12-2018

Procedurally Generating Biologically Driven Bird and Non-Avian Dinosaur Feathers

Jessica Renee Baron
Clemson University, jrbaron@gmx.com

Procedurally Generating Biologically Driven Bird and Non-Avian Dinosaur Feathers

A Thesis
Presented to
the Graduate School of
Clemson University

In Partial Fulfillment
of the Requirements for the Degree
Master of Science
Computer Science

by
Jessica Renee Baron
December 2018

Abstract

A key element in computer-graphics research is representing the world around us, and immense inspiration may be found in nature. Algorithms and procedural models may be developed that can describe the three-dimensional shape of objects and how they interact with light. This thesis focuses particularly on bird and other dinosaur feathers and their structure. More specifically, it addresses the problem of *procedurally generating biologically driven geometry for modeling feathers in computer graphics.* As opposed to previously published methods for generated feather geometry, data is derived from a myriad of real-world specimens of feathers and used in creating graphical models of feathers.

Modeling feathers is of interest both for media production and also for various fields of research such as ornithology, paleontology, and material science. In order to create realistic, computer-graphics feathers, the anatomy of feathers is analyzed in detail with the aim of understanding their structure and variation in order to apply that understanding to modeling. Data concerning the shape of actual feathers was collected and analyzed to drive attribute parameters for modeling accurate synthetic feathers, during which methods for generating geometry informed by the data were investigated. Synthesized image results, capabilities, limitations, and extensions of the developed techniques are presented.

Acknowledgments

I wish to thank the following people and institutions for their help and guidance throughout my feather study: Paolo Selva, Olivier Gourmel, Matthias Baas, and Jen Gillespie at Weta Digital in Wellington, New Zealand; Arnulph Fuhrmann, Stefan Grünvogel, and Ursula Derichs at Technische Hochschule Köln in Cologne, Germany; the Carolina Raptor Center in Huntersville, North Carolina, USA; and the Bob Campbell Geology Museum in Clemson, South Carolina, USA.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

The motivation for modeling feathers comes from three main areas: Feathers are a source of inspiration for humans, desired in production entertainment, and a source of scientific discoveries. Human beings find much inspiration through nature, and this holds especially true in the realm of computer graphics where accurately representing the real world is a core goal. Beyond graphics, feathers and birds have frequently been incorporated with human life especially through intrigue and literature, art and adornment, flight and insulation, and more [37], and entire cultures have focused heavily on birds and prizing feathers, such as the Pacific-Islander Maori of New Zealand [43].

In the entertainment production industry, feathered creatures are often characters of interest including birds and fantasy beings either realistic or artistically stylized. Several examples include the realistic owls from *Legend of the Guardians: The Owls of Ga'Hoole* (2011) [40], Pegasus from the film *Clash of the Titans* (2010) [44], the rodent-like Niffler and other feathered creatures from the movie *Fantastic Beasts and Where to Find Them* (2016) [38]. Beyond entertainment, computer-generated feathers can aid interdisciplinary, scientific inquiry through visualization. Active areas of interdisciplinary research include the origin of birds and feathered dinosaurs in ornithological and paleontological studies as well as biological and material-science research concerning the strength, aerodynamic, and insulating properties of feathers due to their unique structure [37] [41]. Visualization through rendering feathers may assist in these investigations and understanding the properties of feathers.

1.2 Overview of Approach

This thesis incorporates the biology of feathers into a parameterized and editable, computer-graphics model of 3D geometry representing real-world inspired and artistically customized feathers. The biological structure of feathers contains several main components - the shaft, barbs, and barbules - that vary across the different types of feathers, stages of growth, and systematic position of birds. Chapter 3 on related work presents studies in biology on the structure of feathers and in visual computing on defining the main shape of a generated feather through procedural modeling and parametric curves. Feathers for film and game productions have been created for proprietary, studio software primarily following similar curve-based approaches to those taken to generate hair and fur. These solutions are tailored to each studio's production pipeline from geometric modeling to final rendering and are focused on editability and artistic control to drive desired appearances. In this thesis, some work by studios that have published regarding feathers is discussed in chapter 3, and aspects of developing new feather-generating software at the visual effects studio Weta Digital are also described in chapter 4. Previous academic and production work primarily focused on the general look of a feather without basis on real-world data. This thesis presents novel acquisition and derivation of data from feather specimens to parameterize and drive the shape of procedurally generated feathers. A publicly available database and regional resources helped secure initial feather data, and computer-vision techniques were applied for extracting additional characteristics.

This work attempts to model feathers, such as for birds or non-avian feathered dinosaurs, in a novel and more accurate manner by incorporating data from biological samples. Work presented here includes designing and building a production tool for Weta Digital, collecting and analyzing data on feathers, and procedurally generating feather geometry in a data-driven manner.

Chapter 2

Background

Related material for understanding the contributions of this thesis is provided and categorized into biological and visual-computing areas of study. The biological aspects center on feather structure, and the computing section approaches working in both 2D and 3D spaces through image analysis and final image rendering.

2.1 Biological Background

This section provides an overview of feathers in the natural world, discussing where they are present, which academic fields are involved with feathers, the anatomy of an individual feather, the variation in the main anatomical components on a feather, and the data on feathers that is currently available.

2.1.1 Feathers in Birds and Other Dinosaurs

Feathers cover the bodies of birds, and they vary greatly in size and shape depending on their placement, stage of growth, and the type of bird. They are found on no other living organisms, although in 1861, the first complete specimen of *Archaeopteryx* was discovered. This finding began discussion of the evolutionary bridge between dinosaurs and birds as *Archaeopteryx* had the skeleton of a reptile and the feathers of modern birds [37]. Further discoveries have been made concerning the presence of feathers in dinosaurs, investigating their shape, functionality, and coloration [52] [45].

2.1.2 Interdisciplinary Study of Feathers

Beyond computer graphics, feathers are an actively present subject of interest in other scientific, academic fields, namely ornithology, paleontology, physics, and material science. Ornithology is the zoological study of birds and has a voluminous literature on feathers. Many paleontological studies focus on feathers in the context of the origin of birds and their evolution. Physics work concerning feathers especially considers their aerodynamic properties, and the strong and insulative properties of feathers influences research in material science [37].

2.1.3 Feather Anatomy

The anatomy of a feather primarily consists of three branching components: the shaft, barbs, and barbules. The *shaft* is the central structure of a feather, off of which extend thin, tube-like *barbs*. The part of the shaft that is connected to the barbs is called the *rachis*, and the bare part on the proximal portion of the feather (nearest to where the feather attaches to a bird's body) is the *calamus*, or quill [41]. From the barbs branch smaller *barbules*, which may or may not possess hooklets, or *barbicels* [24]. Barbules are often visible to the naked, human eye, but barbicels are microscopic. The *ramus* of a barb is the central structure from which barbules extend, but in this thesis the structure is included in the term barb. Barbs with barbules that hook together (i.e. have barbicels) are called *pennaceous barbs* and form the *vanes* of a feather; these structures appear as smooth surfaces extending almost perpendicularly from the shaft and can be asymmetric in width and dihedral angle from the shaft [61]. Barbs with barbules that lack barbicels are *plumaceous barbs*; these barbs have a soft, fluffy appearance as they loosely lie beside each other. The plumaceous barbs nearest to the calamus collectively form an *afterfeather* in some species. The anatomy of a feather as presented here is described throughout resources in ornithology [24] [68] [41] [61] [29] [37] [67] [3]. Figure 2.1 is an illustration of a feather composed primarily of pennaceous barbs created by Emily Nastase, used with permission, and the labels modified [49].

Feathers are composed of keratin, the same protein that composes human hair. The appear-

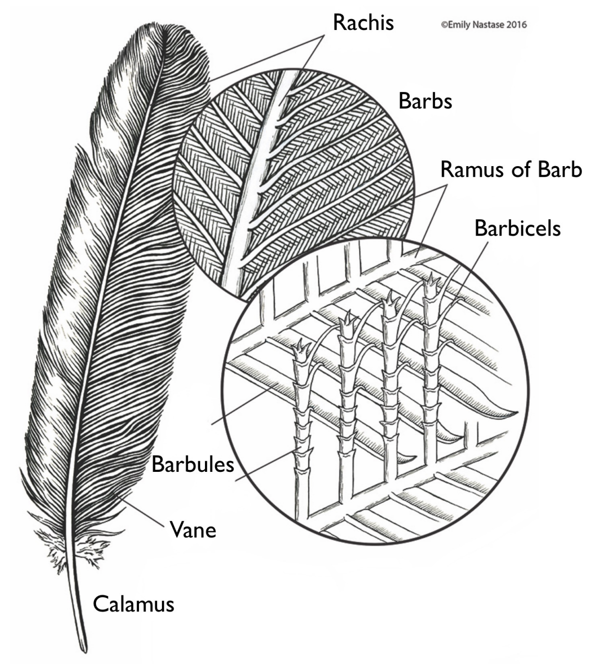


Figure 2.1: Scientific illustration of the structure of a pennaceous feather modified from [49].

ance and coloration of feathers is due to pigment in the tube-like structures of the shaft, barbs, and barbules. The reflection and refraction of light on the barbules is affected by the layers of keratin and melanin that compose their structure, resulting in optical phenomena such as iridescence and the bright violets, blues, and greens through amplification in feathers [37] [67] [62].

2.1.4 Variation in Feather Structure

The main components of a feather discussed above vary based on the type, stage of growth, and bird species. Alterations in the vanes and afterfeather are discussed in terms of the barbs. The observations presented lead to later decisions on building and parameterizing a computer-graphics feather model.

2.1.4.1 Feather Types

The six main types of feathers are flight, contour, down, semiplume, filoplume, and bristle, each of which varies in their function, placement on body, and anatomical components such as the distribution of pennaceous or plumaceous barbs [61]. This study primarily focuses on flight feathers.

Refer to figure 2.2 from [61] as a diagram of these types for side-by-side comparison.

Flight feathers are the largest and most varied in shape. These include wing feathers (*remiges*), tail feathers (*rectrices*), and *coverts*. *Primaries* and *secondaries* are specific types of *remiges*. Primaries are attached to the distal portion of a bird's wing called the carpometacarpus located furthest from the body. There are typically 10 to 12 primary feathers per wing, labeled with descending numbers ending with 1 from most distal to least distal. Secondaries follow the primaries towards the body and attached to the ulna, what would be equivalent to the forearm on a human. The number of secondaries depends on the bird's body size and the wing length; this value can range from a minimum of 6 (on a hummingbird) up to 32 (on an albatross) [61]. Rectrices, or tail feathers, are numbered in pairs from center outward. On average there are 12 rectrices, but the count can range from 8 to 32. The narrower, leading vane faces away from the tail's center. Coverts are a type of flight feather that covers the base of the wing and tail feathers for protection [67].

Contours produce the majority of visible feathers on a bird's body. *Down* feathers provide insulation, are layered beneath other types of feathers, and are unevenly distributed throughout a body; this distribution varies per group of bird. *Semiplume* feathers form a layer between contour and down feathers and are mainly used for insulation [61]. *Filoplumes* are found just past contour feathers, and they are believed to be used like whiskers for judging feather displacement and airspeed. *Bristle* feathers are similar to mammal whiskers in providing sensory information [61]. Additional types of feathers that do not belong to the six main categories include ornamental feathers with unique structure with no apparent function beyond display, particularly in birds of paradise, and hairlike feathers in flightless birds such as kiwis and emus [60].

Feathers grow from a bird's skin in set tracts called *pterylae*. During the growth of new feathers, an outer sheath emerges from each follicle of these tracts, called a pin feather. The barbs and barbules are curled around a growing shaft within this structure and do not unfurl until the bird grooms off the outer sheath when it dries. Upon unfurling, vanes form when the barbules of the pennaceous barbs connect.

There exists variation of feather structure across bird genera and species. The following are several examples of such differences. Penguins and ostriches have evenly distributed pterylae over their bodies; evenly feathered birds are uncommon, and this might have been a characteristic of early birds [61]. Flightless birds such as kiwis lack barbicels on their barbules, even on the pennaceous barbs. The barbs on waterfowl connect to make a tighter, flat surface along a portion of the vane

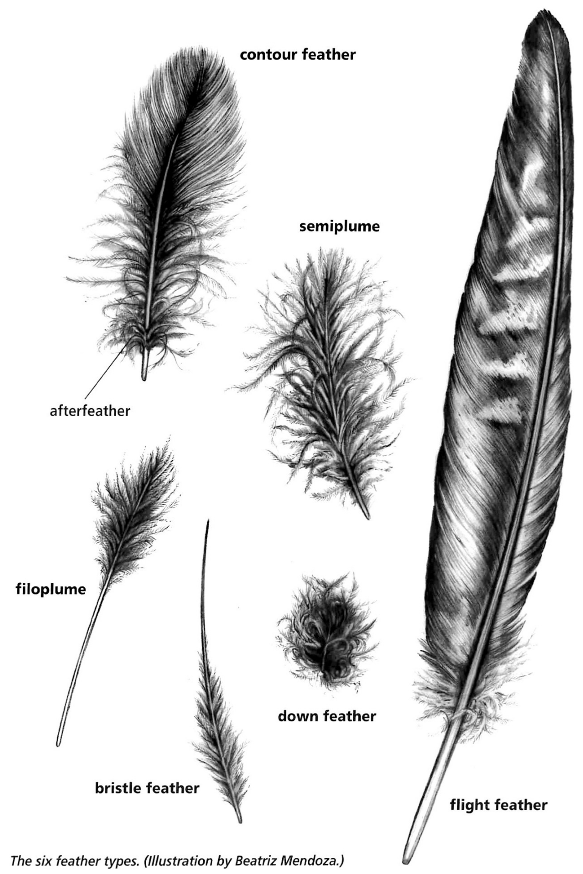


Figure 2.2: Scientific illustration of the types of feathers.

to keep it from splitting. The facial-disk feathers and distal ends of flight-feather barbs of owls have large and long barbules [67] [60].

2.1.4.2 Variations in Shaft

This section highlights changes in the shape of the shaft based on the different types and stages of feathers mentioned. Concerning the feather type, all flight, contour, and semiplume feathers have a supportive shaft. Most primaries and rectrices are the longest feathers on a bird's body, although the leading primary feather, the most distal to the body, is considerably shorter than the other primaries; it may be concealed by wing-covert feathers and thus is sometimes misidentified. Coverts for the greater primaries and secondaries are easily identifiable by a kink in the shaft curve where calamus meets rachis; this shape allows the covert's calamus to attach directly to its paired primary or secondary's calamus [61]. Both filoplume and bristle feathers have stiff, hairlike shafts.

Down feathers have a small or nonexistent feather shaft. When observing the flexibility of shafts, flight feathers are very stiff, and down and contour feathers are flexible, visible through their curling shape when not pressed among neighboring feathers. During the various stages of growth, the shaft remains at the center of a feather. It changes length over time as blood is channeled through the tube until the feather has finished emerging from the follicle.

2.1.4.3 Variations in Barbs and Barbules

In this section, changes among feathers concerning vanes, included as collections of barbs, pennaceous and plumaceous barbs, and barbules are discussed. Primaries are distinctly asymmetric about the shaft; the *anterior vane* of a primary is noticeably more narrow than the other and indicates leading edge of the wing, facing externally. The *posterior vane*, or trailing vane, is proximal to the body and is wider than the anterior vane. Secondary feathers have a rounder curvature to the calamus, the vanes form a wider, more rounded tip, and the vane widths are almost symmetric. Contour feathers are usually symmetric about the shaft; they consist of both pennaceous barbs that compose the vanes and an afterfeather of plumaceous barbs. Down and semiplume feathers contain solely plumaceous barbs [61]. Filoplumes have no barbs or a little at the distal tip, and bristles possess barbs only at the base of the shaft.

The age of a feather also affects the shape of the barbs. The amounts of barbs and barbules on a feather are determined before the feather is fully grown. Also, juvenille feathers may appear to be down feathers as their pennaceous barbs are messy and do not have barbules connecting yet, but these differ from adult down of true plumaceous barbs. Length and the tapering of the tips of feathers varies by feather type, bird species, and ontogeny, the development of an organism. As an example of age, the feather tips of juvenille crows are rounded rather than squared as they are on adult crows [47].

2.1.5 Biological and Statistical Data

As discussed, there exists great variation in feathers across types and species. Concerning a feather's shape, the silhouetting curves, the angles between barbs and the shaft, the length of barbs and barbules, the widths of vanes, and the thicknesses of shaft, barb, and barbule tubes are several core areas of where variation can be observed.

There is a lack of computer-graphics and -vision research on determining these variations beyond work that targets only the feather-down and poultry industries. For example, computer-visions studies exist on denoising microscopic images of down feathers for China's down industry [70] and using pattern recognition for the separation of chicks by sex [66]. Despite the absence of visual-computing research on the diversity of feathers, several very specific, biological studies exist where measurements are taken from the particular datasets collected. These studies include observations on only primary feathers but those across 60 bird species [32]; the covert and rectrix feathers of parrots [33]; the contour feathers of a couple types of tit [21]; dihedral barb and barbule angles in hummingbird feathers [36]; on barb growth angles based on the rachis cylinder, barb cross-sections, and resulting barb curves [53]; and on water repellency of feathers, measuring barb diameters and spacing between barbs along the rachis [55].

This thesis uses feather images from The Feather Atlas, a research project from the US Fish and Wildlife Service Forensics Lab [3]. The database consists of primarily flight feathers (primaries, secondaries, and tail feathers) searchable by bird order, family, and species. Each entry contains all of the primaries, secondaries, or rectrices in an image aligned on a grid. The measured length of each feather is also included. The scans are used in this study to extract shape variance in the outer edges of feathers from a large variety of species.

2.2 Visual-Computing Background

This section presents a brief overview of computer-graphics and -vision topics covered in the thesis, first discussing the motivation behind feathers in computer graphics, followed by an overview of rendering, procedural modeling, curves, and image analysis.

2.2.1 Applications and Motivation

Motivation for having feathers in graphics is driven by desired applications in areas such as visualization, production, anthropology, and material science. The scientific visualization of feathers can be useful in biological, zoological, ornithological, and paleontological research. Productions often require feathers as assets in film, show, and video game media where they may be as realistic as possible or artistically stylized. Using accurately detailed and modeled feathers in material and physical sciences may be helpful in understanding the fascinating properties of real-world feathers such as water resistance and strength.

2.2.2 Modeling for Rendering

The process of transforming a virtual, 3D environment into a visible, 2D image is called *rendering*. The necessary components to simulate the rendering process are cameras, lights, geometry or objects, and materials on those objects. *Modeling* is the generation and data representation of the geometry to use for rendering. This thesis is primarily focused on modeling the structure of feathers in 3D space based on collected and extracted real-world data.

2.2.2.1 Basics of Rendering

As mentioned, rendering involves simulating lights, cameras, geometry, and materials. *Cameras* frame a portion of the 3D environment as a single scene and provide the 2D image of pixels to which the scene is ultimately mapped. The purpose of simulating *lights* is to illuminate a scene from a 3D environment, and they may be modeled as sources emitting light, wavelengths and particles of light, or both. *Objects* populate a scene and need a geometric representation, usually as collections of either polygonal primitives or curves. When light hits an object, that part of the object reflects or refracts a color based on the object's *material* which describes how it reacts to light; computer-graphics materials in rendering are often derived from real-world optical responses. The remainder of this section focuses on the primitive representation of the objects' geometry.

2.2.2.2 Polygonal Primitives

One way to represent geometrical objects is with polygonal meshes. In this sense, a mesh is a collection of connected, primitive shapes, or *faces*, that form the surface of an object. By the time a scene is to be rendered, these primitives are usually tessellated to their most basic 3D form as connected *triangles*. Quadrangle meshes are common in modeling and simulation applications but are often triangulated for rendering. Polygons with more than four edges, n-gons, are uncommon due to complexities.

How the triangles are connected in creating edges among the vertices of a mesh is the called the *topology* of the mesh. There exist constraints on the topology to ensure the mesh has no gaps and a clear distinction between the “inside” and “outside” of the mesh. The vertices and triangles of a mesh have normal vectors indicating the direction they face. Normals are a key component in calculating the surface reflectance upon light interacting with the surface. Ultimately, the individual triangles are shaded one fragment of each triangle at a time in the rendering process based on the camera, light, and material properties of the scene.

2.2.2.3 Subdivisions

Often polygonal meshes do not appear smooth when they are rendered, but rather individual facets are visible in the shaded result. One common technique to avoid this is to smooth the mesh via subdividing. *Subdivision surfaces* would result from passing a control polygonal mesh through a subdividing algorithm, relocating and adding vertices and edges of the mesh into more faces. This smoothing often occurs in preparation for rendering, creating more triangles to shade but those triangles appear more closely oriented to one another. Catmull and Clark first presented the recursive algorithm to subdividing polygonal surfaces in 1978 which is widely used today [22]. An additional technique for smoothing the ultimate appearance of a mesh is manipulating the normals of the faces to reduce the rendered facets.

2.2.2.4 Curves and Curved Surfaces

Rather than storing vertex and connection information for many triangles, particularly for smooth, subdivided surfaces, curves and curved surfaces provide a more compact geometry representation as parametric functions. When a curve or surface is to be rendered, it is often tessellated into a series of triangles. Such compact representation allows for scalability, meaning that as little as two triangles but up to thousands or more can be produced by the functions. Instead of tessellation, the curve functions may also be sampled directly in some implementations.

2.2.3 Procedural Modeling

Procedural modeling is the process of generating geometry by a set of rules, and parameters typically accompany the rules in order to generate a variety of results. Lindenmayer systems (L-systems) are often used for procedural modeling. They are rewritable grammars used for a variety of purposes, namely those recursive in their nature [54] and have many applications. Examples of where they have been applied include generating and updating 2D or 3D curves and modeling vegetation, cell growth, and feathers [54] [58] [48] [15] [25]. L-systems were considered initially for this work but were not ultimately used as they imposed constraints that were not practical for the two implementations discussed here.

2.2.4 Curves

A *curve* is an infinitely large set of points over an interval in space where any point of the curve has two neighbors except possible endpoints which each have one neighbor. (Curves can be infinite - with one endpoint - or a loop - with no endpoints - but this thesis uses curves with two endpoints.)

Curves can be represented *implicitly* or *parametrically* by functions. Implicit functions test if a given point is on a curve or not. Parametric functions require only one input, a time or progress along the curve which typically can range between 0.0 (the start of the curve) and 1.0 (the end of the curve) and outputs a corresponding location on the curve. Parametric representation is more flexible for modeling.

An implicit function is continuous in its domain if its limit of input x approaching some value c is equal to the function evaluated at c; this implies that there are no holes or gaps in the function. Continuity in curves relates to levels of aesthetic smoothness of the curves, categorized into 3 main distinctions: C^0, C^1, and C^2. C^0 continuity adheres to the definition of continuous for implicit functions (no gaps) but allows the curve to have kinks and sharp edges. C^1 continuous curves have also their first derivative continuous, and likewise C^2 curves are continuous in their second derivative.

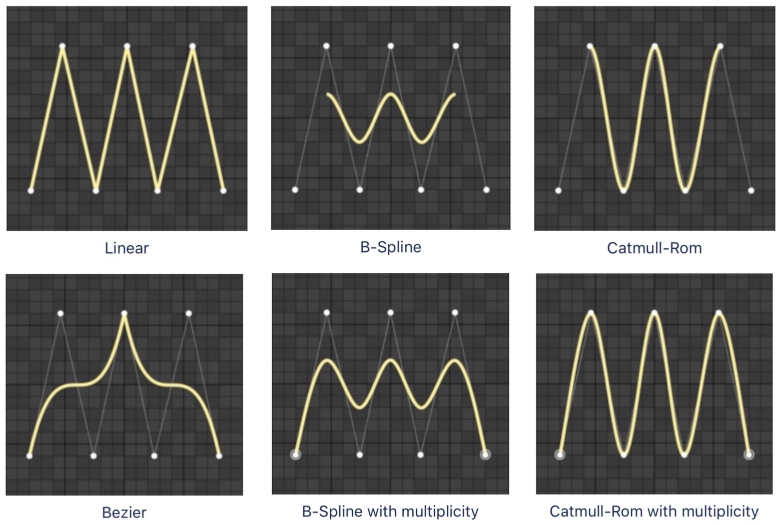


Figure 2.3: Curves sampled with 4 different basis functions using the same CVs in RenderMan [11].

2.2.4.1 Types and Methods

There exist multiple methods for representing continuous curves, several of which are Bézier, NURBS (B-splines), and Catmull-Rom splines. These are parametric, polynomial curves driven by *control vertices* (CVs) that define where and how a series of basis functions are combined to represent the curve precisely. Control vertices of a curve are points that guide a curve and provide more control over its shape rather than does interpolation between exact samples along the curve. The number of CVs is related to the degree of the curve, often being one more than the order.

Bézier curves represent polynomials of degree n, which are based on the Bernstein polynomial, and have $n + 1$ control vertices. They are affine invariant, meaning they can translate, rotate, scale, and skew while preserving the respective points of the curve. Bézier curves can also be efficiently subdivided into more Bézier curves. One downfall of these curves is that altering one control vertex, in addition to influencing the nearby points of the curve, affects the opposite end of the curve (or surface if a Bézier patch) [46] [13].

B-splines are a general representation of curves as a linear combination of simpler, basis functions of weights and control vertices; the "B" stands for "basis." They have a higher degree of continuity than Bézier curves. Non-uniform rational B-splines (NURBS) are a type of B-splines that use scalar weights per CV and a knot vector to define the influence of each CV on the curve. NURBS curves and surfaces are used in a wide variety of applications where very smooth curves and curved surfaces with more control over their shape than Bézier are desired. NURBS surfaces are curved patches with CVs in two dimensions, U and V; these two dimensions can be viewed as rows and columns of the surface that individually represent curves themselves [13].

The Catmull-Rom basis for splines is also often used for curve representation. It differs from Bézier and NURBS in that the spline passes through each interior control vertex but not the first and last (exterior) control vertices. They have C^1 continuity, though, and may have kinks in their shape where a B-spline would produce a smoother curve but not pass through the CVs [23] [11] [46].

Figure 2.3 is composed of figures from Pixar RenderMan's documentation [11] to illustrate intuitive differences in the curve bases when using the same control vertices. The linear basis is the most simple and connects the CVs via lines. Bézier curves pass through every third CV but their smoothness may be discontinous at these points. B-splines are smoother, and Catmull-Rom splines go through the CVs; for either, they need multiplicity - duplication of knots - to pass through the endpoint control vertices.

2.2.4.2 Representing Feather Parts

This thesis presents considerations for representing parts of a feather with curves, further discussed in the implementation section. The geometry generation is handled in Autodesk's professional modeling and animation software, Maya [6], which supports modeling with NURBS curves and surfaces. Ultimately, there is a desire for continuous, curved shapes within feathers as they appear in the real world. The shapes considered are the shaft, barbs composed of barbules, and vanes. Cubic curves with 4 control vertices for shaft and barbs may be sufficient in representing the shape of these components, including both the pennaceous barbs of the vanes and plumaceous ones of the afterfeather, but higher-degree curves may be considered to capture more characteristics such as a sharper curling of the pennaceous barbs at the edge of vanes. For efficiency, simple lines (degree-1 curves with 2 CVs) may be sufficient for representing barbules due to their small size; higher orders may be used if viewing the final feather at small scales. NURBS surfaces with 4 CVs in either axis may also be considered for modeling the vanes of a feather, from which curves can be derived in generating barbs and barbules. Similarly, degree-3 NURBS curves can represent the outlining shape of the vanes.

2.2.4.3 Rendering

In considering how to render the feathers, curves are the primary model for rendering investigated due to the curve-based nature of the structure of feathers. Pixar's path-tracing renderer RenderMan is used in this thesis. Parameters affecting the appearance during the rendering stage include the axial diameters of each curve at the beginning and the end of the curve, a material to describe how the curve surface interacts with light, texture or colors to assign to the curve, and the spline basis for evaluating the curves. As Maya NURBS curves are created for the parts of a generated feather, the basis for rendering is B-splines [11].

2.2.5 Image Analysis

Image analysis is observing and extracting meaningful data contained within images and pertains to this thesis as a means of incorporating real-world data of feathers in their procedural modeling. This section particularly focuses on separating an image into different portions, analyzing similarities in shape and texture, or color, among a set of images, and fitting curves to features present in an image. The programming language Python [9] was used for the implementation of this thesis with the scientific-computing libraries SciPy [12] and NumPy [7] for the image-analysis portion.

2.2.5.1 Image Segmentation

Image segmentation is the process of separating an image into sets of pixels based on per pixel properties such as color and intensity and is useful in determining where multiple, individual feathers lie in an image in this case. Techniques involved include observing gradients in pixel intensities and edge detection.

Watershed segmentation is one method of image segmentation that divides an image by similar features via treating the grayscale version of that image as topographical surfaces based on minimum and maximum intensities. This process detects a flow of pixels via "flooding" the map of the image starting at the minima and spreading in the directions of the pixel gradients. Keeping the water sources separate, the image becomes separated into two types of regions: topologically low "basins" around the minima and high "watersheds" remaining. The watershed method creates an image transformation that can find gradients and ultimately contours of objects in the image [17] [16].

2.2.5.2 Active Shape and Appearance Models

Active Shape Models (ASMs) [27] and Active Appearance Models (AAMs) [28] are techniques for extracting and representing shape and appearance information respectively of a particular class of objects. They are built using an alignment process and statistical analysis and encoding usually via PCA. ASMs focus on the placements of landmarks in an image and their relationship with one another based on edge connections provided through a 2D mesh or derived from an algorithm such as Delaunay triangulation [27], and AAMs include the same shape component but also similarly sample and encode the texture for each triangle in a shape-free context. The model resulting from the training phase for either ASM or AAM methods may be used for fitting a particular object in previously unseen images resulting in landmarks (capturing the shape) or creating a new shape by altering the parameters. An AAM additionally reconstructs shaded pixels (the texture) [28]. AAMs have been extensively used for analyzing images of human faces but can be applied to any object of interest. The implementation of AAMs through Imperial College London's Menpo project, implemented as a Python library, for such deformable models is used in this thesis [19].

2.2.5.3 Curve Regression

Curve regression is the process of best fitting a curve function to a given set of points, also called finding the curve of best fit. These curves can be polynomial or splines connecting piecewise functions. Linear regression is a specific type where a line of best fit (polynomial of degree 1) is found to match the dataset.

Polynomial curves are mathematical functions of a defined degree n and $n + 1$ coefficients with one or more inputs; in 2D space, one input x is used. The goal of curve regression is to find such a function given a set of points to guide the curve and a degree. The coefficients are typically contained within a matrix when solving this problem, and they are updated and ultimately found by iterations of reducing a squared error measurement between the points on a intermediate function using the current coefficients and the desired, given points [18]. This technique is used here for matching curves in feather images.

Depending on the application, a simple polynomial regression may not best represent the data. An alternative approach to curve regression is using regression splines. This process begins with fitting piecewise stepping functions to individual portions of the data along the input dimension, and each section may have a different basis function than the exponential input in polynomial regression. To assure continuity, splines are used to connect the piecewise polynomials, and the new function is C^0 continuous and likely has sharp bends. The second derivative of the new function achieves desired C^2 continuity; this derivative is a spline function [63].

Concerning the shape of the polynomial curve, higher degrees can align more closely to the set of points to regress to. For example, degree-2 (quadratic) curves can fit data with a smooth, parabolic shape containing a local minimum or maximum value. Degree-3 (cubic) curves have one local minimum and one local maximum, producing an S-like shape. Degree-4 curves have one minimum and two maxima, or one maximum and two minima. Varying degrees of curves are later considered in analyzing portions of a feather image.

Chapter 3

Related Work

This chapter is a survey of the previous work done on generating computer-graphics feathers in both research and production realms as well as ornithological studies on feather structure.

3.1 Computer-Graphics Research

The research in computer graphics presented here covers academic student theses and published work from companies and universities. A large portion of the research on generating individual feathers comes from the early 2000s, applying curves (commonly Bézier) and inspired by L-systems. Some of the work in this section addresses creating many feathers as coats on geometric surfaces but without improving the model of a single feather.

3.1.1 Academic Theses

Strett published a Ph.D. thesis on procedurally generating feathers, particularly focused on how feathers of a bird change throughout its lifetime. A chapter is dedicated on feather structure, describing a feather in terms of a shaft curve and barb guide curves but also can treat the entire feather as one tessellated surface. The individual feathers are modeled and parameterized for easy interpolation between attribute values as one ultimate goal is placing multiple feathers on a surface. Growth is also modeled after a designed interpolation schedule created based on the summarized process, but no real-world data measurements are presented to back such decisions. A portion of the thesis is concerned with creating multiple feathers, and the feather instances look similar to one another [65].

Newport created a Master's thesis on a feather-coat tool that heavily cites [25] and [64] for background material and modeling the individual feather structure. With the tool, a user creates multiple, groomable curves on a surface then can set "control barb" curves for some of the shaft curves and interpolation for the rest of the feathers [50]. No new work is presented on the structure of the individual, generated feathers but rather simplifies the model from [64] by eliminating the calamus.

Two Ph.D. theses observe the geometric components of feathers by means of understanding their structural coloration. In their separate work, Harvey and Eliason measure the reflectance of visible parts of feathers, the barbs and barbules, but also investigate their micro- and nanoscopic structure [39] [30], a topic towards which research on feather geometry can extend.

Although these academic works have novel contributions to feathers in computer graphics, this thesis addresses areas of the work previously unobserved such as modeling feather geometry at the barbule level and generating the lower-detail shape of the feather informed by images of feather samples.

3.1.2 L-Systems and Bézier Curves

Parametric L-systems are useful in defining a variety of types of feathers. The implementation in the work of Chen et al. [25] incorporates Bézier curves and defines the rachis and the barbs of feathers in L-system modules. A user can define curves that are used as outlines for the vanes and shapes of the barbs. The feathers seem to exist only as 2D planes, having curvature in two dimensions. The authors also add random "forces" to determine splits among the barbs, and feather barbules are present only through a texture.

Franco and Walter have implemented a similar approach, also using a collection of Bézier curves but not organized into L-system modules. Their work has the user specify key barb curves. Variation on these guide barbs allows the user to design the different kinds of feathers [34]. Streit and Heidrich's work is closely related to Franco and Walter's in the collection of curves, which where implemented in Newport's thesis. They additionally simulated feather growth and implemented the ratio between barb and rachis lengths and the afterfeather from feather growth in their approach [64].

No barbule geometry was created in any of these approaches, and model and parameter decisions are not supported by measurements of physical feathers.

3.2 Production

Film and visual-effects studios often develop their own in-house software tools for procedural content generation such as hair and fur, forests, and clothing. Particularly concerning feathers, studios have published some work within the past 15 years primarily including proprietary software tools and often grouped with the grooming and simulation stages of production pipelines. Although they provide procedural solutions, the tools are artist-driven, and a lack of real-world measurements persists here.

3.2.1 Feather Tools

Software systems for procedurally generating feathers have been developed at several film and visual-effects studios and companies and mainly presented as high-level conference talks or article interviews rather than in-depth publications.

Animal Logic's feather system, Quill, is briefly documented for a SIGGRAPH talk in [40] as a tool specifically for the film *Legend of the Guardians: The Owls of Ga'Hoole.* Concerning the structure representing a feather, it is noted that "extensive level-of-detail and procedural data management is required to efficiently render close shots of feathers with millions of individual barbs as well as crowd shots with hundreds of feathered characters" and each feather can be "rendered from a simple quadrilateral to thousands of curves" [40]. A Computer Graphics World article provides more detail on the work in describing artist control for modeling and grooming, rigging and simulation, and rendering. During modeling, artists placed about 1000 guiding feathers, particularly the highly visible remiges, from which thousands more were interpolated; and down feathers were created procedurally. In the article, individual feather parameters are described as where the feather "becomes smooth and uniform" (the rachis and vanes) after a "scraggly base" (the calamus and afterfeather) along with length and width. Many barb curves were generated from each feather with no barbules, and each curve was rendered through Autodesk's Maya and Pixar's RenderMan software [56].

Additional studio feather work includes ILM's solution to solving penetration-free feathers in *Rango* [20], Moving Picture Company's (MPC's) feathers on Pegasus creatures in *Clash of the Titans* [44], and DreamWorks's animation of the large feather in Puss in Boot's hat in *Shrek 2* [51]. These resources are mostly talk abstracts with high-level discussion on the internal tools.

Tighe Rzankowski developed a feather plug-in for the visual-effects software Houdini made by SideFX during an internship with the company with editability and animation in mind and is built using the Houdini libraries and node system. The tool allows for control over the feather shape in terms of a shaft curve and barbs and has level-of-detail output options as polygonal surfaces or a series of curves. No barbules are included in the shape model, and texture may be applied to the curves or surfaces via texture coordinates and shaders. Concerning multiple feathers, it supports generating feathers from hair nodes and animating the feathers based on a path to use as a bird wing for example [59].

3.2.2 Hair and Fur

A related type of software that studios develop are hair and fur tools. These and feather tools are mostly focused on a grooming portion of the modeling stage in a production pipeline, and several feather tools have been built upon hair systems.

Disney Animation Studios, Animal Logic, MPC, and Weta Digital have developed their own proprietary fur and hair creation systems. Recent work in particular observes differences between human hair and animal fur [14] [2] [35]. Dexter Studios also developed their own fur and hair grooming tool and expanded it to a feather tool where feather shafts are made from fur strands; feather instances can be represented as polygonal surfaces or as sets of curves (the barbs) [26]. Rzankowski incorporated a similar approach in Houdini with letting a hair tool drive a feather groom.

Feathers are more complicated structures than individual strands of hair. Rather than single curves, they are a series of curves connected in specific ways, which is briefly discussed in the previous section and further elaborated upon in the implementation portion of this thesis.

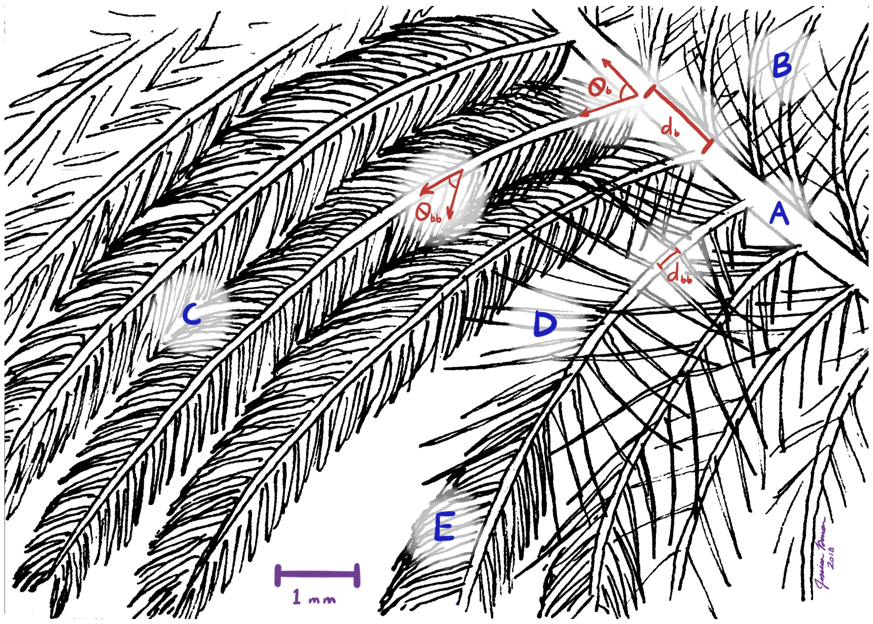


Figure 3.1: Illustration of barbs and barbules on a contour feather with labels.

3.3 Data from Ornithological Studies

In computer-graphics research and visual-effects production software, there has been a lack of incorporation of physical and biological data for real-world feathers. As mentioned in the background, several biological studies do exist that have gathered some statistics on variation in specific sets of feathers but spanning 21 different orders of birds. Differences discussed such as variation in barb and barbule densities, spacing, and angles are visualized in figure 3.1 with labels of parts and attributes: A. shaft, B. plumaceous barbs, C. pennaceous barbs, D. plumaceous barbules, E. pennaceous barbules, distance/spacing between barbs d_b, distance between barbules d_{bb}, barb angle θ_b, and barbule angle θ_{bb}.

3.3.1 Barb Angles, Barb Lengths, and Vane Widths

In investigating evolutionary connections of flight through asymmetry in feathers, Feo and Prum observed the vanes of 3 particular primary feathers from 60 extant bird species and measured barb angle, barb length, and vane width. With barb angle defined as the angle between the rachis and the barb, their diagrams show a range of angles of the vanes from 5 to 50 degrees but the average in 20 to 40 degrees. They show that the angle increases with the vane width; the thin notches in primaries have the shortest barb angles, and the widest portions of the feathers have barbs rotated out more. Their measurements show barb lengths from 10 to 60 millimeters, and vane width is dependent upon the growth morphology of feathers, specifically concerning the the barb length L and barb angle θ. They estimate vane width W as $W = Lsin\theta$ [32] which is incorporated into the work of this thesis.

They also had a similar, previous study on the vane asymmetry in leading (distal) and trailing (proximal) vanes of coverts and rectrices of two species of parrots (family *Psittacidae*). The vanes measured 5 to 15 millimeters wide, slightly increasing when further away from the feather tip. Barb length exhibits the same pattern as being greater the further away from the tip and cause the width to increase; they measured 5 to 25 millimeters long. The barb angles they found ranged between 10 and 40 degrees. They observed another characteristic, barb curvature, as the distance of the actual barb tip from where it would be placed in respect to the rachis if it protruded in a straight line along the barb angle; this offset was often measured to be between under 1 but up to 4 millimeters [33]. Prum additionally has earlier work analyzing barbs' growth from the helical shape of the shaft and mathematically modeled the length and curve of barbs extending from the rachis but does not discuss measurements from feather samples [53].

3.3.2 Barb Diameters and Spacing

Rijke investigated the diameters of and spacing between barbs of aquatic birds to understand why cormorants (family *Phalacrocoracidae*) are less water resistant than ducks (family *Anatidae*). Measuring the barbs of species of ducks and cormorants, he found an average barb diameter from samples of both familys as 56 micrometers and an average spacing between barb centers on the same plane as 271 micrometers.

3.3.3 Barb and Barbule Densities

Broggi et al. collected and observed differences in contour feathers from the chests of 3 different types of captive and wild great tits (family *Paridae*) in Scandinavia. In exploring differences in insulation among the birds, they measured densities of barbs and barbules, finding an average of 1.47 pennaceous barbs and 2.97 plumaceous barbs per millimeter of the rachis; pennaceous barbs are spaced along the rachis at almost twice the distance of the plumaceous ones on the tit feathers. The barbule densities were measured at 2.17 pennaceous barbules per one tenth of a millimeter on a barb and 2.66 plumaceous barbules per one tenth millimeter; similarly as the barbs, pennaceous barbules are slightly more spread out than the plumaceous ones. The barbule spacing is about a one-tenth scale of that of the barbs. The average length of the feathers collected was 21.5 millimeters, and the ratio of plumaceous barbs to all barbs was 72.67% on average [21].

3.3.4 Barb and Barbule Angles and Lengths

In his book on hummingbirds, Greenewalt investigates the iridescence in the appearance of hummingbirds (family *Trochilidae*) through growth and reflection angles of barbules, two defined as the "barbular" and "vaneular" angles. The barbular one measures the rotation of the barbule away from the barb estimated as ranging from 0 to 70 degrees. The vaneular angle is the dihedral angle of barbules branching from the same location on a barb and is measured from 0 to 45 degrees from their shared plane. It is also noted that the length of a barbule is about 100 micrometers and the diameter as around 15 micrometers.

3.4 Summary

Overall, there are key trends in what has been done before concerning generating and studying (1) the shaft, (2) the barbs, and (3) the barbules. Artist control has been emphasized for computer-graphics purposes without the incorporation of real-world data as the material presented in the previous section. Firstly, digital feather generation often begins with the shaft (as do real feathers during growth). Often in production tools, the shaft is a user-controlled curve sometimes starting from a hair or fur groom. Some data observing the lengths of real feathers was found but not concerning the curvature of the shaft.

The following components of a feather are the barbs and barbules which compose vanes. The vanes of a procedurally generated feather may be represented as a single polygonal surface as the highest level-of-detail or by a low level-of-detail as a large set of barbs. When barbs are desired, artist-controlled curves have been used to guide individual barb curves and overall outer curvature to a feather's shape. In computer-graphics work, there has been no effort in producing the geometry of barbules but rather synthesizing with texture only. Concerning real-world data, several ornithological publications study various angles of the barbs, their length, the widths of the vanes they form, the proportion of plumaceous barbs to all barbs, and densities of barbules in terms of density along barbs.

Chapter 4

Implementation

This chapter presents new work on the procedural generation of computer-graphics feathers, divided into three main categories: designing and building a feather tool for production, highlighting work completed as a software-engineering intern at a visual-effects studio; the data collection and analysis of real feathers; and the data-driven generation of feather geometry.

For consistency, a right-handed coordinate system is chosen to describe parts of a feather relative to a local orientation, each of the three axes perpendicular to one another. Starting with the system origin at the base of the calamus, the Y axis points its positive direction along the shaft, calamus to rachis. The Z axis orients the dorsal side of the feather in the positive Z direction, and the ventral side faces in the negative Z direction. Vanes lie primarily along the X axis, and the positive X direction is the positive Z one rotated clockwise about the Y-axis by 90 degrees; if viewing a feather by its distal side, the vane to the right of the shaft would be extending along positive X and the left-side vane in negative X. See figure 4.4 displaying these axes on a generated feather from the current study, not Apteryx, though both programs use this coordinate system.

4.1 Designing and Building Apteryx

Weta Digital is a world renown visual-effects company located in Wellington, New Zealand known particularly for their in-house software tools and work on films such as *The Lord of the Rings* franchise, *Avatar*, and the reboot of *Planet of the Apes*; their effects lean towards realism rather

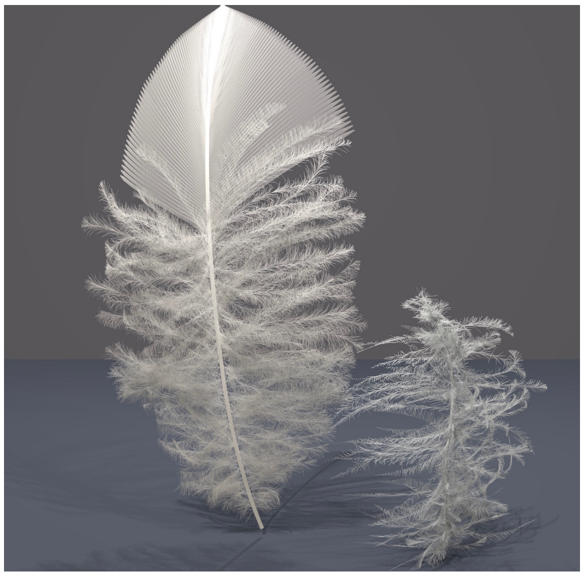


Figure 4.1: Down feathers generated with the Apteryx feather software library at Weta Digital.

than stylized. Through a software-engineering internship, I helped begin a new software library for Weta Digital for the procedural generation of feathers. This section summarizes the experience and highlights the components of creating such software for production.

4.1.1 Weta's In-House Tools

Weta Digital develops many proprietary software tools that solve particular tasks. Their largest projects focus on rendering, simulation, and procedural content generation. The feather-generation tool falls under the latter category. Other procedural-content software includes their hair and fur tool, Wig, and the tree tool, Lumberjack [2] [1]. These tools have similar tasks as the software for feathers, existing in the same production pipeline. Previous efforts on feather generation were employed at Weta, but the development of a new system was desired based on needs for more realism, better control, and improved simulation.

4.1.2 The New Tool: Apteryx

During an internship at Weta, I helped drive the development of the new software library which was focused on the biologically inspired geometry of a single feather as a collection of parametric surfaces and curves under artist control. The tool is named *Apteryx*, meaning “without wing” and is the genus name for the flightless kiwi bird species of New Zealand [57]. Contributions to this tool through the internship were focused on creating a parameterized and biologically accurate model of an individual feather. Design discussion included planning for future needs within Weta’s production pipeline such as physically accurate rendering and shading, many-feather layouts, collision detection, and simulation. The software is a central, C++ library with plug-in components for interacting with third-party professional software such as Maya and Katana [6] [5]. Figure 4.1 is a real-time render of down feathers procedurally created with this tool where handling barbules is needed for accurately representing down feathers.

4.1.2.1 Feathers in the Production Pipeline

Feathers, along with other procedural assets such as Wig’s hair and Lumberjack’s trees, undergo various processes within the production pipeline at Weta Digital, starting with modeling to form the shape of the asset and ending with a final render and composite in a complete film shot. The following is a description of the lifetime of an Apteryx feather.

At the forefront, a modeling artist would initially generate a feather as a collection of NURBS curves and surfaces using the Maya plug-in of Apteryx. The artist would provide a NURBS curve as input to the system; this defines the curve of the generated feather’s shaft, off of which two NURBS surfaces form the main shapes of the vanes. Based on user parameters such as barb and barbule densities and afterfeather attributes, barbs and barbules are sampled as curves from these vane surfaces. During use in Maya, the Apteryx feathers are rendered in real time using a hair shader for initial visualization.

After modeling, the collection of curves would be sent to the look development department. Extra information describing these curves, called primitive variables or prim vars, would be sent along with the geometry; this data includes texture coordinates based on the vane surfaces so that texture may be assigned to the shaft, barbs, and barbules. After the surface of a feather is described by textures and materials, it may be rendered offline with Weta’s proprietary path-traced renderer Manuka [31].

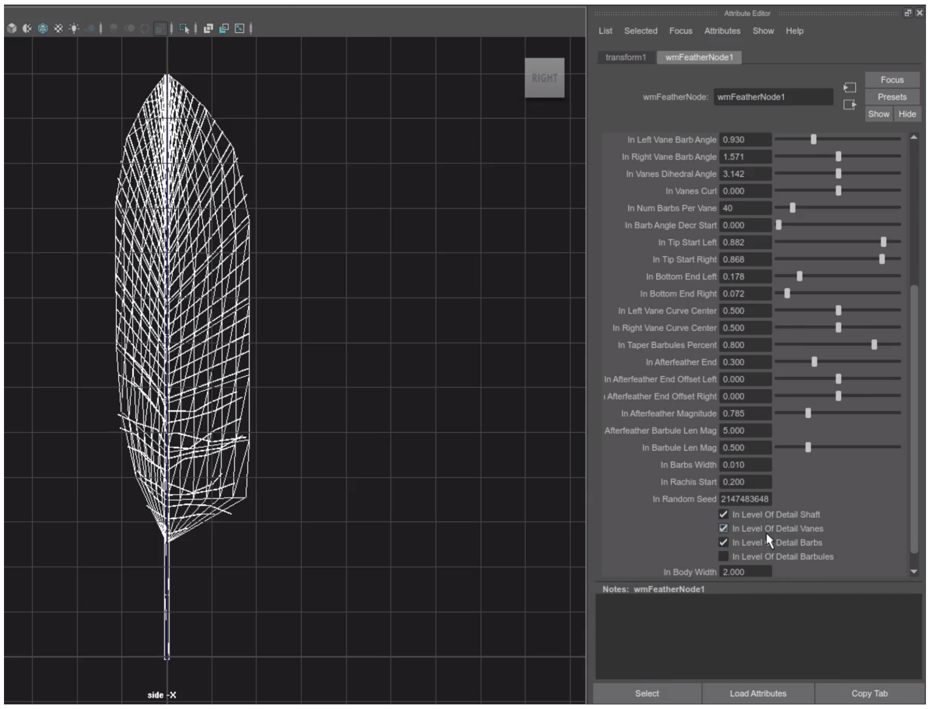
During the internship, the primary interaction of Apteryx with the production pipeline was in the modeling stage, editing parameters as they were implemented and improved, and further preparing for the role of feathers in the production pipeline by understanding of desires for the tool from the grooming artists in the modeling department. Also while working at the studio, an asset containing feathers was used to test the Apteryx tool first in the modeling stage, followed by look development, and completed after offline rendering; the result was promising for further use of Apteryx.

There exist additional parts of the production pipeline, particularly the creatures and simulation departments that implement behavior of assets for realistic animations. Throughout the research and implementation of Apteryx, the development team discussed using the current feather model for creating grooms of feathers and simulating their movement and interaction. Due to time constraints, though, these components were not the focus of the internship work.

4.1.2.2 Parameter Decisions

Feathers from the real world have great variety based on factors such as their type, the bird species, and the age of the bird. Based on discussions from meetings, information gathered from research and outreach, and studio pipeline requirements, parameters were decided upon and implemented concerning control over the feather shaft, the barbs and vanes, and the barbules. Weta strives for photorealism in their work. Attribute values were restricted with particular ranges chosen based on the look produced; the resulting appearance matched observed patterns of variation in real feathers, but these decisions are not backed up by statistics from feather analysis. Figure 4.2 is screenshot of using the Apteryx plug-in for Maya with attributes exposed for editing to the right and the viewport to the left shows a wireframe display of the barbs sampled over the vane surfaces.

The shaft of an Apteryx feather is primarily controlled by a NURBS curve created in Maya (**shaft curve**). An artist has full control over its shape (meaning the number of CVs and where they are in 3D space), and from this curve the rest of the feather components are driven. In transitioning between Maya input and the C++ library, a number of samples is taken along that curve for internal representation (**number of shaft samples** being a parameter option though typically set to 10). Another parameter for the shaft is the width of rendered curve (**shaft width**) in Maya units, which is considered based on the overall size of the feather. The user defines percentages along the shaft (increasing along Y-axis) where the rachis begins (**rachis start** with default as 0.1, one tenth) and where the afterfeather ends (**afterfeather end**, greater than rachis start and small on flight



wmFeatherNode: wmFeatherNode1

In Left Vane Barb Angle	0.930
In Right Vane Barb Angle	1.571
In Vanes Dihedral Angle	3.142
In Vanes Curl	0.000
In Num Barbs Per Vane	40
In Barb Angle Decr Start	0.000
In Tip Start Left	0.882
In Tip Start Right	0.868
In Bottom End Left	0.178
In Bottom End Right	0.072
In Left Vane Curve Center	0.500
In Right Vane Curve Center	0.500
In Taper Barbules Percent	0.800
In Afterfeather End	0.300
In Afterfeather End Offset Left	0.000
Afterfeather End Offset Right	0.000
In Afterfeather Magnitude	0.785
Afterfeather Barbule Len Mag	5.000
In Barbule Len Mag	0.500
In Barbs Width	0.010
In Rachis Start	0.200
In Random Seed	2147483648
In Body Width	2.000

✓ In Level Of Detail Shaft
✓ In Level Of Detail Vanes
✓ In Level Of Detail Barbs
☐ In Level Of Detail Barbules

Notes: wmFeatherNode1

Figure 4.2: Using the Apteryx plug-in in Maya with both vane surfaces and barbs visible.

feathers, such as 0.15).

There are parameters to control how the vane NURBS surfaces are shaped and how barbs are sampled from them. In changing the surface shapes, a dihedral angle may be provided to drive how both vanes fold in towards or away from each other about the shaft (**vanes dihedral angle**) with a default value of π or 180 degrees so the vanes lie on the same plane. An additional parameter may change the initial shape of the vanes by bending the surfaces back towards the shaft as they grow out (**vanes curl**) though by default this is set to a value of 0 for no effect. There is also some user control over outer curve shapes through editing 2 cubic functions that round off either end of the vane, such as where the tapering at the bottom of the vanes (**bottom end**) and where tapering at the distal end of the vanes begins (**tip start**).

Several per-vane attributes affect barb generation from the vane surfaces, including the **number of barbs** per vane (the length of the vane, and therefore density of the barbs should be considered), an offset to the main **afterfeather end** attribute so that the afterfeather can be positioned slightly differently for either vane (often 0 for no difference on symmetric contour feathers or a small percentage of the shaft such as 0.01), and the desired **average width** of the vane. One key parameter is the angle from which the tangent of the shaft curve at a certain point on the shaft is rotated towards the normal perpendicular to the feather (**barb angle**), often between 25 and 45 degrees. Each barb's length is calculated using this angle and the width of the vane. Because there is variation in this angle along the rachis where it decreases as approaching the distal tip of the shaft, a percent along the shaft may define when the barb angle should begin decreasing with linear interpolation between the given barb angle and 0 degrees by the end of the shaft (**barb angle decrease start**). A curve width for rendering may also be set as a fraction of the shaft's curve width (**barb curve width**). A random seed value as 64-bit unsigned integer to be used in sampling random values from a Gaussian distribution (**random seed**) can be changed and is applied to displacing the control vertices of the barb curves sampled from the vane surfaces to represent plumaceous barbs of the afterfeather. For each barb control vertex, a random uniform value scaled by an **afterfeather magnitude** rotation (with a default value of $\pi/4$) drives rotation about the tangent and normal axes of the surfaces to help create a soft look to the afterfeather.

Barbule properties may be set per pennaceous and plumaceous barbs as the barbules on a real feather may differ in these different sections. Similar to the **number-of-barbs** attribute, the user can control a barbule density as the **number of barbules** per barb; this value also drives the barbule length as the barb length divided by the number barbules per barb. A **barbule angle** is used in creating each barbule as the barb tangent rotation towards the barb normal, set to a default of 45 degrees in either positive or negative direction of the normal; figure 3.1 again is an illustration showing barb and barbule angles. Barbule tapering is also incorporated to affect the length of the barbules along a barb; the user can define at what progress along the barb the barbules will start shortening (**taper barbules start**), typically 0.8 as eighty percent of the barb length) and the minimum percentage of the barbule length to shrink to by the end of the barb end (**taper barbules to**) with linearly interpolated barbule lengths in between. There are extra options primarily for the plumaceous barbs of the afterfeather such as the magnitude of rotation variation, length increase (compared to pennaceous version), and random seed (64-bit unsigned integer) to drive the displacement away from vane surfaces of the plumaceous curves.

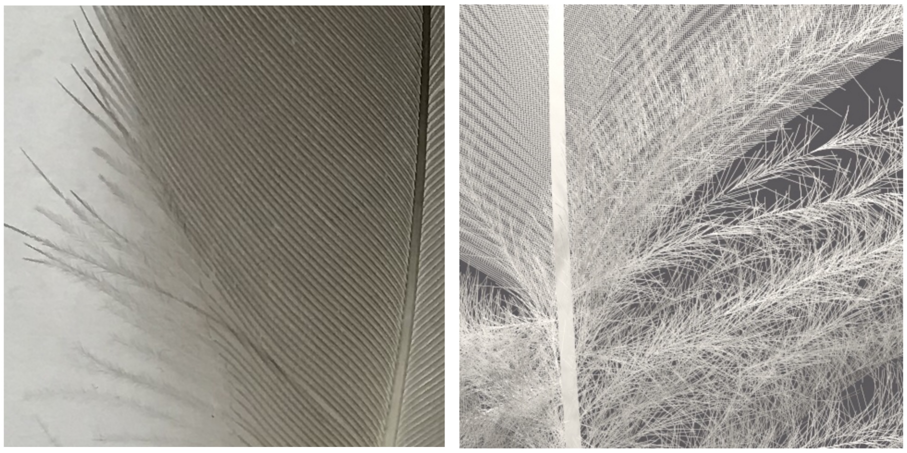


Figure 4.3: A comparison of pennaceous and plumaceous barbules in a real feather (left) and a feather from Apteryx (right).

4.1.3 Summary

In observation of forming the geometry of a feather under production requirements, certain characteristics may be summarized on creating the shaft, barb, and barbule components to produce photorealism and support artist control. In creating feathers with Apteryx, a desired feature is to have presets based on real feathers divided into categories such as feather type and bird species. In order to do so, though, information from real feathers is desired to drive the parameter value decisions.

4.1.3.1 Shaft

Through Apteryx, the artist has full control over the shaft curve in terms of being able to assign any NURBS curve from Maya to the feather object. Based on the algorithms controlling the generation of the rest of the feather components, though, the shape of the shaft curve should not vary much but rather be somewhat rigid for most feathers and curled about the local feather X-axis for feathers such as down. For rendering, a single thickness is assigned although existing feather shafts have various widths among their length.

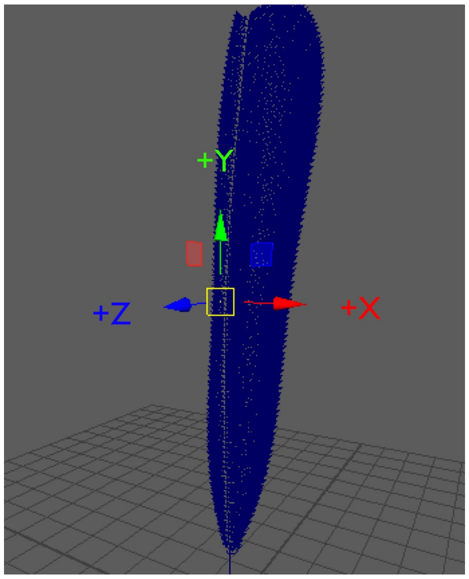


Figure 4.4: Local 3D feather axes on a feather generated by the new system.

4.1.3.2 Barbs

Many geometric parameters control the shape and appearance of the vanes as surfaces and barb curves sampled from the surfaces. They are inspired by the real world but default and restricted values are not verified by statistics. Further, the vane outlines in real feathers have great variation that may be better observed using higher-degree curves to model their shape than being driven by a vane width and cubic-curve tapering at either end.

4.1.3.3 Barbules

Decisions on barbule parameters mainly reflect differences in pennaceous and plumaceous barbules and how they appear at the end of the barb. Attributes such as the barbule length is affected by the spacing between barbs driven by the number of barbs, and the number of barbules relies on this. The width of one barbule's curve for rendering is a fraction of the barb width. Figure 4.3 compares the pennaceous and plumaceous barbules in both a contour feather from a Southern Black-Backed Gull and a feather generated by Apteryx. Rendering a feather at the highest level-of-detail entailed including the barbules, and this performed in Weta's real-time renderer during editing within Maya with no issue.

4.2 Data Collection and Analysis

In preparation for generating computer-graphics feathers more informed by real feathers, methods for collecting and extracting data were developed from investigating the data readily available, representing the data in a specific format for later use, and acquiring additional data from images.

4.2.1 The Feather Atlas

The core source of image data used was The Feather Atlas, a database of feather scans available online [3]. Each scan consists of one image of multiple feathers on a grid. Figure 4.5 illustrates how their scans are set up; the rectrices are from the right half of the tail and are numbered according to an ornithological standard for flight feathers. This standard defines an order to the remiges and rectrices. Primaries are numbered from 1 to 10 or 12, from most proximal or inner to most distal or outer. The labels are the character "P" followed by the number: "P1", "P2", and so on. Secondaries are ordered in the opposite direction – distal to proximal – and labeled with "S" so that feathers "S1" and "P1" lie beside each other. Rectrices are labeled with the prefix "R" and are numbered per half of a tail from 1 to 5 or 6, located centermost in the tail to outermost. Additional metadata provided with each scan includes the common name of the bird, the Latin name, the bird's order and family, the type of the feathers, and, if known, the age and sex of the bird along with location of discovery. For this work key selections were taken manually, but this could be extended to the whole set. The maintainers were contacted, and web scraping was considered, but constraints prevented both options.

4.2.2 Data Representation

Each scan sampled from The Feather Atlas is organized mainly by bird and by feather. Most per-bird and per-feather information is stored and updated in dictionaries written to files. A dictionary in a software program is a data type that is a collection of key-value pairs; keys

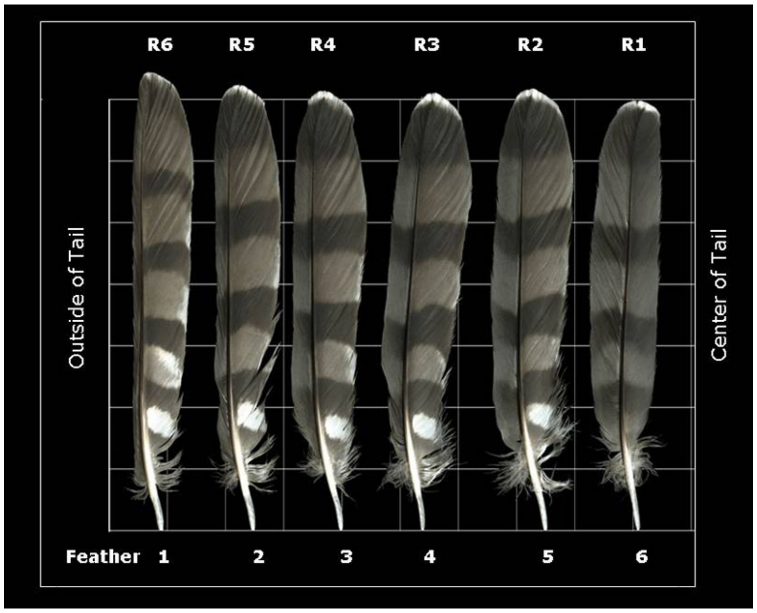


Figure 4.5: Image from The Feather Atlas displaying how rectrices are laid out for a scan.

provide names for retrieving values of particular attributes. For example, one key-value pair of a bird dictionary is “Common Name”: “Wild Turkey” and for a feather dictionary “Length”: 27.5. Python’s built-in dictionary type was used along with the JavaScript Object Notation (JSON) module for storing object-value information to disk as the JSON text file [4]. The image and metadata per bird and per Feather Atlas scan provide a start of feather-specific data, particularly the image name and subset of the metadata containing bird common name, Latin name, order, family, and labels, total lengths, and vane lengths of the feathers scanned. Tables 4.1 and 4.2 display the information stored in the bird and feather dictionaries; the additional information to the feather dictionary will be discussed further in the next section. Also note that the values of some attributes match between the bird and feather dictionaries; for example, the “Bird” value of a feather should match the “Common Name” of a bird, and the feather “Type” should be within its bird’s “Feathers” list.

Altogether, the information pertaining to one bird with three feather scans, one per type of flight feather, involves the bird dictionary (one JSON file), multiple feather dictionaries (with one JSON file each), the original image from The Feather Atlas, that image segmented into separate images per feather, and landmarks for the feather images (each stored as a PTS file discussed later in this section).

Per-Bird Data	
Common Name	Blue Jay
Latin Name	*Cyanocitta cristata*
Order	*Passeriformes*
Family	*Corvidae*
Images	BLJA_primary_adult.jpg, BLJA_secondary_adult.jpg, BLJA_tail_adult.jpg
Feathers	P10, P9, P8, P7, P6, P5, P4, P3, P2, P1, S9, S8, S7, S6, S5, S4, S3, S2, S1, R6, R5, R4, R3, R2, R1

Table 4.1: An example of the data stored in a dictionary per bird.

Per-Feather Data	
Type	R6
Image	BLJA_tail_adult_0.png
Bird Name	Blue Jay
Length (cm)	12.2
Vane Length (cm)	11.8
Shaft Degree (Poly)	4
Shaft Coefficients (Poly)	0.4048, -0.7821, 0.5638, -0.1594, 0.0647
Shaft Degree (Spline)	3
Shaft Coefficients (Spline)	0.0662, 0.0500, 0.0560, 0.0523, 0.0467, 0.0537, 0.0517, 0.0558, 0.0564, 0.0629, 0.0671, 0.0721, 0.0764, 0.0939
Shaft Knots (Spline)	0.0, 0.0, 0.0, 0.0, 0.1122, 0.2105, 0.3070, 0.4177, 0.5221, 0.6037, 0.6831, 0.7633, 0.8297, 0.8881, 1.0, 1.0, 1.0, 1.0
Outer Left Degree (Poly)	5
Outer Left Coefficients (Poly)	-19.5329, 18.1961, 20.4375, -27.5968, 8.6484, 0.1830
Outer Left Degree (Spline)	3
Outer Left Coefficients (Spline)	0.1665, 1.2186, 0.9553, 0.6583, 0.7766, 0.7604, 0.8409, 0.7277, 1.1280, 0.4751, 1.2441, 0.0603
Outer Left Knots (Spline)	0.0, 0.0, 0.0, 0.0, 0.2758, 0.3998, 0.5510, 0.6710, 0.7561, 0.8304, 0.9026, 0.9568, 1.0, 1.0, 1.0, 1.0
Outer Right Degree (Poly)	5
Outer Right Coefficients (Poly)	-23.3850, 42.7729, -22.8364, 0.3336, 3.1941, 0.0619
Outer Right Degree (Spline)	3
Outer Right Coefficients (Spline)	0.0669, 1.0623, 0.4273, 0.8556, 0.9371, 1.0084, 1.0049, 0.9766, 0.8568, 0.7779, 0.4357, 0.0176
Outer Right Knots (Spline)	0.0, 0.0, 0.0, 0.0, 0.3048, 0.4228, 0.5262, 0.6310, 0.7261, 0.8112, 0.8956, 0.9588, 1.0, 1.0, 1.0, 1.0
Rachis Start	0.1406
Left Vane Width	0.0339
Right Vane Width	0.1161
Vane Length Calculated (cm)	10.4846
Vane Lengths % Difference	0.1078

Table 4.2: An example of the data stored in a dictionary per feather.

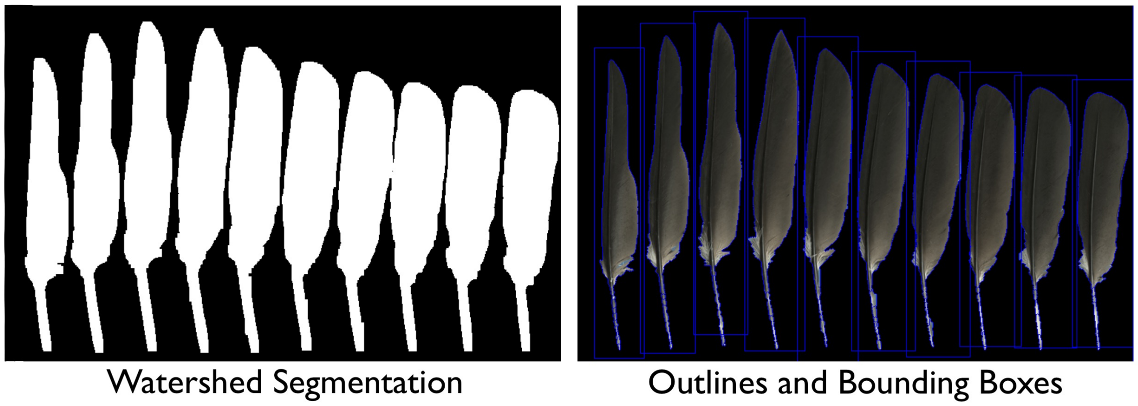


Figure 4.6: Scan of Double-Crested Cormorant primaries segmented via the Watershed algorithm.

4.2.3 Image Analysis and Data Extraction

The metadata readily available from The Feather Atlas is not enough to drive the creation of feather geometry; this motivates the desire to extract additional information from the scan images. In particular, automatically landmarking significant regions of a feather image and regressing curves and finding component proportions from those locations provides information to apply to the generation of a feather in 3D space.

The data extraction begins with an individual feather scan entry from The Feather Atlas. For each scan, the available metadata for the bird and feathers is stored and updated appropriately (i.e. adding a different feather-type scan to a bird already with information on at least one type of feather). Also during this stage, the multiple-feather image is segmented into multiple images by feather. The splitting is achieved using the Watershed image-segmentation algorithm for finding regions of high gradients due to the problem of finding features of the image with similar appearance – the feathers; the algorithm is implemented in OpenCV and used through the library's Python bindings after replacing the variable background color of the original image with white and converting this image to grayscale for the algorithm. Two stages of the Watershed process are shown in figure 4.6. An extra effort is taken to assure that the order in which feathers are numbered and saved to separate files correlates to the left-to-right order of placement on the grid and therefore the separate images are organized by their ornithological labeling. The first for-loop in algorithm 1 reflects this process.

Algorithm 1: Per-Scan Data Extraction

```
input  : Feather scan as multi-feather image and metadata table
output: New/updated bird and feather dictionaries as JSON files, split images, and
        landmark PTS files
1  for each scan do
2      JSON files = Create and update bird and feather dictionaries.
3      Split images = Watershed(scan image)
4  end
5  for each feather image do
6      PTS file = Landmark(image)
7      Shaft curve = Regress polynomial curve for shaft using PTS.
8      Vane-width functions = Regress and derive curve functions per vane using PTS.
9      Updated JSON files = Save curve functions and other derived data.
10 end
```

4.2.3.1 Landmarks

After the feathers are split into separate images, each one is processed. The first step per feather image is to place all 2D landmarks from a 52-point scheme designed to outline and sample the shape of the shaft and borders the outer edges of the vanes; a smaller amount of points was insufficient in the remaining steps of the image analysis. These landmarks may be placed semi-automatically using an Active Appearance Model trained on manually landmarked samples to learn the relationship among the shape and texture of the training images; after being built, an AAM is used for fitting unseen images with the landmarks it was built upon. The landmarks are stored in a PTS file which is a simple and standard text format for saving such landmarks, or points, to disk by their location as 2D pixel coordinates. After a feather is landmarked, it undergoes further analysis based on the points. Figure 4.7 displays the 52 points placed on a feather image.

4.2.3.2 2D Coordinate Spaces

Several normalized coordinate spaces in 2D are established to help define and apply certain functions and attributes per image: *normalized image space*, *feather space*, and *vane space*. The new image space is a scaled version of the original pixel-by-pixel dimensions of an image with the aspect ratio skewed to a uniform 1:1 value by dividing each X and Y coordinate by the width and height of

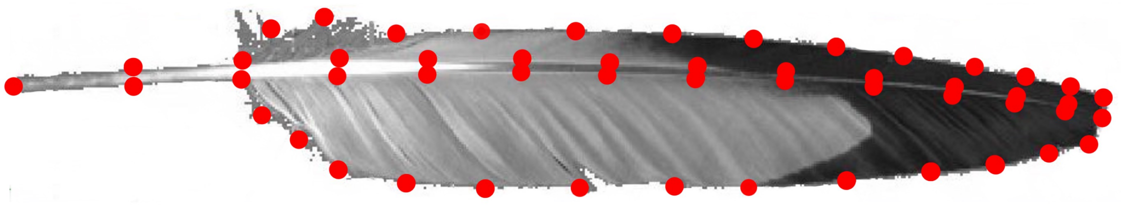


Figure 4.7: Landmarks for a feather image.

the image, additionally subtracting 1 by the scaled y so the origin is at the bottom left of the image rather than the top left. During image analysis and curve regression, each point's coordinate is first transformed from the original pixel coordinate (ranging from (0,0) to (image width, image height)) to the normalized image space, and the new y coordinates marked by landmarks for the start of the feather as the tip of the calamus (**feather start**), the end of the feather as the distal-most point (**feather end**), and the start of the rachis (**rachis start**) are stored.

Image- and feather-space x and y coordinates are defined by the y coordinate following along the length of the shaft and x coordinate perpendicular to the shaft along the widths of the vanes. In feather space, the tip of the calamus has a y coordinate of 0.0 and the distal end of the feather is at 1.0, and the shaft lies on the x coordinate value of 0.5, both achieved through translation and scaling transformations from image space using the start and end positions found in the previous step. The vane widths preserve their proportion to the feather's length but are shifted along the X axis during transformation of the shaft to x = 0.5, and a maximum width per vane is stored as a percentage of the feather's length.

The vane-width functions are defined in a "vane space" where the Y axis ranges in [0.0, 1.0] along the length of the rachis, and the X axis is the width of the vane perpendicular to a certain progress along the rachis given as a value acting as a percentage of the maximum width. Figure 4.8 displays the vane-space width curves where green represents the vane in the image to the left of the shaft and the red represents the other, right vane. The additional chart shows the same curves transformed back into the original image space and drawn on top of the image.

4.2.3.3 Curve Regression and Additional Attributes

Once a feather image has been landmarked, useful information can be derived from the landmarks' positions, namely several curves and proportions of parts of a feather in respect to each other. This thesis uses the NumPy and SciPy scientific-computing libraries in Python, particularly for polynomial- and spline-fitting functions in curve regression. These functions find the coefficients for a polynomial of a given degree or the coefficients and knots of a cubic B-spline that best match the provided points based on least-mean-squared errors and piecewise splines. Figure 4.9 displays both regressions and vane-width functions on a single feather with the resulting renders. Each approach was implemented for comparison with one another. Additionally, statistics for larger analysis, such as averages based on species and feather type, may be calculated based on the coefficients in the polynomial representation; comparison among the spline fittings would however require identical knot vectors.

In preparation for curve regression, three values are found during landmark processing and stored for later use: the start of the feather, the end of the feather, and the start of the rachis, each represented as a y-coordinate value in the normalized image space. The feather's position within the image are crucial in determining feather-space coordinates for points used in finding the shaft curve. The rachis start converted to feather space is stored to the feather dictionary and often has a value near 0.1, or 10% of the feather length.

The first curve found is regressed to represent the shape of the shaft of the feather. A series of landmarks represent the left and right sides of the shaft due to the AAM needing shape information (features such as edges within an image) to help define the relationship between the landmarks. The final shaft curve to be stored is the average of the curves fitted to either border of the shaft. In regressing both intermediate curves, the landmark coordinates are first converted to feather space before calling the SciPy and NumPy functions.

The width functions per vane are created next using the shaft curve, and the goal is to produce a parametric function for each vane that takes the progress along the rachis portion of the shaft (between 0.0 and 1.0) and outputs how wide the vane is perpendicular to the shaft at that progress. For each vane, each corresponding landmark is converted to vane-space coordinates and the shaft curve sampled to find a width value between the current landmark and point on the shaft. After all widths are found, the maximum width is calculated, converted to feather space, then

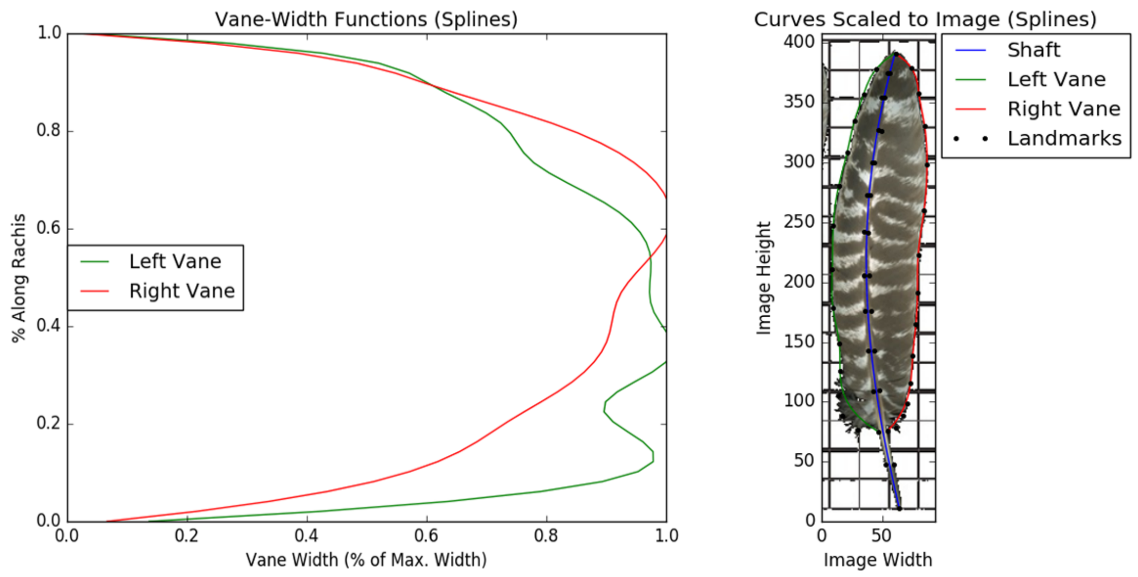


Figure 4.8: Plots of the vane-space width functions (left) and the curves scaled to match the input image (right) for the Wild Turkey's S7 feather based on B-spline regression.

used for scaling all of the widths down as a percentage of the maximum width. These new width values become the vane-space x coordinates to regress to, each paired with the y coordinate that corresponds to its progress along the rachis. The resulting curve is a polynomial or spline function that returns a width of the vane perpendicular to the shaft as a percentage of the vane's maximum width at a given progress along the rachis. This processed is summarized by algorithm 2.

After the curves are generated, they are stored to disk via writing the degree and coefficients per curve as feather dictionary entries. An estimate of the feather's real vane length is calculated using the rachis-start percentage and the feather length provided from The Feather Atlas. A difference between the vane lengths scaled down from centimeters to be a fraction of the feather length is calculated for comparison and may help identify errors in measurements from the Feather Atlas scans or placement of landmarks on the images. These new values, the calculated vane length and the difference, along with the rachis start in feather space and the maximum vane widths are additionally saved to the feather dictionaries. An example of a completed feather dictionary is table 4.2; the coefficients and other floating-point values are displayed in tables truncated for space in the document. Double-precision is used in the code and saved to file.

Decisions on polynomial representation during developing this regression portion of the project code were decided upon after testing on a variety of feather samples. In particular, the

Algorithm 2: Finding Per-Vane Width Functions

```
input  : Shaft curve, feather landmarks
output: Vane width functions
1  for each vane do
2      for each landmark do
3          Convert landmark coordinates to vane space.
4          Find corresponding x coordinate on shaft.
5          Find width between the shaft's and the current point's x coordinates.
6      end
7      Find maximum width.
8      Scale all widths by the maximum width.
9      Regress curve to those x coordinates paired with vane-space y coordinates.
10     Updated JSON files = Save curve and other data.
11 end
```

degrees of the shaft and vane curves were selected to be standard per curve type for all feathers, allowing further analysis through statistical comparison among multiple feathers but also preventing underfitting and overfitting of curves to the data. The shaft curve is represented by a degree-4 polynomial; degree-3 polynomials often underfit the shaft and similarly higher degrees overfit, skewing the shape rather than more accurately representing the shaft. The outer curves have more variation than shafts amongst flight feathers; for example, many primaries have a notch, a kink in the vane where it suddenly narrows towards the distal tip of the feather. The degrees need to be high enough to capture such detail but also low enough to not overfit the data. Cases of overfitting occurred when testing degrees 6 and 7; although these sometimes ideally reflected detail from the vanes, they strayed greatly from the outlines of feathers with simpler curves than others, such as secondaries compared to primaries. Frequent occurrences of underfitting happened using a degree of less than 5 as the curves did not capture subtle variations in the contour shape. Thus, 5-degree vane-outline polynomial curves were decided upon for the purposes in this thesis.

4.3 Data-Driven Generation of Feather Geometry

This section covers the application of collected data to 3D geometry generation similar to what Apteryx had achieved. As an overview of the entire pipeline for creating a procedural feather with a shape informed by real-world data, the three core stages of the are: (1) gathering feather

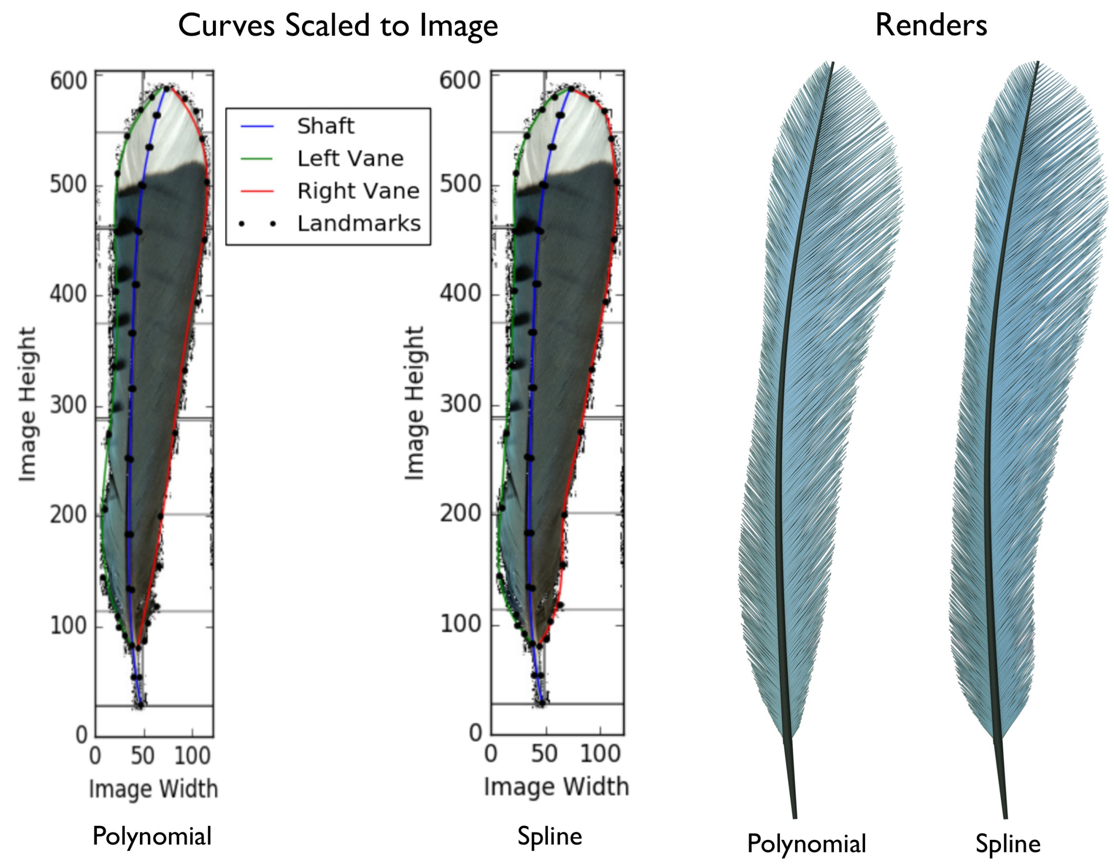


Figure 4.9: Polynomial versus spline regression for the outer vane curves used for drawing on the image and generating final corresponding renders for a Blue Jay's R4 feather.

scans, (2) storing the data provided with and extracting additional data from each scan, and (3) applying this data to guide the procedural generation of a computer-graphics feather. The previous section focused on stages (1) and (2), and this section concludes the process in describing stage (3).

4.3.1 System for Creation and Rendering of Curves

Maya 2018 with RenderMan 22 for Maya was chosen as the system for handling the modeling and rendering of the procedural feathers. The Maya API and Maya scripting commands are exposed for Python use [10]. A Python script drives the geometry generation, namely using the curve and pointOnCurve commands and setting rendering attributes for the procedurally created shapes.

The newest version of RenderMan, 22, supports rendering a large amount of curves and directly from Maya's NURBS and subdivision representations. Attributes that can be set per curve are the Pixar shader used for describing its surface and response to light and the width of the curve at the base and the tip. In this project, the PxrMarschnerHair material, which is an implementation of a popular hair-shading model, is applied to the barb and barbule curves [8]. For readily available materials, hair shading represents feather fibers the best though there is desire for a material model targeted for feathers; this idea is expanded in the later discussion on future work.

4.3.2 Incorporating Data into Geometry Generation

Data from the image-analysis process is incorporated into the generation of their geometry through reading the saved information for a feather and creating NURBS curve instances as the shaft and barbs with barbules.

The parameters for creating a feather are the name or type of the feather by which to retrieve the stored data, the curve widths for rendering the shaft with RenderMan (with the barb and barbule widths as fractions of these values), the density of barbs on the shaft given as a number of barbs per vane, the barbule density used based on the barbule lengths that fit along a barb's length, and the barb and barbule rotation angles from the tangent of their parents.

The remaining data needed in evaluating feather geometry comes from the feather information read from a file. These are the attributes from parsing The Feather Atlas metadata and the data extraction that follows: feather length in centimeters, rachis start, shaft curve polynomial,

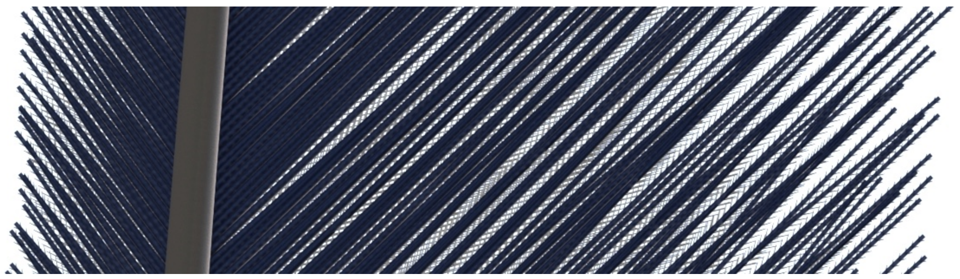


Figure 4.10: Geometry of a procedural feather containing barbs with barbules.

maximum width per vane, and width polynomial function per vane.

Creating the shaft curve for the procedural geometry concerns taking the polynomial form of the curve found by regression and creating a Maya NURBS curve from it. A polynomial rather than a spline was sufficient in representing the curvature of a feather's shaft during regression; therefore the knots accompanying a regressed spline do not have to be saved and passed into the Maya representation. Rather, 4 points are uniformly sampled from the shaft polynomial, evenly spaced by the percentage along the shaft. The values directly sampled from the regressed curve are in the normalized feather space and are next scaled by the feather length to values in centimeters. These points are passed as *edit points* (EPs) in generating a NURBS curve; edit points are used rather than control vertices as the curve would not exactly fit those points but rather be guided by them as CVs, and therefore not accurately recreate the regressed curve. When creating a curve with EPs, Maya establishes the appropriate control vertices and knot values when fitting a curve so that it passes directly through the given points.

Barbs with barbules are generated from the shaft, starting at the rachis-start value scaled by the length of the feather to the distal tip of the feather. For each barb, a progress along the rachis is found along with the corresponding point on the shaft to which the barb will be attached. The orientation axes (the tangent along the shaft, the normal perpendicular to the shaft and the vanes, and the bitangent orthogonal to the other 2 directions) are also calculated at that point. Using the width function for the vane to which the barb belongs, the width of the vane perpendicular to that shaft point is calculated and scaled by the feather length to a width in centimeter units. The length of the barb is then calculated based on the desired rotation using the barb angle from the shaft tangent about the normal towards the bitangent, and displacements along the tangent and bitangent are also calculated using the barb length and angle. Using these displacements, 4 edit

Algorithm 3: Generating Feather Geometry

```
input  : Feather name/type, curve rendering widths, barb and barbule densities, barb
         and barbule angles
output: A collection of NURBS curves in Maya
1  feather data = Set feather and read its attributes.
2  shaft curve = makeShaft(feather data)
3  for each vane do
4      makeBarbs(shaft curve, feather data)
5  end
6  Function makeShaft(feather data)
7      Uniformly sample points from shaft polynomial.
8      edit points = Scale points by feather length.
9      shaft curve = Create Maya NURBS curve with edit points.
10     return shaft curve
11 Function makeBarbs(shaft curve, feather data)
12     for each barb do
13         Find progress along rachis and point on shaft.
14         Retrieve shaft tangent, bitangent, and normal.
15         Calculate width of vane and barb length.
16         Find displacements (for rotation) along tangent and bitangent.
17         Generate 4 EPs for barb curve.
18         barb curve = Create Maya NURBS curve with EPs.
19         Compute barbule length and density.
20         makeBarbules(barb curve)
21     end
22 Function makeBarbules(barb curve)
23     for each barbule pair do
24         for each barbule in the pair do
25             Find progress along ramus and point on barb.
26             Retrieve barb tangent, bitangent, and normal.
27             Find displacements along tangent and bitangent.
28             Generate 2 EPs for barbule curve.
29             barbule curve = Create Maya NURBS curve with EPs.
30         end
31     end
```

points may be generated with the endpoints as the shaft point and the shaft point translated by the previous values along the tangent and bitangent directions; the distance between the endpoints is equal to the barb length. The other 2 EPs are placed in between the endpoints, translated by fractions of the displacement values. Small noise subtly distorts the other 3 edit points save the one on the shaft. These points are then used in creating a Maya curve as the shaft was done, now representing the tube-like ramus of a barb.

In preparation for the creation of barbules, the barbule length is computed as fraction of barb spacing, and a barbule density per barb is found based on the barbule length and the desired density as number of barbules per barbule length throughout the barb length. Starting from the base of the barb on the shaft towards the tip of the barb, 2 barbules that stem from the same location of the ramus are processed simultaneously. Each barbule pair per barb is generated similarly to the barb curves.

For each pair of barbules when processing a barb, the progress along the ramus, the corresponding barb point, and the orientation axes at that point are retrieved. Next the tangent and bitangent displacements are calculated using desired barbule length and rotation angle. Due to their small size, 2 endpoints will be the only points defining a barbule's curve, set as the barb point and the barb point displaced by the previous values in tangent and bitangent directions. As before, a NURBS curve is generated given these EPs. The only difference in the generation of both barbules in a pair is the barbule angle; one barbule is created by rotating from the barb tangent about the normal clockwise and the other counterclockwise. Figure 4.10 is a portion of rendered geometry with barbules.

Figures 4.11 and 4.12 are final renders of the geometry generated by the process described. Per image, the first input argument per image is the feather name, California Gull P7 feather ("CAGU_primary_adult_3") and Wild Turkey S7 feather ("WITU_secondary_female_1") respectively; the resulting feather shape reflects the image analysis process per feather that is illustrated in figure 4.8. The barb and barbule angles are set to 45 degrees, which rank within the ranges for such angles found in the zoological research presented earlier. The barb density is set to 500 barbs per vane, and the barbule density is 3 barbules for each barbule length within a barb's length (with barbule length set to 0.75 * barb spacing so the barbules touch as in real feathers). The curve rendering widths are small values starting with 0.1 to 0.01 centimeters for the shaft curve from calamus to distal tip, the barbs' one tenth those values, and the barbules' one tenth of the barbs'.

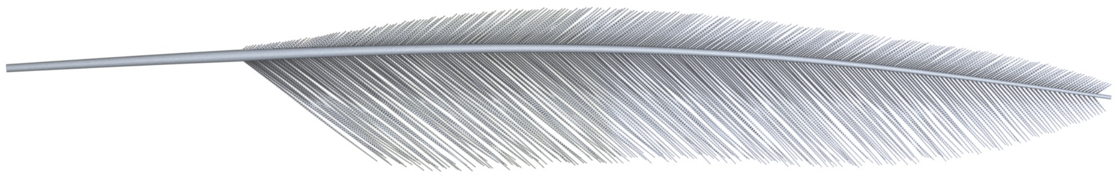


Figure 4.11: A final render using the shaft curve and width function polynomials from a gull feather.

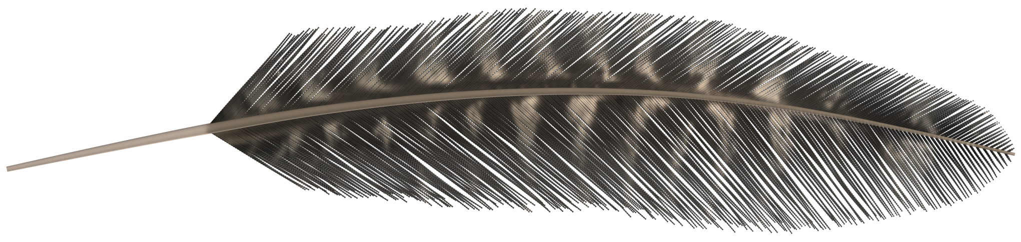


Figure 4.12: A final render using the shaft curve and width function splines from a turkey feather with the original image as a projected, diffuse texture.

4.3.3 Summary and Additional Details

The new geometry generation has a similar input and output behavior as Apteryx, but it was developed particularly to support incorporating measured and extracted data from real feathers. Multiple small details affect the final rendering, and notes on implementation decisions were noted throughout the description. Further ideas are commented below.

Although not measured during the image-analysis stage of this pipeline, decisions on barb angles influenced by observations of real feathers and information found in [32] and [33]. Rather than finding vane width from barb angle as in those studies, this thesis calculates barb length from the set angle and vane width found by the regressed outer-curve functions.

Extra information that could be extracted from the 52-landmark scheme includes the width of the shaft (as percentages in respect to the entire length of the feather such as what was done for the vane widths) at various progresses along the shaft. This width could be converted to use as curve widths for rendering; the values currently tested visually work well with the overall feather size based on centimeters but may not suit all feather samples.

Chapter 5

Discussion and Conclusions

In response to the demands for realism encountered in developing feather-generating software for production and considering the use of feathers in other fields of study, a pipeline was established for inputting a feather image, extracting useful information about its shape, using this information to drive the creation of geometry through 3D modeling software, and rendering the curves with hair-reflectance models. This chapter presents uses of the work from this thesis and areas for expansion.

5.1 Contributions to Research and Production

The new framework developed for this thesis stems from the experience of designing Apteryx and understanding where in research and production that feathers may be desired. The new software incorporates real-world data of feathers into the procedural generation of computer-graphics versions which is lacking in previous software in graphics research and production, including Apteryx.

5.1.1 Production

At Weta Digital, the development of Apteryx provided an artist-driven tool for creating feathers within the studio's production pipeline. Many parameters were exposed for the user to control, and although they numbered as a reduction compared to the amount of control in previous tools, automating and estimating attributes was still desired.

Production studios are interested in this inclusion of data as it provides an informed geometric model of a feather based on several parameters with no artist input. In my discussions with artists at Weta, I found that they would like to use presets of feathers; the new tool allows for automatically creating the detailed shapes of flight feathers which can serve as presets for those feather types, and these feathers could populate major sections of a bird with presets informed by type and placement. The AAM built during image analysis could allow generating plausible but different feathers.

5.1.2 Research

Previous computer-graphics research lacks providing shape information from real feathers to initialize the shape of a procedural feather but rather focuses on full control over several exposed guiding curves. Feather shape does vary widely across all species that have feathers, the type of the feather, and other characteristics such as stage of growth, but this thesis presents a framework to find patterns in these different categories of feathers.

Graphics research has also largely ignored the contribution of barbule geometry to the appearance of feathers, incorporating them into textures if at all. Adding barbules as a post-process after geometry generation but before rendering may improve computational efficiency, but developing a geometric feather model that incorporates the barbules better matches the real world and allows for expanded work in studying the near- and far-field reflectance of these components and physical simulation. Additionally, computer vision has not been applied much for analysis of the natural world where this thesis contributes to such an application combining computer-vision and -graphics techniques.

Other fields of study may be interested in using this work for further applications such as for paleontology, learning the shapes of extant bird feathers can help predict the shape appearance of fossilized, non-avian dinosaur feathers and information about their type and characteristics based on the shape and comparing to metadata of the similar extant birds analyzed. In the areas of physics and material science, barbule-level detail in a feather model can lead to simulation work in efforts to understand the physical properties of feathers.

5.2 Future Work

Areas for expanding the work in this thesis include collecting and extracting more data from real feathers, applying additional data to the geometry generation, improving rendering techniques, and increasing public involvement. Many of the ideas presented are interrelated in improving the structural and visual qualities of the procedural feathers.

5.2.1 Expanding Data Collection and Analysis

To increase the amount of data used, more feather scans from The Feather Atlas should be evaluated, ideally pulling information from each of the scans; and other resources containing images and metadata may be investigated such as federn.org and own imagery of feathers. Average statistics based on feather type, bird taxonomy, and other categories could be calculated from an Active Appearance Model built on a larger dataset.

The features extracted from the initial data may also be increased. For example, barb and barbule angles can be measured if access is available to full-resolution scans. Additionally, the 52-point scheme designed for this project has several landmarks bordering the plumaceous barbs near the calamus but are used only in finding the entire vane outlines rather than locating plumaceous regions. Although these barbs are not prominent on flight feathers, they often form larger portions of contour and semiplume feathers and entirely compose down feathers. Methods for extracting the location of plumaceous barbs on a variety of feathers should also be investigated.

Images from other perspectives rather than the dorsal and ventral sides of feather are desired; most of The Feather Atlas and federn.org samples capture the dorsal view, and none observe any perspective along different axes of the feathers. Such images would allow for extracting curve data along the Z axis of the feather, how it arches front and back.

Another method of incorporating data beyond The Feather Atlas scans would be to analyze and implement as much data as possible from zoological research such as the measurements described in the related work section. The parameter decisions used were influenced by these resources but can further be supported in the system such as barb spacing and barbule density.

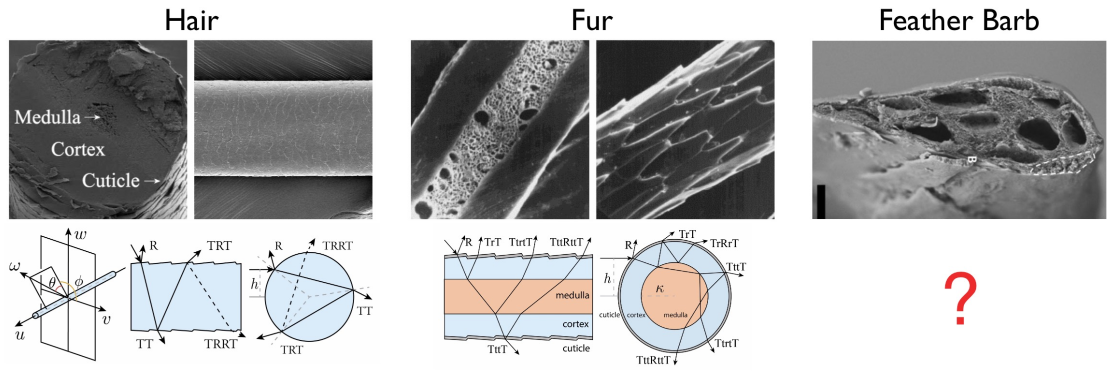


Figure 5.1: Hair, fur, and feather barb images and reflectance models modified from [69][42].

5.2.2 Further Applying Data

A key application of having statistics of feather shapes based on species and feather type is the layout of feathers in respect to each other and where they are placed on an organism's body. The goal may be to automatically generate feathers as a collection such as those on a wing that follows a scheme across species, entailing placing and slightly adjusting shapes of primaries, secondaries, their coverts, the alula, and contour feathers.

The arch in the feather shape to be analyzed through new images changes when the feather is pressed against other feathers versus standing alone. This especially applies to feathers with a more flexible shaft such as contours; flight feathers have stiff shafts but the bend may change slightly when encountering external sources such as wind. The slight arching in flight feathers was observed and implemented in the geometry-generation stage of this thesis, but analysis and application of this aspect of the shape should be investigated further.

5.2.3 Improving Rendering

Curve rendering is core area of increasing the visual fidelity of procedurally generated feathers. One approach not dissimilar to previous research is applying a texture to the geometry using the Feather Atlas images. During the generation of each barb and barbule, rachis and barb curve parameter values may be stored per curve along with their correspondence to locations in the original image based on the landmarks; these values may be used for finding texture coordinates to map portions of the image to each barb and barbule curve.

The most work in rendering to be desired is the development of a new reflectance model to better represent feathers and replace the hair shader. Models of hair and fur reflectance have often been used to shade feather curves at production studios although hair fibers and components of a feather differ structurally and optically. Figure 5.1 displays microscopic images and respective reflectance models for human hair (the popular Marschner model) and animal fur (an improved approach by Yan et al. [69] observing differences in the medulla component of the fiber more between human hair and animal fur) beside a microscopic cross section of a feather barb with an unknown model to match. Feathers are structured significantly differently from mammal hair and fur even at the micro- and nanoscopic levels, and they exhibit structural coloration in addition to pigment due to their unique structural composition and layers of keratin and melanin [62].

5.2.4 Increasing Public Involvement

Expanding this thesis also involves creating a publicly available Maya plug-in of the feather generator bundled with the collected feather data. Allowing other people worldwide to use the system will lead to improvements on artist usability and the extracted data. Additionally, efforts to increase the database itself entail reaching out and seeking to use the data gathered by other institutions and researchers, such as the high-resolution scans from Feather Atlas for capturing additional detail at the barb and barbule level.

5.3 Conclusions

The creation of feathers in computer graphics have been artistically driven and relied on user input to define their shape. Based on experience in visual-effects production through the development of procedural feather software, further realism was desired through the analysis of real-world data. This thesis presents a framework for analyzing and extracting information from real feathers to drive the procedural generation of geometry for computer-graphics feathers, and suggests interdisciplinary applications and expansions to further improve the results.

Bibliography

[1] Lumberjack. `https://www.wetafx.co.nz/research-and-tech/technology/lumberjack/`. Accessed: 2018-11-04.

[2] Wig. `https://www.wetafx.co.nz/research-and-tech/technology/wig/`. Accessed: 2018-11-04.

[3] (2015). The feather atlas. `https://www.fws.gov/lab/featheratlas/`. Accessed: 2018-01-17.

[4] (2018). Introducing json. `http://www.json.org`. Accessed: 2018-11-13.

[5] (2018). Katana: Lighting and look development. `https://www.foundry.com/products/katana`.

[6] (2018). Maya: Computer animation and modeling software. `https://www.autodesk.com/products/maya/overview`.

[7] (2018). NumPy reference. `https://docs.scipy.org/doc/numpy/reference/`. Accessed: 2018-11-03.

[8] (2018). Pxrmarschnerhair. `https://rmanwiki.pixar.com/display/REN22/PxrMarschnerHair`. Accessed: 2018-11-18.

[9] (2018a). Python. `https://www.python.org`. Accessed: 2018-11-13.

[10] (2018b). Python in maya. `https://knowledge.autodesk.com/support/maya/learn-explore/caas/CloudHelp/cloudhelp/2018/ENU/Maya-Scripting/files/GUID-C0F27A50-3DD6-454C-A4D1-9E3C44B3C990-htm.html`. Accessed: 2018-11-16.

[11] (2018). Renderman docs: Curves. `https://rmanwiki.pixar.com/display/REN22/Curves`. Accessed: 2018-11-11.

[12] (2018). SciPy reference. `https://docs.scipy.org/doc/scipy/reference/`. Accessed: 2018-11-03.

[13] Akenine-Möller, T., Haines, E., Hoffman, N., Pesce, A., Iwanicki, M., and Hillaire, S. (2018). *Real-Time Rendering*. CRC Press, 4 edition.

[14] Andrus, C. and Manca, M. (2015). Furtility: Robust hair styling. In *ACM SIGGRAPH 2015 Talks*, SIGGRAPH '15, pages 74:1–74:1, New York, NY, USA. ACM.

[15] Baele, X. and Warzee, N. (2006). Real time l-system generated trees based on modern graphics hardware. In *Real Time L-System Generated Trees Based on Modern Graphics Hardware*, IEEE International Conference on Shape Modeling and Applications (SMI).

[16] Beucher, S. (2010). Image segmentation and mathematical morphology. `http://cmm.ensmp.fr/~beucher/wtshed.html`. Accessed: 2018-10-02.

[17] Beucher, S. and Lantuejoul, C. (1979). Use of watersheds in contour detection. In *In International Workshop on Image Processing: Real-time Edge and Motion Detection/Estimation*, volume 132.

[18] Bishop, C. M. (2006). *Pattern Recognition and Machine Learning.* Springer.

[19] Booth, J., Roussos, A. T., Zafeiriou, S., Ponniahy, A., and Dunaway, D. (2016). A 3d morphable model learnt from 10,000 faces. pages 5543–5552.

[20] Bowline, S. D. and Kačić-Alesić, Z. (2011). Dynamic, penetration-free feathers in rango. In *ACM SIGGRAPH 2011 Talks*, SIGGRAPH '11, pages 35:1–35:1, New York, NY, USA. ACM.

[21] Broggi, J., Gamero, A., Hohtola, E., Orell, M., and Nilsson, J.-. (2011). Interpopulation variation in contour feather structure is environmentally determined in great tits. *PLoS One*, 6(9).

[22] Catmull, E. and Clark, J. (1978). Clark, j.: Recursively generated b-spline surfaces on arbitrary topological meshes. computer-aided design 10(6), 350-355. *Computer-Aided Design*, 10:350,355.

[23] Catmull, E. and Rom, R. (1974). A class of local interpolating splines. *Computer Aided Geometric Design - CAGD*, 74.

[24] Chandler, A. C. (1916). *A study of the structure of feathers, with reference to their taxonomic significance*, volume v.13:no.11 (1916). Berkeley, University of California press. https://www.biodiversitylibrary.org/bibliography/15062 — Cover title. — "April 17, 1916." — Also published as author's thesis (PH.D.) University of California, 1914.

[25] Chen, Y., Xu, Y., Guo, B., and Shum, H.-Y. (2002). Modeling and rendering of realistic feathers. In *Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '02, pages 630–636, New York, NY, USA. ACM.

[26] Choi, W., Kim, N., Jang, J., Kim, S., and Yang, D. (2017). Build your own procedural grooming pipeline. In *ACM SIGGRAPH 2017 Talks*, SIGGRAPH '17, pages 54:1–54:2, New York, NY, USA. ACM.

[27] Cootes, T., Taylor, C., Cooper, D., and Graham, J. (1995). Active shape models - their training and application. *Computer Vision and Image Understanding*, 61:38–59.

[28] Cootes, T. F., Edwards, G. J., and Taylor, C. J. (1998). Active appearance models. In *Computer Vision—ECCV'98*, pages 484–498. Springer.

[29] Dove, C. and Koch, S. (2011). Microscopy of feathers: A practical guide for forensic feather identification. *The Microscope*, 59(2).

[30] Eliason, C. M. (2014). *Mechanisms and evolution of iridescent feather colors in birds.* PhD thesis. Copyright - Database copyright ProQuest LLC; ProQuest does not claim copyright in the individual underlying works; Last updated - 2016-11-22.

[31] Fascione, L., Hanika, J., Leone, M., Droske, M., Schwarzhaupt, J., Davidovič, T., Weidlich, A., and Meng, J. (2018). Manuka: A batch-shading architecture for spectral path tracing in movie production. *ACM Trans. Graph.*, 37(3):31:1–31:18.

[32] Feo, T., Field, D., and Prum, R. (2015). Barb geometry of asymmetrical feathers reveals a transitional morphology in the evolution of avian flight. *Proceedings in Royal Society B.*

[33] Feo, T. J. and Prum, R. O. (2014). Theoretical morphology and development of flight feather vane asymmetry with experimental tests in parrots. *Journal of Experimental Zoology*, 322B:240–255.

[34] Franco, C. G. and Walter, M. (2002). Modeling and rendering of individual feathers. In *Proceedings of the 15th Brazilian Symposium on Computer Graphics and Image Processing*, SIBGRAPI '02, pages 293–299, Washington, DC, USA. IEEE Computer Society.

[35] Green, M., Fransen, R., Gray, D., and Kranz, B. (2018). Hair today, cloth tomorrow: Automating character fx on peter rabbit. In *ACM SIGGRAPH 2018 Talks*, SIGGRAPH '18, pages 11:1–11:2, New York, NY, USA. ACM.

[36] Greenewalt, C. H. (1960). *Hummingbirds*. Doubleday.

[37] Hanson, T. (2011). *Feathers: The Evolution of a Natural Miracle*. Basic Books.

[38] Harris, M. (2017). How framestore, mpc, image engine, rodeo fx created the fantastic beasts of jk rowlings new film. *Digital Arts*.

[39] Harvey, T. A. (2012). *Spatially- and Directionally-varying Reflectance of Milli-scale Feather Morphology*. PhD thesis.

[40] Heckenberg, D., Gray, D., Smith, B., Wills, J., and Bone, C. (2011). Quill: Birds of a feather tool. In *ACM SIGGRAPH 2011 Talks*, SIGGRAPH '11, pages 34:1–34:1, New York, NY, USA. ACM.

[41] Herzog, K. (1968). *Anatomie und Flugbiologie der Vogel*. G. Fischer.

[42] Igic, B., D'Alba, L., and Shawkey, M. D. (2016). Manakins can produce iridescent and bright feather colours without melanosomes. *Journal of Experimental Biology*, 219(12):1851–1859.

[43] Keane, K. (2007). Nga manu birds. *Te Ara - the Encyclopedia of New Zealand*. Accessed: 2018-09-28.

[44] Leaning, J. and Fagnou, D. (2010). Feathers for mystical creatures: Creating pegasus for clash of the titans. In *ACM SIGGRAPH 2010 Talks*, SIGGRAPH '10, pages 52:1–52:1, New York, NY, USA. ACM.

[45] Long, J. and Schouten, P. (2008). *Feathered Dinosaurs: The Origin of Birds*. CSIRO Publishing.

[46] Marschner, S. and Shirley, P. (2016). *Fundamentals of Computer Graphics*. CRC Press, 4 edition.

[47] Marzluff, J. M. (2007). *In the Company of Crows and Ravens*. Yale University Press.

[48] Mech, R. and Prusinkiewicz, P. (2003). Generating subdivision curves with l-systems on a gpu. In *Generating subdivision curves with L-systems on a GPU*, ACM SIGGRAPH.

[49] Nastase, E. (2016). Pennaceous feather structure. `http://www.emilynastase.com/fauna/`. Accessed: 2018-08-01.

[50] Newport, M. (2005). Start to finish feathers solution. Master's thesis, N.C.C.A. Bournemouth University.

[51] Peterson, S. (2004). Animating puss in boots' feather in shrek 2. In *ACM SIGGRAPH 2004 Sketches*, SIGGRAPH '04, pages 41–, New York, NY, USA. ACM.

[52] Pickrell, J. (2014). *Flying Dinosaurs: How Fearsome Reptiles Became Birds*. Columbia University Press.

[53] Prum, R. and Williamson, S. (2001). Theory of the growth and evolution of feather shape. *Journal of Experimental Zoology*, 291:30–57.

[54] Prusinkiewicz, P. and Hanan, J. (1989). *Lindenmayer Systems, Fractals, and Plants*, volume 79 of *Lecture Notes in Biomathematics*. Springer-Verlag.

[55] Rijke, A. M. (1968). The water repellency and feather structure of cormorants, phalacrocoracidae. *Journal of Experimental Biology*.

[56] Robertson, B. (2010). In fine feather. *Computer Graphics World*. Accessed: 2018-11-01.

[57] Robertson, C. J. R., editor (1985). *Complete Book of New Zealand Birds*. Reader's Digest.

[58] Rozenberg, G. and Salomaa, A. (1974). *L Systems*. Lecture Notes in Computer Science. Springer-Verlag.

[59] Rzankowski, T. (2017). Feather tools. `https://www.sidefx.com/tutorials/author/trzanko/`.

[60] Sartore, J. and Strycker, N. (2018). *Birds of the Photo Ark*. National Geographic.

[61] Scott, D. and McFarland, C. (2010). *Bird Feathers*. A Guide to North American Species. Stackpole Books.

[62] Shawkey, M. D., D'Alba, L., Xiao, M., Schutte, M., and Buchholz, R. (2015). Ontogeny of an iridescent nanostructure composed of hollow melanosomes. *Journal of Morphology*, 276(4):378–384.

[63] Singh, G. (2018). Introduction to regression splines (with python codes). *Analytics Vidhya*. Accessed: 2018-11-11.

[64] Streit, L. and Heidrich, W. (2002). A biologically-parameterized feather model. In *Eurographics 2002*, volume 21. The Eurographics Association and Blackwell Publishers.

[65] Strett, L. M. (2004). *Modelling of feather coat morphogenesis for computer graphics*. PhD thesis. Copyright - Database copyright ProQuest LLC; ProQuest does not claim copyright in the individual underlying works; Last updated - 2016-05-23.

[66] Tao, Y., Chen, Z., and Griffis, C. L. (2004). Chick feather pattern recognition. *IEE Proceedings - Vision, Image and Signal Processing*, 151(5):337–344.

[67] Tekiela, S. (2014). *Feathers: A Beautiful Look at a Bird's Most Unique Feature*. Adventure Publications.

[68] Voitkevich, A. A. (1966). *The Feathers and Plumage of Birds*. October House.

[69] Yan, L.-Q., Tseng, C.-W., Jensen, H. W., and Ramamoorthi, R. (2015). Physically-accurate fur reflectance: Modeling, measurement and rendering. *ACM Trans. Graph.*, 34(6):185:1–185:13.

[70] Yan, X., Yang, B., Zhang, W., Liu, C., and Wang, Y. (2016). An improved denoising algorithm of feather and down image based on ksvd. In *2016 8th International Conference on Information Technology in Medicine and Education (ITME)*, pages 419–423.