] offers a project to produce a fully modeled, textured and rigged bird wing that can fold, open and flap effectively. She describes various methods of modeling and rigging that have been utilized in past film and games and how they apply to her wing rig as well as offering a tutorial of how the rig was created. Much like previous methods, such as those used on Gwaihir in *Lord of the Rings*, her rig uses two main rows of feathers: remiges and coverts. The author bases the rig on a real bird’s wing and uses the same skeletal structure as a real bird in order to improve the realism in the wing [Hutchings 2007]. There are many similarities between Hutchings' project, and this thesis project such as the research of previous methods of bird wing rigging, creation of a methodology for wing rigging, and basing the rig off real bird anatomy to try and improve realistic bird motion.

II.8 Development of CG Birds: Scientific Visualization

Rigging for film has different requirements than that of rigging for scientific visualization; the goal in rigging for film is that of facilitating performance requirements defined by the animators, whereas rigging for scientific purposes is to facilitate learning and hypothesis testing. This means that the rigs for the former generally are more art direct-able, and while based in realism, will often give the bird a range of motion and abilities that are not found in real birds, whereas rigs for the latter try to strictly adhere to the motions and abilities of a real bird.

Wu and Popovic [2003] describe a physics-based method to synthesize bird flight in which the bird follows a specific trajectory. The authors rig the bird using an articulated skeleton with elastically deformable feathers and use joint torques and aerodynamic forces over time [Wu and Popovic 2003]. Wing beat motion is solved separately. All these calculations are put together to simulate realistic bird flight for different birds in scenarios such as landing, taking off, and rapidly descending [Wu and Popovic 2003]. This rig and resulting animation is highly realistic regarding the motion of the bird as a “living” creature as well as the biological motion of its wings and feathers. While the result is highly realistic, the resulting animation is a product of inputting variables such as wind, drag, and momentum which controls and animates the rig instead of an animator’s effort to produce a performance- there is very little artistic control over this and it would not be very feasible to use in film.

II.9 Development of CG Birds: Videos and Tutorials

Not all techniques are published by Industry professionals. There are many techniques, tutorials, and videos produced by armature riggers and others on the internet. The internet community, especially in the field of computer graphics, is a valuable source of information because it hosts a great deal of readily available knowledge such as tutorials on techniques, videos of others works, and even useful tools that are available to download. While the net can be a great source of information, it can also be problematic because what is seen and claimed in these videos and tutorials cannot always be validated or often contains no information on how the technique was implemented. Despite usually lacking detailed information, the following videos and tutorials have relevance to this project, often as an inspiration point for idea generation.

Addanki [2010] demonstrates a modular bird rig created in the 3D software program Maya. This rig is generated using several scripts written using Mel and Python [Addanki 2010]. The control system for the feathers in this video was the inspiration for the final method of control on the feathers for this thesis project; it allowed for flexibility to shape the wing as desired while maintained the realistic movement of the feathers themselves. What detracted from the realism in this rig, however, was the use of extraneous bones in the wings. While a real wing has three main rotation points in it- shoulder, elbow, and wrist - Addanki's rig has eight. This makes the wing very flexible and able to make smooth arcs, but breaks the realism of it because the wing becomes too flexible, almost cartoon-like.

Mike Paixao has written several tutorials, many of which are on his site at http://simpletofind.ca/. Most relevant to this project is a tutorial on how to set up a wing using a feather tool called *Mfeather*, found on Creativecrash.com [Paixao 2010]. While the resulting rig is not realistic at all in either form or motion, it has an amazing amount of flexibility in the control system. The movement of the feathers, in particular the splaying and folding of the feathers when the main joint control is manipulated inspired the high level feather control system for my thesis.

Jo Plaete has written a tool that procedurally generates and rigs feathers in a wing. This tool is created in Python, uses Actionscript to create the user interface, and

Vbscript [Plaete 2008]. The control system for this wing is similar to the one created by Addanki which affords the wing a great deal of control and flexibility. Also similar to Addanki's rig is that the wing uses an unrealistic bone structure, causing the wing to have non-realistic wing movement. Unlike Addanki's tool, this tool uses different layers of feathers that the program allows the user to switch between. This allows the user to control the feather behavior and add as many layers of feathers as needed.

CHAPTER III

METHODOLOGY

III.1 Scope of the Project

The main goal of this project is the creation of a tool that can be integrated into a 3D software package which provides a work flow that is efficient and effective for both rigging and animation of realistic avian wing motion. To accomplish this goal, the scope of the project is to covering the needs of artists that are specific to these two tasks and limited to a specific type of wing form and motion. Avian characters are so diverse, in anatomical variation and style of flight that it would be difficult for one project to cover that entire range in aves. Since this project focuses solely on one part of anatomy, avian wings, the methodology will therefore not cover a broader range of issues typically considered when dealing with a more comprehensive approach to character rigging.

The audience for this tool is artists dealing with rigging and animation issues common to the visual effects film industry. As such, biological realism must be balanced with artistic control. The rig must contribute to the creation of an animated performance which, in turn, contributes to storytelling. Thus, providing the capacity for purely biologically accurate movement will not be sufficient.

The main ideas of this project are:

- **Control over feather placement**: Feathers are laid out into groups as they would be on a real bird's wing.

- **Management of feather interaction**: Feathers move as they would on a real bird. Real bird’s wing feathers move like fans or venetian blinds, each feather stacking in a certain order allowing the bird to change the shape of the wing as it is opened and closed. Each feather group also interacts properly with the groups surrounding it. This is an important aspect to the realistic movement- it is more biologically accurate which facilitates natural movement and contributes to believability.

III.1.1 Defining Realism

A digital characters looks, or design style, can be classified into three categories: primitive, abstract and naturalistic. Primitive is a character that is simplified down to base characteristics and this style is often associated with the cartoon aesthetic. Abstract characters are those whose elements are, while plausible, proportioned or combined in ways not found in nature. Lastly, naturalistic characters are those which look and behave like a creature or person from the real world [McLaughlin, 2006, p. 5-6). This project focuses mainly on creatures that are naturalistic in style and excludes those which are primitive. Examples of these primitive style birds are Kevin in Pixar's *Up* (2009), Vlad from Blue Sky's *Horton Hears a Who* (2008), and the birds in Pixar's short *For the Birds* (2000). While abstract characters may be mentioned, such as Buckbeak the Hippogryph from *Harry Potter and the Prisoner of Azkaban* (2004), the main focus is on the wings of the character, which are naturalistic in style even if the overall character is abstract.

Even narrowing the scope to naturalistic birds, the range is still too broad. Birds have a great amount of diversity of form and have many different types of wings for different types of birds, even if the overall structure of the wings is the same. Different wings have different shapes; long and narrow for seagull, broad and round for monkey eagle for example. In the interest of time and workload, the scope is narrowed to the form and movement of one type of wing: broad hawk wings, more specifically red-tailed hawk.

III.1.2 Feather Types Considered

Some parts of avian wings have been implemented successfully before, such as the body feathers on a bird and the covert feathers in the wing. Some examples of such work can be seen in the papers *Generating feather coats using Bezier curves* [Streit and Heidrich 2002] and *Animated feather coats using field lines* [Bangay 2007]. These feathers do not have much range of motion and their placement over the wing is more like that of scales compared to the other feathers in the wing, which act as venetian blinds or fans. This project focuses on the feather groups in the wing that have not had as much prior work done on them, therefore this tool only focuses on feathers that act as venetian blinds or fans and disregards those that do not. The following groups of feathers are built into the rig:

- primaries,
- secondaries,
- primary coverts,
- secondary coverts,

- median coverts, and
- alula.

Scapulars and lesser secondary coverts are not included because these feathers fall under the category of scale-like feathers.

III.1.3 Ignoring Surfacing Issues

Lastly, texture and design of the feathers themselves also are not taken into consideration because the focus is on the rig itself and the movement and interactions of the feathers. Like scale-like feathers, there has been numerous works done on how to make effective and realistic looking feathers such as *Modeling and rendering of realistic feathers* [Chen et al. 2002] and *Rendertime procedural feathers through blended guide meshes* [Seddon et al. 2008].

III.2 The Importance of Form

A bird's wing without feathers is a stubby awkward looking thing that makes one think more of buffalo wings instead of flight. The feathers and the way they grow create the form and look of the bird. In order to create a realistic looking bird it follows that the form must be accurate.

The rigging tool is based around this concept of accuracy of form. A bird's feathers do not grow uniformly over its body; they grow in organized tracts and the resulting coat can flex, expand and contract. The biology of how and where the feathers grow is used as a basis for how the placement tool operates. The user has the capacity to alter limb length, add more feathers in the wing to create a denser coat, and give the program different polygonal models to change the feathers. However, the user cannot redefine the underlying biology, such as adding more bones.

The user has control of how many feathers the program generated, but the program determines how the feathers are propagated over the wing based on the joint placement. Since the wing anatomy of all birds is mostly the same (barring some exceptions such as penguins or long winged birds) the user is not allowed to change the placement of any feather groups on the wing or how the feathers are propagated in any group. Individual feathers can be adjusted once they are generated (scaled in any direction or removing a feather entirely) but altering the placement of the feathers as individuals or a whole group will cause unexpected results or break the rig, therefore the user is not allowed to do so.

The wing generation program creates a control for automated wing folding. This automated wing fold is a control that will set the wing in a default folded pose; this helps to streamline animation by giving the animator a control to quickly put it in a commonly used pose. The default wing fold pose the program generates cannot be altered in the generation program, unlike the feathers which can be changed in number, however, the resulting pose can be used as a base and add animation on top of it or change it as needed once it is generated.

In summary, the user is provided control over three biologically-driven factors that can also have a significant artistic impact:

1. The number of feathers in each group;

2. The length/proportion of the bones;
3. The shape of the feathers.

However, to preserve realism, the user is prohibited from altering these biologically-driven factors:

1. The number of bones;
2. Feather group placement;
3. Base wing folded pose.

III.3 The Importance of Motion

Even when the form is correct, if the wing movement is incorrect then the believability is still lost. Two factors contribute to motion: the flexibility of the rig and the skill of the animator. An artist who is skilled in the craft of animation when given a rig that is hard to use or created to be un-realistic will have a difficult time creating a realistic animation. In creating the rig, a great deal of consideration is given to the motion of the feathers and the best way to give the animator control over that motion while still maintaining biological realism.

III.3.1 High-level and Low-level Control

Control of the wing is broken up into two sets of controls, high-level and low-level. Doing so makes the controls more intuitive and quicker to use. The high-level controls are the overall controls which allow the animator to very quickly pose the wing and feathers. These controls drive the joint movement, either through FK or IK motion, for overall posing of the wing and also controls the overall spread of the feathers. Low-level controls give a finer amount of control to the animator. These controls adjust individual feather groups and feather curling/deformation. The high-level controls have a setting that controls the visibility of these low-level controls, this way they can be made visible when they are needed and invisible when they are not. This reduces visual clutter and confusion.

III.3.2 Interpolation Between Poses

Interpolation is the calculation of intermediate values, in this case keyed poses, between two or more previously determined values [Ford & Lehman 2002]. In most common animation systems the default interpolation between keyed poses is a smooth interpolation, meaning that the action eases into a position and rotation and eases out of the next keyed position and rotation. Some animators may want to alter this default interpolation by breaking the tangency on the animation curves, creating a sharper or slower movement, especially on terrestrial creatures. However, for flight animation where hard ground contacts are absent, this default smooth interpolation is appropriate.

III.3.3 Representation of Physical Forces

While most motion is biologically driven, controlled by the birds bone position and muscle movement, one important aspect of motion is that of dynamic forces such as wind force and air turbulence. A similar approach to that used in *The Legend of the Guardians* is used for this project; wind force can be artistically controlled on the main flight feathers and air turbulence is not given to the rig. The animator can artistically control the wind force on the main feathers via a control that curls the feathers. This allows the artist to create smooth and visually appealing arcs. Without this control the wings would look stiff, even if everything else in the wing moves in a realistic manner. The covert feathers have no such control because the force of the wind does not primarily affect them; they are more affected by wind turbulence on the bird because the covert feathers act as eddy-flaps that help to maintain lift [Ritchison 2010]. This effect of wind turbulence is better achieved through a simulation than it is through animation of the rig, so it has not been added into the rig itself.

III.3.4 Range of Motion and Control Limitation

Setting limits on a characters controls defines the range of motion the character will work within. Locking or limiting a control prevents the user from possibly using the control in such a way that would impede rig function or give undesirable results. Limits can be set on a control by setting specific attributes (such as translation and rotation) to only function within a defined range or by locking the attribute entirely so that the user cannot alter it. For this project, range of motion is only limited in cases where it would break the rig entirely. This choice was made for several reasons:

1. **To accommodate extensibility**: The rig is a general rig and does not have a specific character or set performance to tailor it to. Due to this, it is impossible to predict what will be required of it or what performances will be needed; whether it is a fully realistic performance such as that of a

real bird, or an anthropomorphic performance such as that in The Legend of the Guardians.

2. **To accommodate artistry**: The ability to have artistic control over the rig and allow for the use of traditional animation techniques.

Putting very little limit on the range of motion of the controls allows riggers to tailor the wing to their specific project. The rigger can choose which controls to limit and which to allow instead of being forced to use an arbitrary range of motion that was chosen for the purposes of this project. Where control is limited or removed entirely, the specific reasons for why for each case are explained in the implementation section.

Traditional animation techniques, originally developed for hand-drawn animation, are used to enhance a character's performance, such as the case of the owls' anthropomorphic performances in *The Legend of The Guardians*. It is important to allow for the use of those traditional techniques within limits.

Artists need rig controls that enable them to move past the motions and actions a real bird is capable of, yet still keep within the scope of possible the biological realism for many reasons. One such reason is visual clarity, such as in the case of "breaking" an elbow- or bending the joint, sometimes in a way that is anatomically incorrect- to make the movement of an arm swing look more natural [Williams 2001]. (See **Figure 8**).

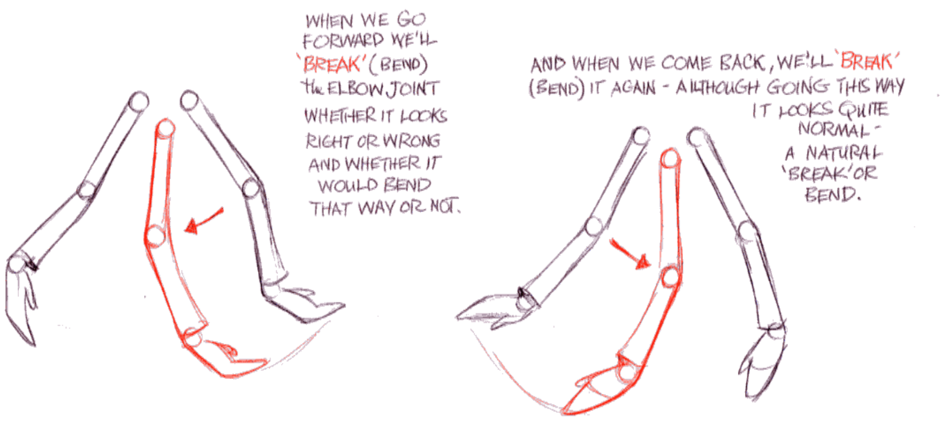
WHEN WE GO FORWARD WE'LL 'BREAK' (BEND) the ELBOW JOINT WHETHER IT LOOKS RIGHT OR WRONG AND WHETHER IT WOULD BEND THAT WAY OR NOT.
AND WHEN WE COME BACK, WE'LL 'BREAK' (BEND) IT AGAIN - ALTHOUGH GOING THIS WAY IT LOOKS QUITE NORMAL - A NATURAL 'BREAK' OR BEND.

Figure 8. Example of elbow breaking for swinging arms from *The Animators Survival Kit* [Williams 2001].

In the special case of Buckbeak the hippogriff in *Harry Potter and the Prisoner of Azkaban* the wings were able to be scaled in size. This was because when the wings unfolded they looked too small to hold the weight of the hipogriffs horse body in the air. The ability to control the size of the wings helped to increase believability even though avian wings do not grown and shrink in reality [Fordham 2004]. In other cases breaking the realism is needed for dramatic effect, visual clarity, or expressiveness such as in the *Legend of the Guardians* which required anthropomorphic performances of the owls such as speaking and gesturing [Desowitz 2010]. It is hard to determine where to draw the line between realism and artistic control and sometimes it must be determined on a case by case basis. For this reason, the controls on this rig default to provide more flexibility rather than less. Control is restricted or limited where an action would cause problems with the rig or look entirely unnatural (such as translation of a joint causing unbelievable stretching).

III.4 Results Analysis

The three main questions to answer based on the project goals are:

- Does the rig program facilitate effective wing rig creation?
- Does the rig program facilitate efficient animation?
- Does the resulting rig give realistic look and movement to the wing and feathers?

In order to answer these questions, the research method for this project is that of qualitative research and observational usability testing. The focus of this rig is to improve the realism, but this begs the question of how is realism measured? The realism relies on many factors such that it would be difficult, if not impossible to measure a single factor, such as effect of a biologically accurate rig, by quantitative means. If it were possible to ignore factors such as lighting and texturing and only focus on the movement of the rig, there would still be two factors affecting it: the skill of the animator and the biological accuracy of the rig. Even if the rig is completely biologically accurate and capable of perfect realistic movement, but in the hands of an artist who is unskilled in the craft of animation, the result will still be a loss in realism.

Due to this I've chosen using a qualitative method of analysis using participant observation and feedback which gives a more in-depth understanding of the effectiveness of the resulting rig. Three Master of Science in Visualization students at Texas A&M University, two riggers and one animator, were chosen based on having had prior experience in rigging and animation and asked to give feedback on *WingCreator* and the resulting rig. The small number of people who gave feedback was due to the limited pool of students with experience in these fields, the method of contact, and also due to time constraints on the students. Prospective students were contacted in person to ask their participation. Possibly a better method of contact that may have generated more prospective student testers is a mass e-mail request for participation over the Visualization Lab's list serve, which is sent to all students in the department . Other students were asked to participate but declined citing a lack of ability to devote the time needed to give adequate testing and feedback. A better method of contact, access to a larger pool of students with experience in these fields and lesser time constraints may have generated more participation in feedback for *WingCreator*.

To test the rig, the animators were asked to animate the rig to a short scene of a real bird to try and match the motion. The riggers were asked to test out the rig generation program and the rig. Both groups were then asked how they felt the rig and program performed, what its strengths and faults were, what they liked and disliked about it, how it could be improved, and their thoughts on the realism results and how they attempted to achieve realism using the rig. This feedback from participants gives empirical evidence to answer the three main questions regarding how well the rig succeeded in its goals and what could be done in the future to improve.

CHAPTER IV

IMPLEMENTATION

The following section discusses aspects of how the *WingCreator* program and rig was implemented, including the software and programing language used, and detailed descriptions of the feather generation and control system build tool.

IV.1 Implementation Environment

This project focuses on the supporting animations as would be used in the visual effects film industry. Therefore, one of the key components to implementation is use of the tool within 3D graphics software that is common in the industry and widely used. For this purpose, Autodesk's Maya 2011 was selected as the 3D animation software within which the tool was built. Maya has wide user base has been used on films such as *Rango* (2011), *Star Trek* (2009) and *Shrek* (2001) [Customer Showcase n.d.]. Maya is also available for multiple operating systems including Linux, Windows, and Mac. Maya uses a modular architecture, also known as node-based or hierarchical architecture, which is needed in order to implement the methods of joint driven and control driven components of the wings.

The Graphical User Interface (GUI) for *WingCreator* was created using QT Designer, a graphical program that is used to build user interfaces. This was chosen because not only is QT Designer is easy to use due to its drag and drop interface and automatic code generation, Autodesk has improved the QT integrated with Maya 2011 so that integrating a QT interface is simple, requiring only a few lines of code to do so [Maya 2011 Highlight 2010]. This makes it easier to use QT Designer to create a GUI that will work in Maya instead of having to learn how to write one from scratch using MEL or Python code.

Lastly, a code editor or text editor is needed. A regular text editor could be used for the task; however, a code editor that can recognize the programming language to highlight syntax makes the job easier. The program can be chosen solely on the preference of the person creating the rig; my preference is Notepad++ since it has syntax highlighting, and supports code folding (expanding and reducing sections of code in a document), and supports plug-ins such as the one I commonly used from creativecrash.com which allows it to recognize MEL script syntax as well as its native python programming syntax [Csaez 2008].

The choice in programming language is based largely on what programming languages the software uses. Maya 2011, supports MEL (its own native language) and integrates python, so the choice is between those two languages. Python 3.1.3 was chosen to write the code instead of MEL for several reasons. First and foremost, Python is becoming the standard programming language used in the computer graphics industry and as such it can be used by a wide audience. Second, the python syntax is clean and easy to understand, making it a good language for novices as well as experts and it also is platform independent so it can be run on any operating system that has python installed.

IV.1.1 Software Terminology

The terminology across 3D graphics software is not standardized. For the purposes of this project, I use the following terminology which is common to Autodesk Maya 2011 and referenced from *Inspired 3D character Setup* [Ford and Lehman 2002]:

- **Forward Kinematics (FK)** – Method of animating skeletons in which each bone is individually rotated.
- **Inverse kinematics (IK) –** Method of animating skeletons in which the computer solves the bending of a joint based on the position of one target object (usually an IK handle).
- **Constraint** – A direct connection of attributes on one object to the corresponding attributes of another object.
    - **Orient constraint** – Connects the orientation (rotation) attributes.
    - **Point constraint** – Connects the position (translation) attributes.
    - **Parent constraint** – Connects all attributes (translation, rotation, and scale.)
    - **Aim constraint** – Connects the aim vector of one object to follow a target object.
    - **Pole vector constraint** – Connects the pole vector of an IK handle.
- **Blendshapes** – Also known as morph targets, a "deformed" or end target version of a mesh is stored as a series of vertex positions. During animation the vertices are then interpolated between the original and target mesh.
- **Keyframe (Key) –** A frame at which an important change in an attribute (such as scale, translation, etc.) are saved to preserve and define the movement in an

animated object or character.

- **Set Driven Keys** – A single attribute is set up to control ("drive") one or more other attributes.
- **Parenting** – Different from parent constraining, parenting is the process of attaching a node above another node in a hierarchy.
- **Node**: The representation of an object or set of objects in which all information for that object or objects is referenced.
  - **Null Node**: A node containing no objects. Known in Maya as a "group" and referred to from here on as a "group node" so as not to cause confusion when referring to a group of feathers instead of a node.

IV.2 Skeletal Setup

Like any animal, the structural foundation for movement is the bones and this is also true of most biologically based rigs. Following the anatomy of a real bird, three digital bones were used to represent the main bones in a bird's wing (humerus, radius/ulna, and bones making up the manus.) Other bird rigs have used more bones in the wing in order to give it flexibility when flapping, however, these can also make the wing look unnatural, bending in too many places such as seen in the rig by Subbu Addanki [Addanki 2010].

To maintain the proportions of a real bird, an image of a Red -Tailed Hawk is overlaid with an X-ray of Red-Tailed hawk bones so that the bone position within the wing was visible as a guide. (See **Figure 9**). The image is imported into Maya 2011 and used as a guide because it allowed identification of where the joints for the rig needed to be placed. For joint placement, the user places locators at the rotation points of the joints (shoulder, elbow, wrist and tip of the hand) then the program will take the XYZ locations of those locators and build the virtual bones between them. (See **Figure 10**).

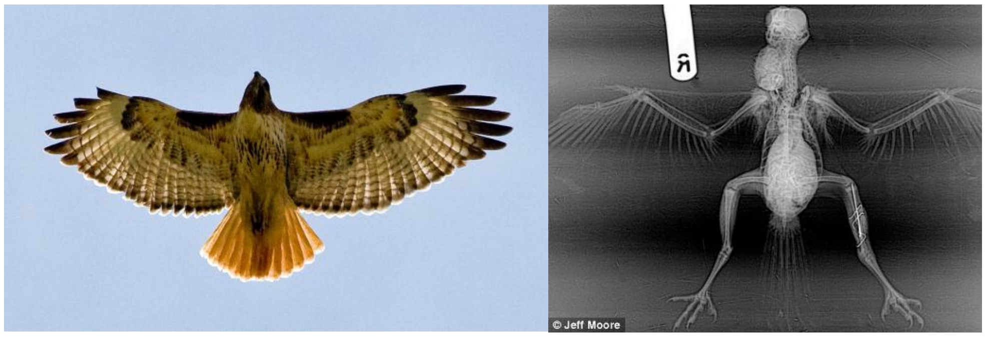

Figure 9. Red-Tailed Hawk image and X-Ray.

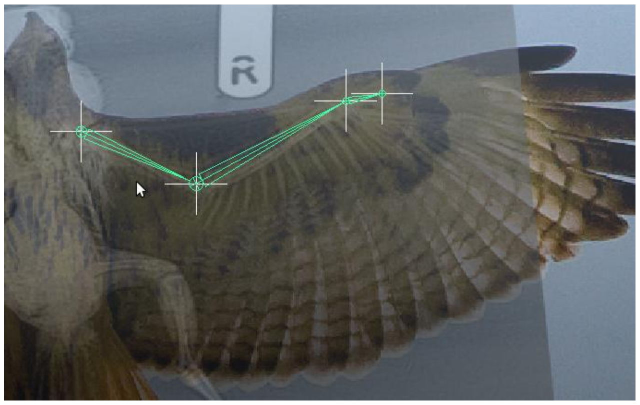

Figure 10. Resulting combined photos and skeleton.

IV.3 Feather Generation

The feathers of main concern were the primaries and secondaries (remiges).

These feathers had the widest range of motion and were the most expressive, so these were the feathers that were focused on first. Once a generation and motion system was created for these feathers, the same or similar code could be used to for generating the other feather groups and creation motion system for them.

Often when creating a wing in CG the rigger will hand place the feather geometry in the wing, such as was done for Pegasus in C*lash of the Titans*. Hand placing feathers in a wing can be time consuming. Procedurally generating the feathers over the wing can be much more efficient. The underlying anatomy is the same for nearly all birds. This makes it possible to write a script to procedurally generate those feathers for each group. The feathers in a bird's wing are not all the same, but within each group the feathers are the same shape so a single geometry for each group can be used as a base from which others can be generated. This means that the program requires a right and left version feather geometry for all six feather groups (excluding the tertiary feathers, which were similar enough for both right and left that only one geometry was needed). Using a Red-Tailed hawk as a base provided the number of wing feathers used for the initial feather generation: 10 primaries, 13 secondaries [Proctor and Lynch 1993].

Several functions are required to determine feather generation for any group of feathers: a function to generate feathers from the start to the end position of the feather group, fanning function of the group, and scaling function of the group.

- **Start to end function**: Feathers in a group are positioned in between two locators which represent the start and end positions of a bone in a real bird. These two locators are the "start" and "end" positions of a feather group. If the feathers are, for example, the secondary feathers, these feathers begin at the elbow on the anterior side of the ulna and end at the wrist. The function takes in the start and end locator's XYZ coordinates. It then divides the distance between those two locators by the number of feathers the user has entered for the group and places the feathers incrementally between those two locators. (See **Figure 11**).

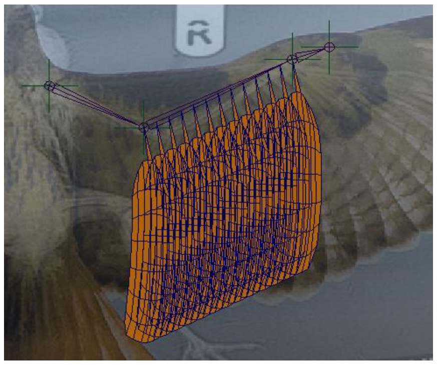

Figure 11. Start and end placement.

- **Fanning function**: Fanning is the amount of rotation an individual feather receives based on its position in the wing. Since feathers grow in a fan shape, each feather had to have a specific rotation to make that overall fan shape. (See **Figure 12**). Since the number of feathers was variable depending on the user input, a function needed to be generated that controlled the rotation of the feathers dynamically. In order to do this, feathers are generated in place and then manually rotated them into position based on the Red-Tailed hawk's feathers in the image. The rotation values for each feather were input into a

Microsoft Excel spreadsheet where a graph was generated from them. A trend line was added through those points resulting in a function for the group. The best fit of the trend line through the points varied between feather groups but was often a 1st or 2nd degree polynomial.

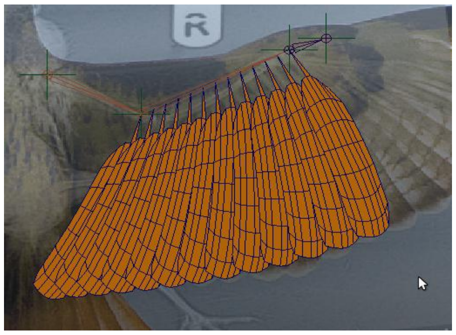

Figure 12. Fanning function results.

- **Scaling function**: Similar to the fanning function, the scaling function is the amount of scaling an individual feather receives based on its position in the wing. (See **Figure 13**).The scaling function is derived the exact same way as the rotation function is- through manual scaling to the picture, then inputting the values into Excel and generating a function. The best fit of the trend line through the points varied between feather groups but was often a 5th degree polynomial. One problem that came up was that the feathers had a tendency to "explode"- they started scaling exponentially regardless of the function. This was caused by not having enough significant digits in the function.

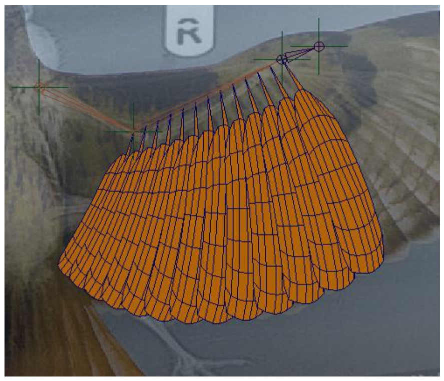

Figure 13. Scaling function results.

For example, the scaling function for primary feathers, where *y* is the resulting scale and *x* is the given number of a feather, is:

$$y = 0.688533 + 0.242191(x) - 0.153958\ (x^2) + 0.041241(x^3)$$

$$- 0.00439272(x^4) + 0.000156154(x^5)$$

When the function only had 4 significant digits, the feathers exploded- precision is important.

Once the start and end positions, scaling function, and fanning functions are determined for a group of feathers, any amount of feathers can be specified and the resulting shape will be the same. (See **Figure 14**).In the program, a separate function for each feather group was created instead of trying to make one function to generate all the feathers. This was done because each group of feathers had special considerations taken to it, therefore one *generateAllFeathers* function would be difficult to implement since it would have to take in each consideration and special case for each group. It makes the code much cleaner and easier to read when the feather generation functions are separate for each group. Primaries and Secondaries were created first, and the rest of the feather generation functions were based off those two groups. The different considerations for the rest of the feather groups are as follows:

- **Secondary coverts**: In reference to the secondary feathers, the placement for the secondary coverts is shifted dorsally so that they lay on top of the secondaries. They have the same shape as secondaries, therefore the secondaries rotation and scale functions are used, but the scale is multiplied by 2/3 to make them proportionally correct based on the Red-Tailed Hawk image.
    - **Tertials**: Tertials are a special case. Since some references list them as part of secondaries, and others list them as apart from them, artistic license is taken with the tertials and they are considered for the purposes of this project to be part of secondaries. Tertials on a real bird wrap around the base of the elbow. To achieve this effect for the rig the tertial feathers as a group have a start position at the elbow and end positions at 1/20th the length of the humerus. This measurement is an approximation of the tertial feathers from the reference images of the Red-Tailed Hawk. The scale and rotation functions are derived in the same manner as the other feather groups.
- **Primary coverts**: Similar to the secondary coverts, these were also shifted dorsally to lie above the primaries, and use the same primary rotation and 2/3 primary scale function.
- **Median coverts**: Similar to secondary coverts in placement, they were shifted dorsally to lie above the secondary coverts. The overall shape of the group is different than the secondary coverts and thus got their own scale and rotation

functions.

- **Alula**: The alula is attached to a bird's pollex, or "thumb" [Proctor and Lynch 1993]. Due to this, the function for start and end was shortened because the pollex is not the same length as the manus. It is approximated at 1/3 the length for the purposes of this project, so the feathers only generate from the wrist to 1/3 the distance to the tip of the manus. The number of alula feathers is hard-coded at 3 since birds usually have only three alula [Proctor and Lynch 1993]. It also has its own scale and rotation functions.

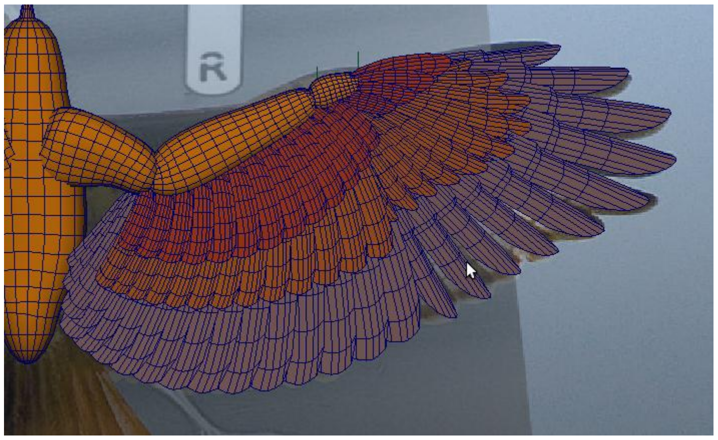

Figure 14. All feathers generated in wing.

One thing that is common to all feathers, regardless of group, is that they need to be rotated slightly around the shaft of the feather (Z axis as seen in **Figure 15**) so that they overlap like real feathers instead of interpenetrating. The rotation used to get this slight overlap is 5.0 degrees along the Z-axis.

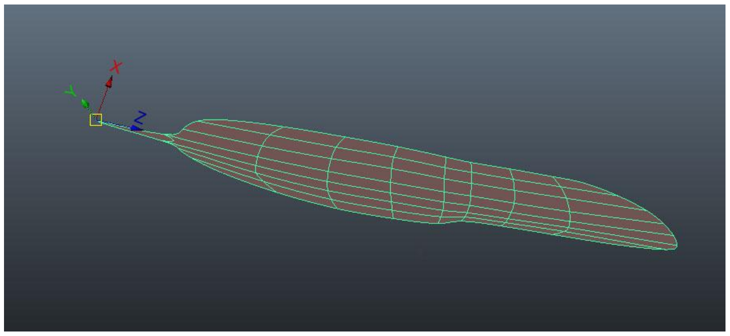

Figure 15. Rotation axes of a feather.

IV.4 Controls

IV.4.1 Feather Controls - Initial Approach

Feather controls are based on the idea that the feather groups in a bird's wing work similar to that of a fan. Originally, the feathers in a given group were individually orient constrained to a controller that, when translated, manipulated the spread and angle of the feathers. All feathers for that group were put into a group node and the group node parent constrained to the joint so it would follow the movements of the joint. (See **Figure 16**).

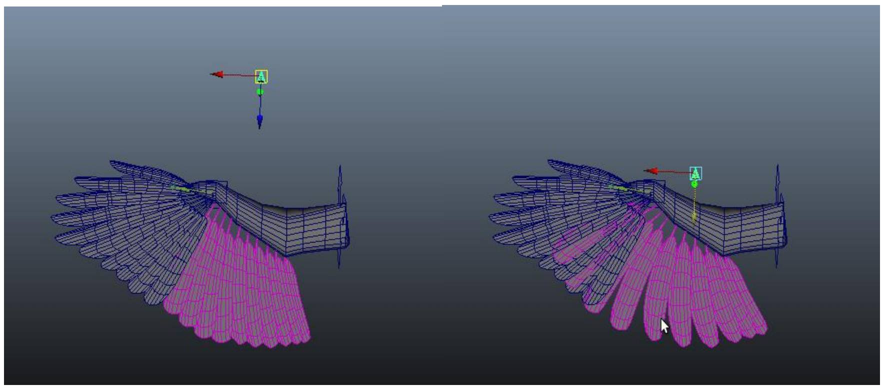

Figure 16. Initial approach- feather groups before and after translation of control.

This solution gave the desired control over the spread of the feathers, but caused several problems:

1. It was difficult to use. The control over the spread of feathers was too indirect; since the controller was not connected to or near any feathers but instead hovered away from the wing it was difficult to judge what direction and how far the controller needed to be moved to get the desired spread or fold.
2. It did not allow feathers groups to work together. One feather group would move entirely independent of the others, such as in the case of the gap seen between the primary and secondary feather groups when the wrist was bent, seen in **Figure 17**.

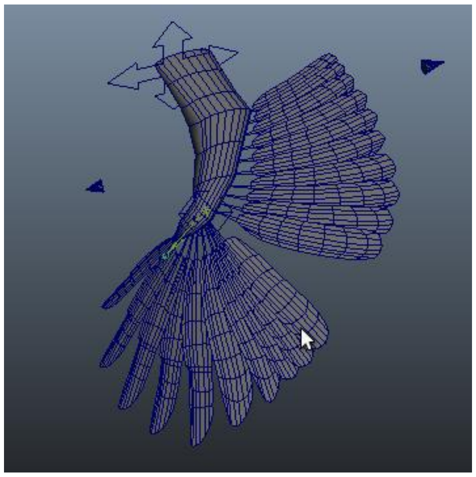

Figure 17. Feather groups splitting, showing gap between primaries and secondaries.

IV.4.2 Feather Sub-controls

The second approach, and the one that is used for the rig, is influenced by the modular bird rig created by Subbu Addanki [2010]. Addanki's rig uses controls between feather groups to manipulate the spread of the feathers. For the rig in this project there are two controls for each group, one at the each end of the group, that control the spread of the feathers. Instead of orient constraining each feather to the one control, each feather is given a weighted percentage based on its position in the wing, and orient constrained to both controls based on that percentage. For example if feather 5 is in the center of the wing, it would be constrained 50% to control A and 50% to control B, whereas Feather 2 may be closer to A, so it would be 90% to A and 10% to B. (See **Figure 18**). This is what causes the fanning motion of the feathers in a group between each control.

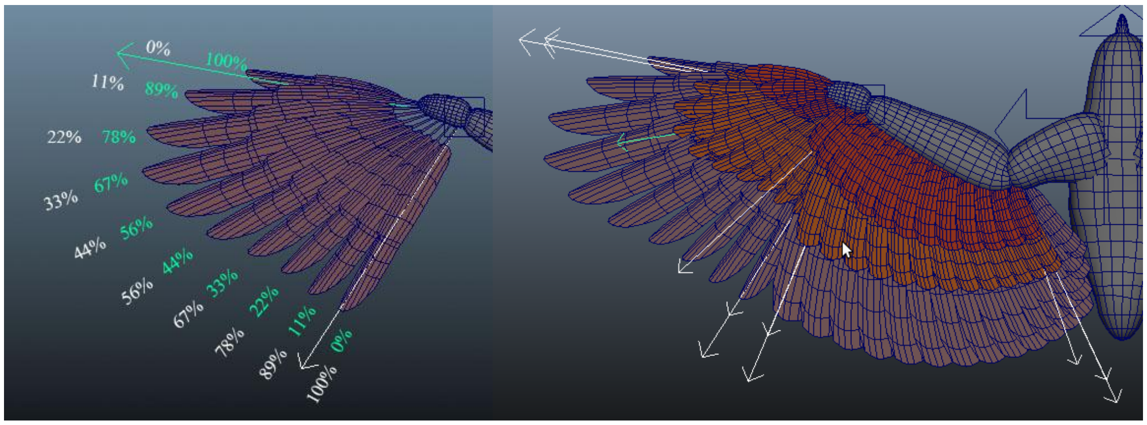

Figure 18. Feather low-level controls.

The feathers are contained in a group node with its origin at the wrist. The group node is then parent constrained to the joint that corresponds to where the feathers originate (e.g. secondaries grow from the ulna, so they are parented to the joint that corresponds to the ulna.) This setup allows the feathers in a group to follow the movement of the joint they are parented to, resulting in the feathers appearing to be attached to the bone and muscle like on a real bird.

IV.4.3 Feather Main Controls

The sub-controls for each group gave the needed flexibility and control, but animating all the controls for all the feather groups in order to animate the wing is tedious and inefficient. To address this problem, three higher level wing controls were added that provide adjustment of large sections of the wing very quickly. These three controls adjust the span of the feathers in the wing as a whole:

1. **Tip control** - controls the distal end of the wing;
2. **Middle control** - controls the midpoint of the wing;
3. **End control -** controls the portion of the wing nearest the body.

Each of the sub controls was then parented one of the three high level controls based which control it was closest to given its position in the wing. (See **Figure 19**). An attribute using set driven keys to control the visibility of the sub-controls was added to the main spread controls. The animators can turn off the sub-controls when they are not needed, which makes the control system easier to use and less confusing.

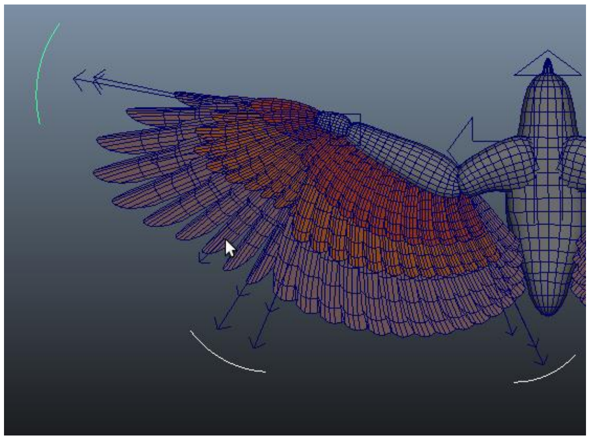
Figure 19. Feather main controls.

IV.4.4 IK and FK Setup

Control over the motion system is broken into two parallel control systems: IK and FK. This allows the animator flexibility to switch between IK mode and FK mode, depending on which they prefer to use or the requirements of the performance they are trying to create. For the two systems to work on one rig, three separate joint hierarchies are required; one for the FK, one for IK, and one as a transition. The IK and FK provide two separate solutions to the configuration of the bones. The transition joint hierarchy it can match either the FK solution or IK solutions or blend progressively between them, allowing the animator to switch modes as needed. (See **Figure 20**). The feather group nodes as well as the stand-in geometry is parent constrained to this transitional hierarchy so that the feathers and geometry will always move correctly whether the rig is in IK mode or FK mode- the tip and mid wing controls are constrained to the wrist joint and the end control to the elbow joint. In order for the wing to transition between the hierarchies, each joint in the IK and FK skeletons are orient constrained to the corresponding joint in the transition skeleton.

For the IK setup, an IK rotate plane solver, a node that uses an algorithm in the software to calculate the joint angles based on the user's manipulation of the IK solvers parts, is used. This solver connects from the shoulder to the wrist, and when the handle at the wrist is translated solver calculates the joint angles between the wrist and shoulder. The controller for the wing is placed at the wrist and the IK handle is point constrained to the controller. The wrist joint of the IK joint chain is then orient constrained to the wrist control. Point constraining the IK to the control allows the user to be able to manipulate the position of the entire wing by translating the wrist control, while orient constraining the wrist join allows manipulation of the wrist by rotating the control. Since the three main feather controls are connected to the transition joins, they will follow the actions of the joints and spread or fold when the wrist control is moved.

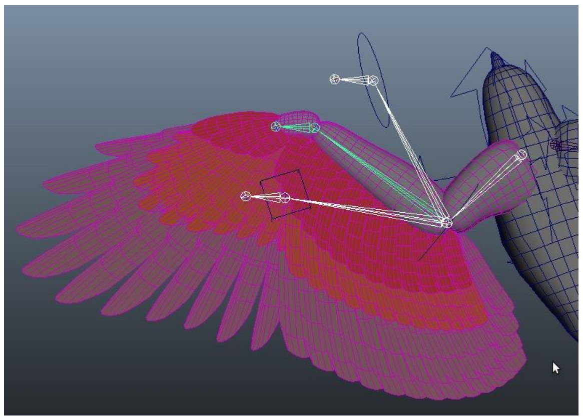

Figure 20. IK (lower white), FK (upper white), and transitional (green) hierarchies.

Sometimes when working with IK solvers, the IK will flip 180 degrees. This is due to an the fact that there are an infinite number of solutions for the joints based on the start and end positions of the IK, so to limit those numbers to a single solution a pole vector, which is an axis that defines which plane the joints will lie in, is used as a method of selecting a plane for the joints [Ford and Lehman 2002]. To prevent the IK from flipping, the IK's pole vector is pole constrained to a control located behind the elbow. This pole control must be located in the same plane as the joints and far enough behind the elbow that it does not interfere with the wing; if it is too close the control is difficult for the animator to find because it can get mixed up with character geometry or other controls. Manipulation of the pole control can change the pole vector for the IK and help position the wing in IK mode.

For the FK setup, a control was created for the shoulder, wrist, and elbow and placed at each joint. The corresponding joints are then orient constrained to the controls. Once the IK and FK setups were complete, an attribute called *IK FK switch* is added the shoulder and uses set driven keys to change the influence of the orient control; if the switch is set to 1 then the skeleton orients with the IK skeleton, if it is set to 0 it orients with the FK skeleton. The last step is to use set driven keys to alternate the visibility of the IK and FK controls with *IKFK switch*; when *IKFK switch* is set to 1 (IK mode) the visibility on the FK controls is 0 (not visible), and vice versa. This keeps the controls clean so that the controls not in use are not seen and thus not obstructing the view or getting in the animators way. (See **Figure 21**).

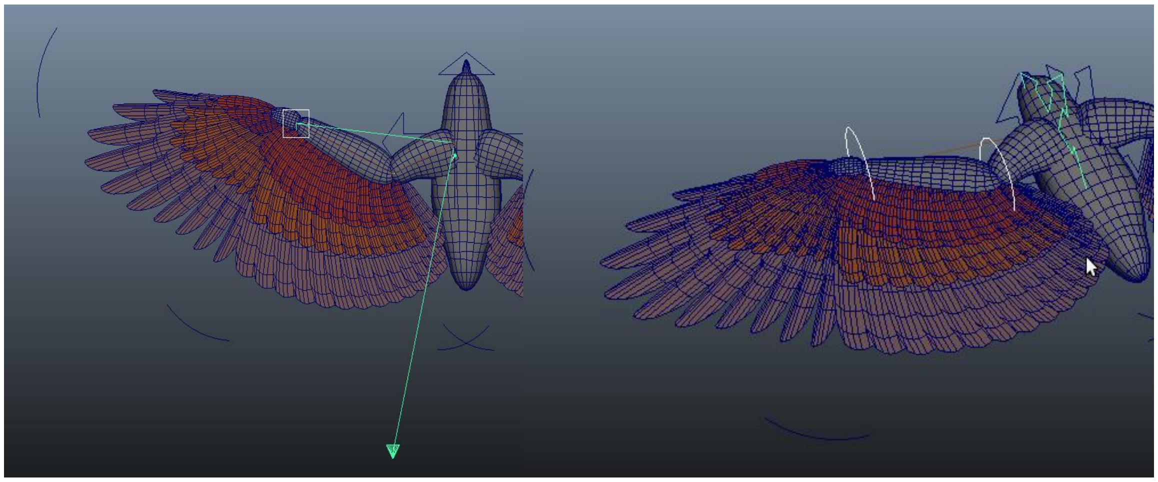
Figure 21. IK control setup and FK control setup.

IV.4.5 Automated Folding

The fold control is created as a set driven key control. An attribute is added to the Shoulder control called *AutoWingfold* with a minimum value of 0 and maximum of 10. This will allow the user to dial in a value and the wing folds that percentage (e.g. *AutoWingfold* is set to 4 and the wing folds 40% of the way) and using a scale from 0 to 10 allows for easier controlling the *AutoWingfold* attribute since the user can dial in whole integers instead of decimals.

The null groups that were parented above each control (major and sub feather controls, FK and IK controls) when it was created will be used for the automated folding. Using these null groups above the controls, the wing is rotated into the desired folded position and the X, Y, and Z rotations recorded for each. Keys are set on each control's group node at 0 for *AutoWingfold* value of 0, this sets the default wing position to be open. For *AutoWingfold* value 10, keys are set on each control's group node at the position recorded previously. If the keys were to be set on the controls themselves instead of the group node above the controls Maya would lock those controls and not allow the animator to animate that control later; putting the keys on the null node keeps the controls animatable. It's also beneficial since the animation is on the node above the control, the animator can manipulate the controls and change the position of the wing and feathers even when the *AutoWingfold* is keyed in.

IV.4.6 Locking and Limiting

The last step in the program is to lock and limit any controls that should not be altered during animation. The following are the controls that are locked or limited and the reasoning for doing so:

- The FK controls are locked in order to prevent translation. Translation of these controls would cause the bones to stretch undesirably and cause problems with the rig.
- The feather controls, high level and sub level, are locked to prevent translation. Translation of these controls would cause the controls to be pulled off the wing.
- The Feather flex and pole vector controls are locked in rotation. Rotating the control does nothing and is therefore locked to prevent confusion for the extraneous control.
- The FK, shoulder, wrist, High level feather controls, low level feather controls, feather flex, and pole vector controls all have the scale locked. Scaling these controls can cause problems with the rig such as the controls scaling, but not the rest of the rig, or the joints scaling but not the feathers.

- The IK, FK and transitional joint hierarchies and IK handles have the visibility turned off. This hides them from the animator so that they cannot be selected and accidentally animated and also reduces the visual clutter so that the animator only sees what they need in order to animate.
- The low level feather controls and feather flex controls have the visibility locked. The visibility on these is controlled through the attributes *Group Controls* and *Flex Controls* so visibility on individual controls is not needed.
- The feather flex is limited to 1.3 units in the positive or negative Y direction (e.g. up and down). This is the only control on the rig that is limited as opposed to locked due to the fact that the feathers for the feather flex are driven by blendshapes. Dialing in the blend shape over 1.3 causes the feathers to not only flex, but start to deform and stretch. The limit is set at 1.3 units in order to prevent the feathers from deforming.

IV.5 The Python Script

The python script for *WingCreator* program can be found in full form with annotations in Appendix A. The program code can be run by copying and pasting into the script editor of Maya 2011. The following digital objects are required in the Maya file in order for the program to run:

- **Feather geometry**: *L_PrimaryBase, L_SecondaryBase, L_PrimaryCovertsBase, L_SecondaryCovertsBase, L_MedianCovertsBase, L_AlulaBase,* *R__PrimaryBase, R_SecondaryBase, R_PrimaryCovertsBase, R_SecondaryCovertsBase, R_MedianCovertsBase, R_AlulaBase, TertialBase*

  - A right feather geometry and left feather geometry of all feathers for each group, centered with the tip of the quill at the origin and pointing down the positive Z axis. Only the tertial doesn't need a right and a left due to it being nearly symmetrical.

- **Locators**: *L_Wing_1, L_Wing_2, L_Wing_3, L_Wing_4, R_Wing_1, R_Wing_2, R_Wing_3, R_Wing_4*
  - These correspond to the shoulder joint, elbow joint, wrist joint, and the tip of the hand (not tip of the feathers) respectively.
- **Controls**: *CONBASE, BoxConBase, ArrowConBase, MainConBase, MoveAllConBase,* and *SubConBase* with parented controls *Wing_Flex_1 and Wing_Flex_2.*
- **Blendshapes**: *L_Primary_BLND, L_Secondary_BLND, R_Primary_BLND, R_Secondary_BLND*
  - Blendshapes must be derived from the feather geometry (for example L_Primary_BLND must be derived from L_PrimaryBase) and must NOT have history deleted. Deleting the history breaks the blendshape, causing it to shift position to the origin when keyed in.

CHAPTER V

RESULTS

To evaluate the effectiveness of *WingCreator*, two riggers and one animator were asked to critique them. Their critique and suggestions, discussed below in regards to how they relate to the goals of the project, can be used to improve *WingCreator* by possible implementation of these suggestions in future work.

V.1 Realism

In regards to realism, both riggers and animators found it difficult to critique on due to the fact that realism is hard to quantify. When one rigger was asked whether he felt that using biologically accurate feather groupings on the wing helped increase the realism he stated that he was unable to judge without the wing being animated. They may have felt unable to comment on the realism, however, they did state that using feather groups gave the rig very good artistic control over the wing and enabled them to shape the wing into visually interesting curves.

Similarly, it was difficult to judge the realism of the folded wing. The stand-in geometry for the bird was modeled as one piece of geometry and modeled in a flying position, however, a bird folds its wings usually when it's perched. The folded wing looked visually appealing, but since the bird body was in a flying position and not a perching position it was impossible to judge if the folded wing is realistic.

While realism may be hard to define and could not be commented on very well by the testers, they were able to give much more in depth commentary on other aspects of the rig including bug fixes, program use, form and control.

V.2 Effectiveness for Riggers

V2.1 Bug Fixes

Most of the problems, or “bugs” that the riggers encountered when running the *WingCreator* program were small and would be simple to fix:

- The program cannot run from the command line. It currently only runs when copy and pasted into the script editor. The riggers find this tedious and would like it to be able to run from command line by placing the python and UI files into the Maya Scripts directory.
- One rigger had a problem with the User Interface appearing off screen when the program is first run. The program needs to have the start position defined so that this is avoided.
- The program will sometimes not generate one side of the motion system for the wing when the user has changed from generating feathers on the opposite side. This is most likely due to a variable not being properly saved or passed to a function.
- The *AutoWingfold* control has rotation problems when interpolating between the open and closed positions. The interpolation should be linear, going directly from the open to the closed position, but in the middle the wing twists and rotates approximately 60 degrees around the Y axis before rotating back and settling in the closed position. This is likely due to a problem called Gimbal lock. Gimbal lock is the loss of one degree

of freedom when the two of the three rotation axes are rotated in such a way that they end up parallel and reduce the control to two-dimensions.

However, there is one large bug fixe that is more difficult to solve: the feathers on the wing tend to interpenetrate each other. There were several suggestions from the riggers on what could be causing this problem. One of the suggestions is that because the default geometry in the Maya scene file is made with planes instead of polygons this is causing part of the problem. Planes, unlike polygons, do not have any thickness so when they overlay one another they tend to disappear, making it appear as if there is more interpenetration than would normally occur with geometry that has thickness. Another suggestion is that the rotation point for the covert feather groups is placed in such a way as to cause the feathers to rotate downward when the group is manipulated, causing interpenetration. More research needs to be done to resolve the major inter-penetrations.

V2.2 Rig Efficiency and Form

Most of the critique on the program was positive. The riggers liked the generation of the wing and the ability to change the number of feathers in separate groups. The readMe file included with the program was descriptive enough that it enabled them to be able to use the program without much difficulty. The riggers also suggested to add a description of the *AutoWingfold* control, describing it as a set driven key that has an input value from 0-10 so that is it is not confused with other controls that are binary which only have a 0 (false) or 1 (true) input value. Most riggers do not know the feather structure of a bird’s wing, so the riggers found the wing diagram image included with the files was very useful. It was also suggested that the program have a button that clears the created feathers so that the user does not have to click undo in the software or manually delete them, which can be time consuming. If this could be broken up so that it could delete specific groups, it may also help the riggers understand which feather groups are which without having to rely heavily on the diagram.

V.3 Effectiveness for Animators

V.3.1 Form and Motion

The riggers and animators both appreciated the layering of the feather groups. They felt that it gave good artistic control and gave them the ability to create visually appealing wing shapes. Similarly, the feather flex control was also highly lauded. The animator felt that while the wings with feather groups were able to create appealing wing shapes, when the feather curl was added to this it made the wing shape look very smooth and natural. The feather flex enabled the animator to make beautiful arcs and curves in the wing when matching the animation of real bird. The one suggestion I received regarding form was to also add this feather flex to the covert feathers as well in order to make the wing even smoother. Currently the covert feathers do not flex with the primaries and secondaries.

V.3.2 Artistic Control

The control system had the most critiques on it, positive and negative. Overall both animators and riggers liked the high-level controls, finding them to be intuitive and easy to use, however the low-level controls they felt were lacking. Low-level controls were somewhat confusing and counter intuitive. The main critique of the low-level controls is that they are confusing and counter intuitive. They were difficult to use because the arrows looked too similar to one another. It was suggested that these arrow controls have a different shape, change them so that they are more visible and don't overlap one another. Another suggestion is that the low-level controls should work more like the high-level controls: have three controls that adjust the beginning, middle and end of the group.

The animator liked the IK control, finding that this control enabled efficient posing of the wing. She has a dislike for FK control, finding that FK control is counter intuitive. In the case of the FK on the bird wing are three controls to pose the wing and therefore three controls to manipulate into position and adjust the interpolation of in the graph editor. With the IK there was only one control which minimized the amount of work and was easier to adjust into position so she felt IK was more appropriate to use.

Another suggestion for the controls was to add more control over the blendshapes. Currently the blendshapes control the right and left sides of a group. One of the riggers suggested that there be a control for the middle of the group as well, allowing for more flexibility over the shape of the wing. He felt that there should be more ability to control the wing, even if it is not necessarily realistic because this would allow animators to be able to have more flexibility in what they can do and the shapes they can achieve.

Finally, both riggers and animators suggested that the controls be visually different. Currently the controls are all one color and therefore when animating it can be hard to differentiate between right and left controls. It would be more efficient if the controls for the left and right were different colors, making them easy to differentiate from one another.

CHAPTER VI

FUTURE WORK

The main area of future work for this thesis is that of implementing some of the feedback from the riggers and animator before putting this work out of public consumption. Suggestions such as those mentioned in the Bug Fix section would need to be addressed in order to make the tool as functional as possible before offering this tool for the public.

Other future work addresses that of extending the tool beyond its current limited range. *WingCreator* currently only creates broad winged birds like a Red-Tailed Hawk. One area of interest would extend *WingCreator* such that it would be capable of creating other wing shapes, such as long and narrow or wide and short. This would require creating different mathematical functions for the different types of wings. Similarly, the program could be extended to include long winged birds such as an albatross. These birds have two extra groups of feathers called the humeral and humeral coverts that would need to be accounted for by the program [Sibley 2002]. The animator that reviewed the rig suggested that more automation would also extend and improve the functionality of the rig. Currently the only automation is the automatic wing fold. Other automatic animation and poses could include something like a flap cycle or flap cycle poses.

Lastly, the tool could be more than just for rigging the wings of a bird. It could be part of a series of tools that facilitate the creation of a complete bird rig. Feather groups such as the scapulars were omitted because they interacted with the body of the bird, causing them not move unlike fans or blinds. These groups could be accounted for in a more complete bird rig and would allow users to make not only biologically accurate wings, but also biologically accurate full birds.

CHAPTER VII

CONCLUSION

While the level of realism from this method of wing rigging could not be determined, feedback from the riggers and animator indicated that the *WingCreator* rig tool is useful for riggers and animators. It creates biologically accurate rigs with realistic wing form and motion that facilitate efficient animation. The resulting rig has an intuitive control system and a high level of artistic variability for the animator making it a good option for those in the film industry. Future work could also help to expand this tool beyond its current limitations to make it even more versatile, adding in more automation, different types of wing shapes, and even extending the tool to include a full bird rig.

Once placed in the public domain, this tool will be of use to artists who are interested in creating more biologically accurate bird wing forms and motion. The methodology of the tools creation will also add to the currently slim body of knowledge for creating realistic avian wings.

REFERENCES

ADDANKI, S. 2010, November 30. Python Scripting : Auto Bird Rigging Show Reel -By Subbu. http://www.youtube.com/watch?v=FqG8Ef93jz0. (Retrieved October 16, 2010).

AITKEN, M., BUTLER, G., LEMMON, D., SAINDON, E., PETERS, D., & WILLIAMS, G. 2004. The Lord of the Rings: the visual effects that brought middle earth to the screen. In *ACM SIGGRAPH 2004 Course Notes* (SIGGRAPH '04). ACM, New York, NY, USA, Article 11.

BANGAY, S. 2007. Animated feather coats using field lines. In *Proceedings of the 5th International Conference on Computer Graphics, Virtual Reality, Visualization and Interaction in Africa* (AFRIGRAPH '07), Stephen N. Spencer (Ed.). ACM, New York, NY, USA, 169-176.

CHEN, Y., XU, Y., GUO, B., & SHUM, H. 2002. Animated feather coats using field lines. In *Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques* (SIGGRAPH '02), Stephen N. Spencer (Ed.). ACM, New York, NY, USA, 630-636.

Customer Showcase. n.d. http://usa.autodesk.com/adsk/servlet/index?linkID=10164097&id=10262386&siteID=123112. (Retrieved September 30, 2011).

CSAEZ. 2008, November 9. MEL language Definition for notepad++ 1.0.1. http://www.creativecrash.com/downloads/applications/syntax-scripting/c/mel-language-definition-for-notepad-. (Retrieved September 30, 2011).

DESOWITZ, B. 2010. Take Flight with the Guardians. Animation World Network. http://www.awn.com/articles/3d/taking-flight-guardians/page/1%2C1. (Retrieved September 30, 2011).

FORD, M. & LEHMAN, A. 2002. *Inspired 3D Character Setup.* Florence, KY: Course Technology PTR.

FORDHAM, J. 2004. Harry Potter and the Prisoner of Azkaban: Something Wicked This Way Comes. *Cinefex.* 99, 55-58.

GILL, F. B. 1995. *Ornithology* (2nd ed). New York: W.H. Freeman and Company.

HIEBERT, B., DAVE, J., KIM, T., NEULANDER, I., RIJPKEMA, H., & TELFORD, W. 2006. The Chronicles of Narnia: The Lion, The Crowds and Rhythm and Hues.

In *ACM SIGGRAPH 2006 Courses* (SIGGRAPH '06). ACM, New York, NY, USA, Article 1.

HUTCHINGS, A. 2007. Bird Wing Development for Computer Graphics. http://ncca.bournemouth.ac.uk/gallery/view/398/Bird_Wing_Development_for_Computer_Graphics. (Retrieved October 16, 2010).

LASSETER, J. 1987. Principles of traditional animation applied to 3D computer animation. In *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques* (SIGGRAPH '87), Maureen C. Stone (Ed.). ACM, New York, NY, USA, 35-44.

LEANING, J., & FAGNOU, D. 2010. Feathers for mystical creatures: creating Pegasus for *Clash of the Titans*. In *ACM SIGGRAPH 2010 Talks* (SIGGRAPH '10). ACM, New York, NY, USA, Article 52 , 1 pages.

LIGHTFOOT, N. 2004. 12 Principles. Animation Toolworks. http://www.animationtoolworks.com/library/article9.html. (Retrieved September 30, 2011).

LUCAS, A. M., & STETTENHEIM, P. R. 1972. Avian Anatomy- Integument, pts. 1, 2. Agriculture Handbook, 362. http://www.ummz.umich.edu/birds/resources/anatomy/feathers/wingfeathers.jpg. (Retrieved September 30, 2011).

Maya 2011 Highlight – QT User Interface Overview (plus GDC). 2010. Area: Digital Entertainment & Visualization Community. http://area.autodesk.com/blogs/stevenr/maya_2011_highlight_qt_user_interface. (Retrieved September 30, 2011).

MCLAUGHLIN, T. 2006. Taxonomy of digital creatures: defining character development techniques based upon scope of use. In *ACM SIGGRAPH 2006 Courses* (SIGGRAPH '06). ACM, New York, NY, USA, Article 1.

PAIXAO, M. 2010, November 29. Wing Setup Example. http://www.youtube.com/watch?v=7xThiBnEyPs. (Retrieved October 16, 2010).

PLAETE. J. 2008, August 8. Wing Feather Tool Version II. http://joplaete.wordpress.com/2008/08/08/wing-feather-tool-version-ii/. (Retrieved October 16, 2010).

PROCTOR, N.S., & LYNCH, P.J. 1993. *Manual of Ornithology.* New Haven, CN: Yale University Press.

RITCHISON, G. 2010. Ornithology Lecture Notes- Flight I.

http://people.eku.edu/ritchisong/554notes2.html. (Retrieved October 8, 2011).

ROBERTSON, B. 2010. In Fine Feather. Computer Graphics World. http://www.cgw.com/Publications/CGW/2010/Volume-33-Issue-9-October-2010-/In-Fine-Feather.aspx. (Retrieved September 30, 2011).

SEDDON, D., AUFINGER, M., & MELLOR, D. 2008. Rendertime procedural feathers through blended guide meshes. In *ACM SIGGRAPH 2008 talks* (SIGGRAPH '08). ACM, New York, NY, USA, Article 76, 1 pages.

SIBLEY, D.A. 2000. *The Sibley Guide to Birds.* New York: Knopf.

SIBLEY, D.A. 2002. *Sibley's Birding Basics.* New York: Knopf.

STREIT, L., & HEIDRICH, W. 2002. Generating feather coats using Bezier curves. In *ACM SIGGRAPH 2001 conference abstracts and applications* (SIGGRAPH '02). ACM, New York, NY, USA, 188-188.

THOMAS, F. & JOHNSTON, O. 1981. T*he Illusion of Life*. New York, NY: Disney Editions.

WEBER, A. J., & GORNOWICZ, G. 2009. Collision-free construction of animated feathers using implicit constraint surfaces. *ACM Transactions on Graphics.* 28, 2, Article 12 (May 2009), 8 pages.

WILLIAM. 2006. Straight Ahead Action and Pose to Pose. Blender's Wiki. http://wiki.blender.org/index.php/Doc:Tutorials/Animation/BSoD/Principles_of_Animation/Principles/Straight_Ahead_Action_and_Pose_to_Pose. (Retrieved September 30, 2011).

WILLIAMS, R. 2001. *The Animator's Survival Kit.* New York: Faber and Faber Inc.

WU, J., & POPOVIC, Z. 2003. Realistic modeling of bird flight animations. In *ACM SIGGRAPH 2003 Papers* (SIGGRAPH '03). ACM, New York, NY, USA, 888-895.

Supplemental Sources Consulted

MILLER, A. 1941. The Significance of Molt Centers among the Secondary Remiges in the Falconiformes. *The Condor.* 43, 2, 113-115.

O'NIELL, R. (2008). *Digital Character Development*. Burlington, MA: Morgan Kaufmann.

TRAIL, P.W. 2001. Wing Feathers. *Identification Notes for Wildlife Law Enforcement B-01-1*. Ashland, OR: National Fish & Wildlife Forensics laboratory.

WRIGHT, WESTENHOFER, B., BERNEY, J., & FARRAR, S. 2006. The visual effects of The Chronicles of Narnia: the Lion, the Witch and the Wardrobe. *Computer Entertainment*, 4, 2, Article 4 (April 2006).

APPENDIX A

The Python code for *WingCreator* tool is included as a separate file.

VITA

Name:	Heather Vernette Howard
Address:	Visualization Dept. c/o Prof. Tim McLaughlin Texas A&M University College Station, TX 77843-3137
Email Address:	falconsong@falconsongstudios.com
Education:	B.A., Architecture, Texas A&M University, 2007 M.S., Visualization, Texas A&M University, 2011